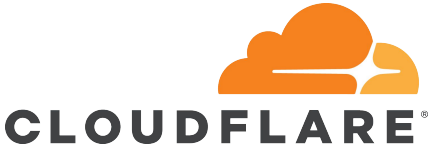


Network Automation with Salt and NAPALM

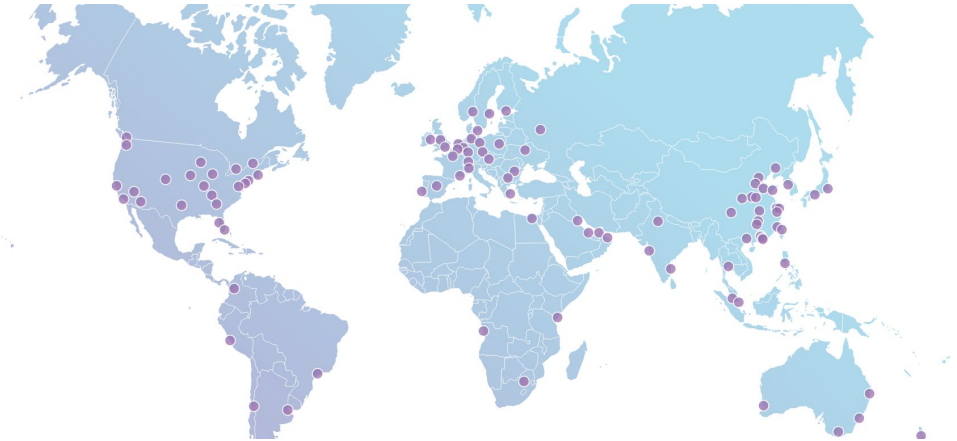
A self-resilient network

Mircea Ulinic
Cloudflare, London

APRICOT 2017
Ho Chi Minh City, VN

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?

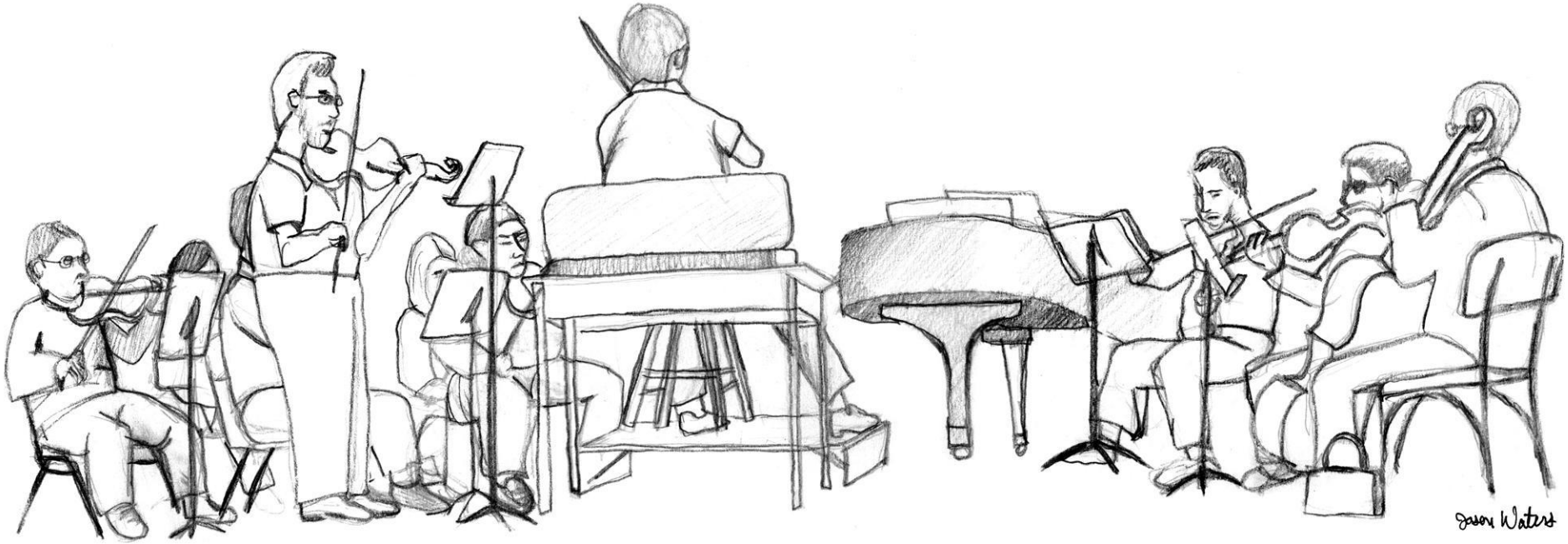
- Deploy new PoPs
- Human error factor
- Replace equipment
- Monitor
- Much faster recovery

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



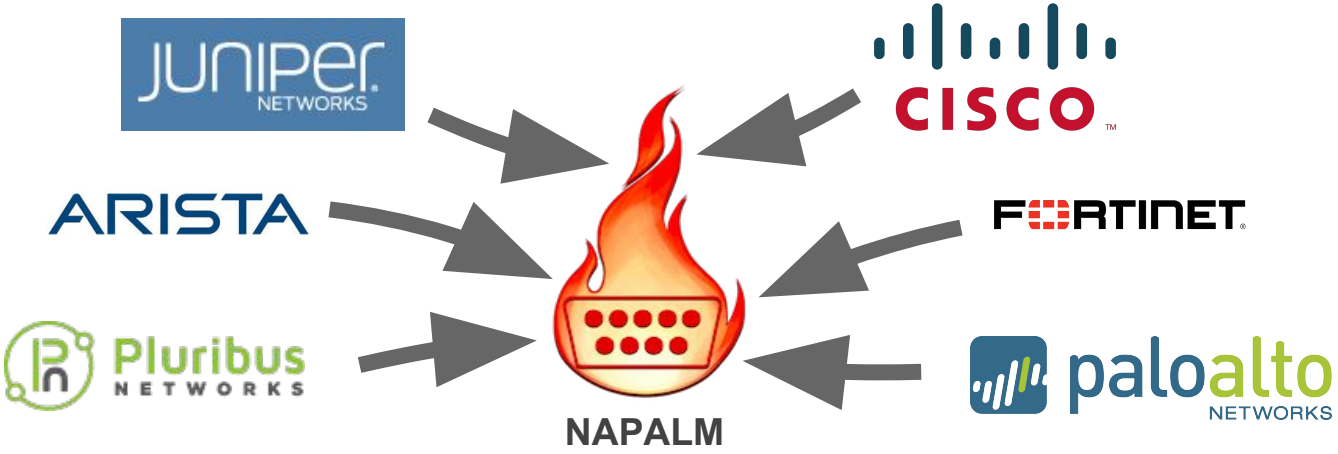
Orchestrator E.g.: Salt	**Configuration management only** E.g.: Ansible
● Long standing sessions ● Real-time job ● Job scheduling ● REST API ● High Availability ● GPG encryption ● Pull from Git, SVN	● open/close session per module ● Real-time job (Tower: $$) ● Job Scheduling (Tower: $$) ● REST API (Ansible Tower: $$) ● HA (Ansible Tower: $$) ● Security (Tower: $$) ● Pull from Git, SVN (Tower, $$)

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

NAPALM integrated in SaltStack

NETWORK AUTOMATION: NAPALM

Beginning with 2016.11.0, network automation is inclued by default in the core of Salt. It is based on the NAPALM library and provides facilities to manage the configuration and retrieve data from network devices running widely used operating systems such as: JunOS, IOS-XR, eOS, IOS, NX-OS etc. - see the complete list of supported devices.

The connection is established via the `NAPALM proxy`.

In the current release, the following modules were included:

- `NAPALM grains` - Select network devices based on their characteristics
- `NET execution module` - Networking basic features
- `NTP execution module`
- `BGP execution module`
- `Routes execution module`
- `SNMP execution module`
- `Users execution module`
- `Probes execution module`
- `NTP peers management state`
- `SNMP configuration management state`
- `Users management state`

https://docs.saltstack.com/en/develop/topics/releases/2016.11.0.html

Network automation in two steps

1. Install
2. Use

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 *-I “type:router”* ntp.**set_peers** 10.1.130.10 10.1.130.18 10.1.130.22

Targeting minions: https://docs.saltstack.com/en/latest/topics/targeting/index.html

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
```

Abstracted

```
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

Us

```
vi /etc/salt/pillar/edge01_sgn01.sls
```

```
proxy:
  driver: VendorA
  proxytype: napalm
  host: edge01.sgn01
  username: apricot
  passwd: xxxx
```

```
proxy:
  driver: VendorB
  proxytype: napalm
  host: edge01.sgn01
  username: apricot
  passwd: xxxx
```

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

A self-resilient network

Monitoring carriers (transit providers)

```
mircea@re0.edge01.sgn01> 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
http://www.cisco.com/c/en/us/td/docs/ios/ipsla/command/reference/sla_book/sla_02.html

How many probes?

```
$ sudo salt-run transits.probes show_count=True

Generated 7248 probes.
```

Generated using:

- net.ipaddrs
- net.interfaces
- bgp.neighbors
- bgp.config

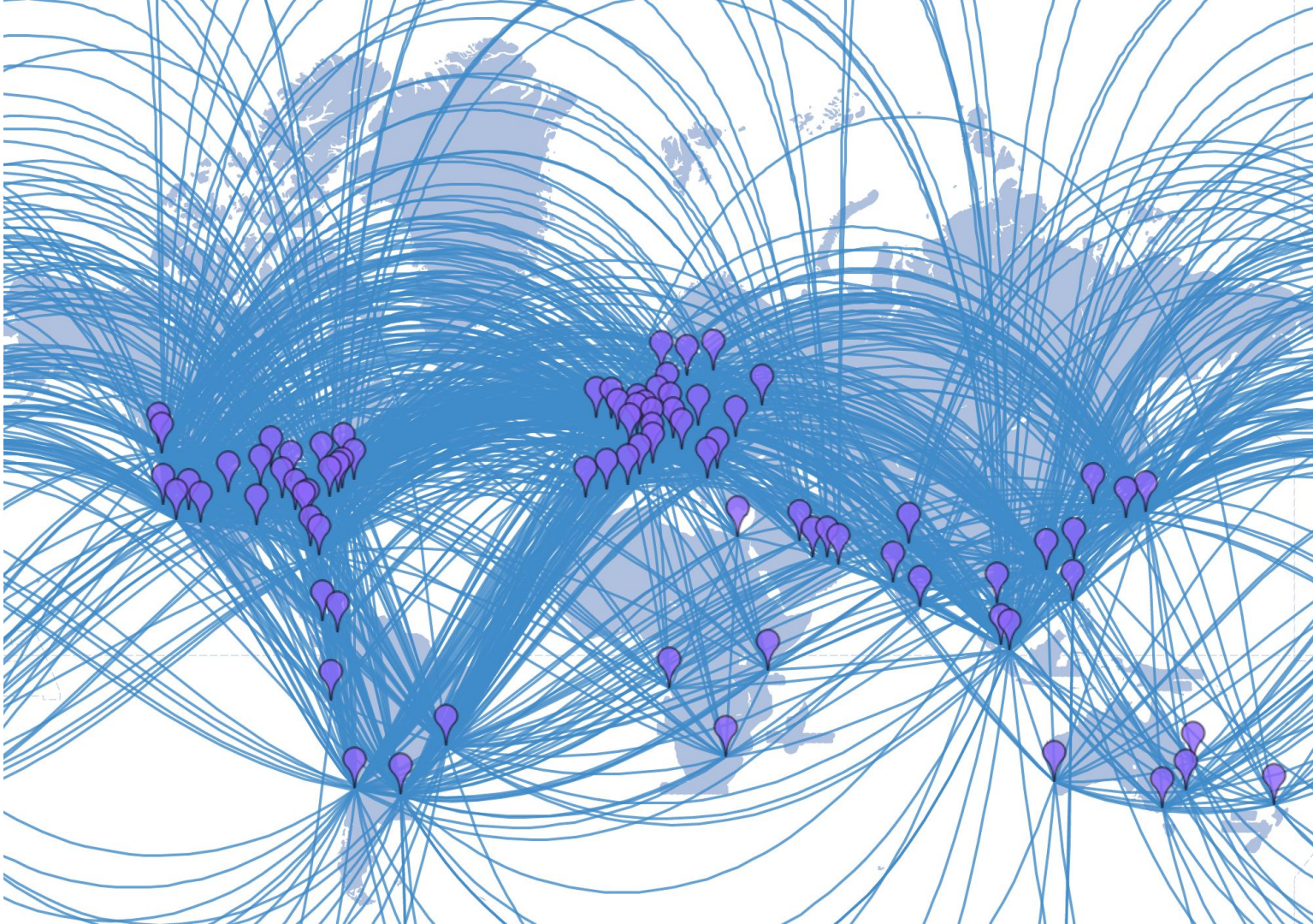
All integrated by default in SaltStack.

How are they installed?

```
$ cat /etc/salt/pillar/probes_edge01_sgn01.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
```

```
$ sudo salt 'edge*' state.sls router.probes
edge01.sgn01:
----------
          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

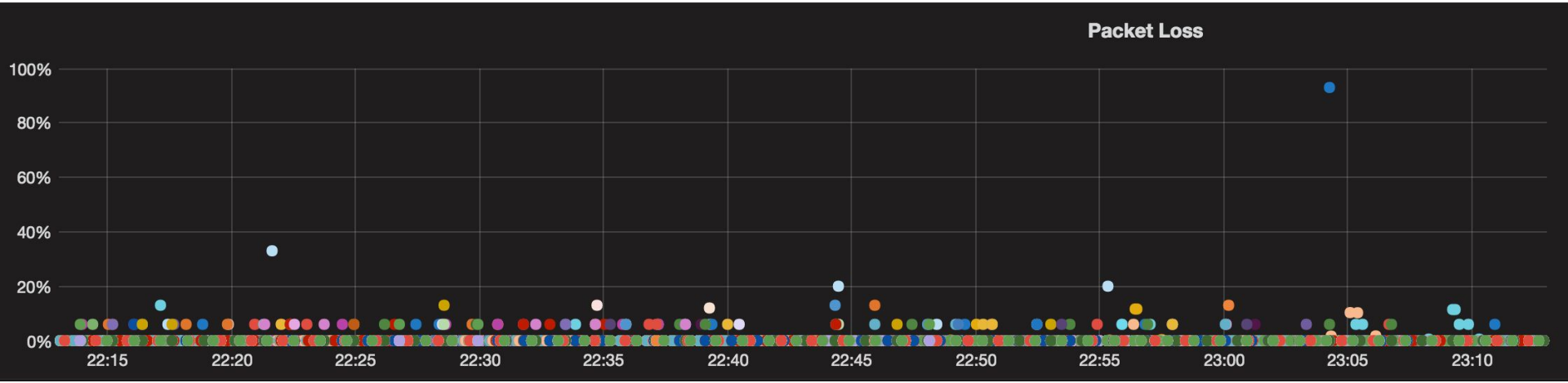


Retrieving probes results

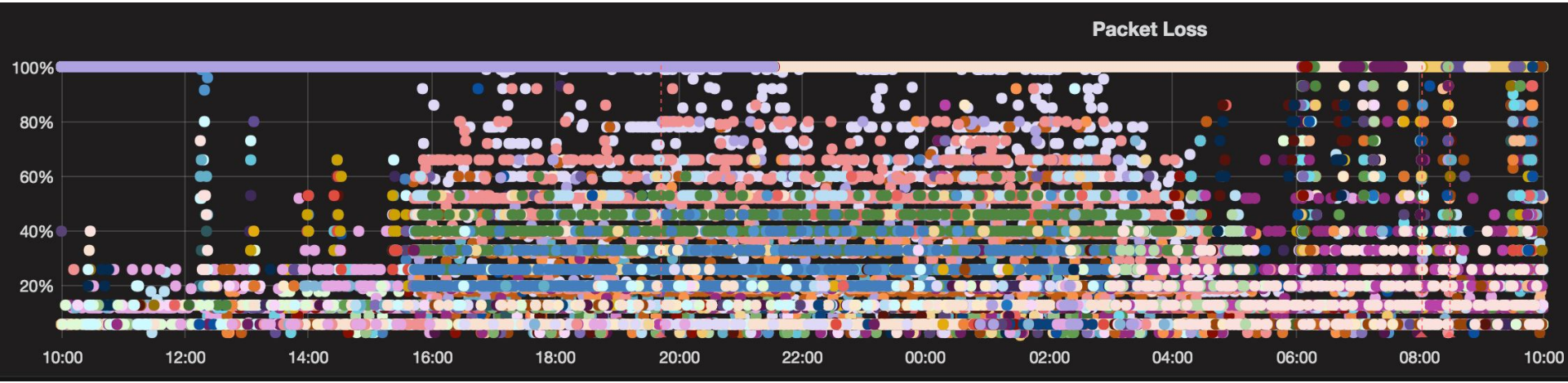
```
$ sudo salt ‘edge*’ probes.results

edge01.sgn01:
    ----------
    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
```

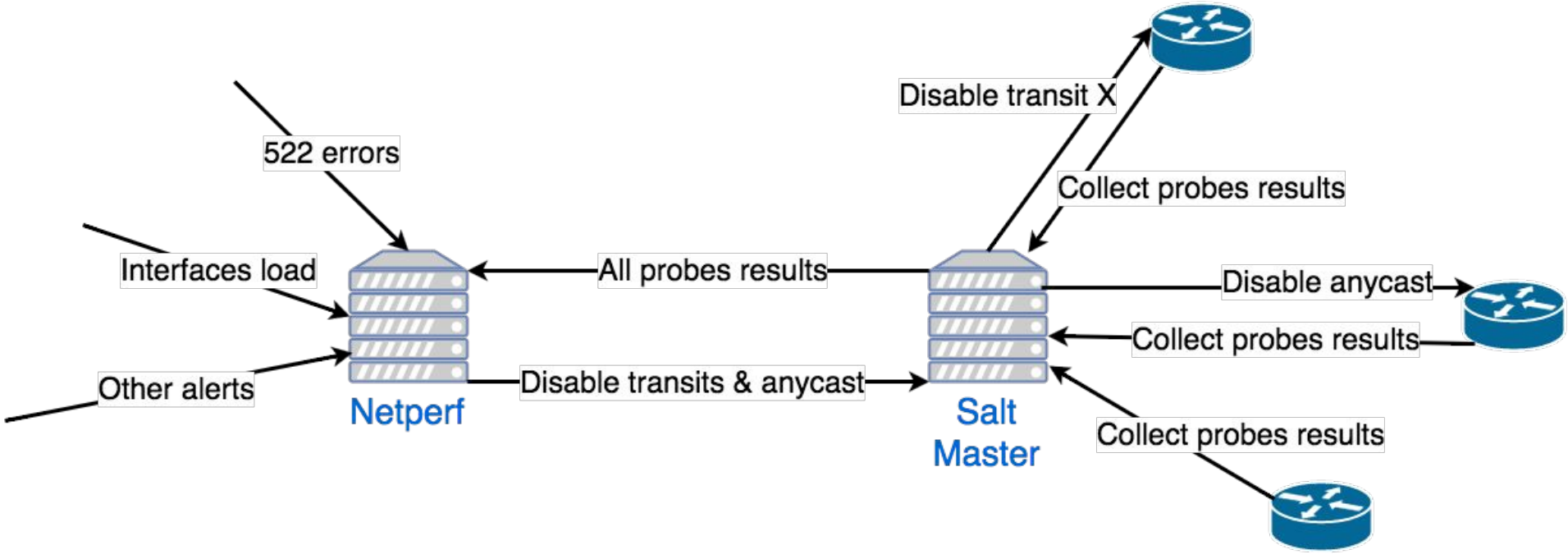
The Internet (during a good day)



But usually the Internet looks like that



Self-resilient network



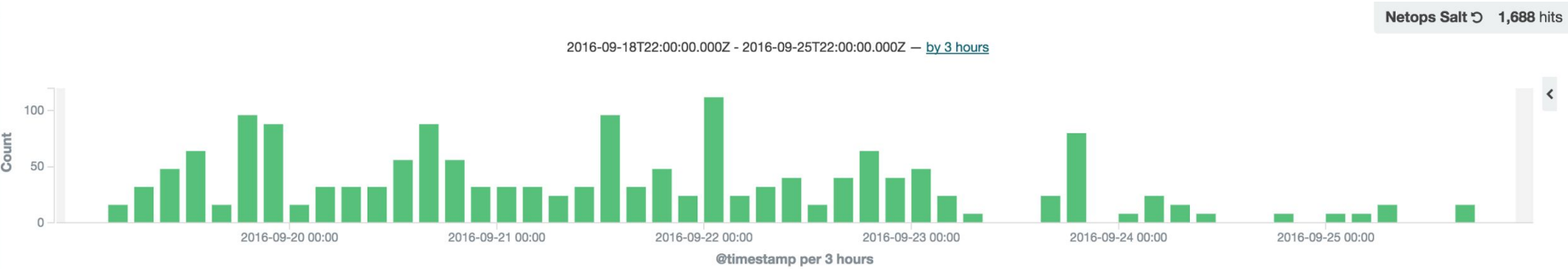
Self-resilient network: HipChat alerts

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:
https://github.com/napalm-automation/napalm-salt

How can you contribute?

GitHub

- 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

CLOUDFLARE