SALTSTACK (FOR SYS ADMINS)

THEO BASCHAK

MUUG 2015-02-10

INTRO

OVERVIEW

- Terminology
- Things possible with SaltStack
- SaltStack docs
- How I use it

TERMINOLOGY

- Topology:
  - Master
  - Minions
- States (/srv/salt)
- Pillars (/srv/pillar)
- Grains
- Returners

THINGS POSSIBLE

- Templating using Jinja
  - Re-use/Template: Less states is simpler
  - Distribution abstraction (apt+yum)
- Standalone Minions
- Return results to CouchDB directly
- Since Python: write your own code
- Salt Cloud
  - ec2, Rackspace, DigitalOcean, Proxmox
  - OpenStack, vSphere, MS Azure, Linode
  - To name a few, more in the docs

SALTDOCS

salt.readthedocs.org/en/latest/

- Very good, useful examples
- Built from main source

HOW I USE IT

- Package installation and configuration
- Remote Command Execution (Intentional!)
- Performing Mass Upgrades
- Distributed troubleshooting
- Deploy new nameserver in under 2 minutes
- Storing periodic nagios and network checks in CouchDB
- I store/backup my States and Pillars to Git
- I’m barely scratching the surface

STANDARD SALT STUFF

STATES N PILLARS

- `salt -v '*' state.highstate`; #refreshes all states on all minions
- `salt -v '*' saltutil.refresh_pillar`; #refreshes pillars on all minions
- `salt '*' nagios.run_pillar ciscodude_services`; #run some nagios checks defined in pillar

RETURNERS

- `salt '*' network.traceroute 8.8.8.8 --return couc`

SYS.DOC

- `salt <minion_id> sys.doc`
  - Shows all modules available, and options for each
- `grep` is handy

USAGE EXAMPLES

INSTALLATION

```
/srv/salt/top.sls
base:
  'os:debian':
    - match: grain
    - settings.ntp.debian
    - settings.fail2ban.debian
    - settings.apt.cron-apt.debian

  'G@os:debian and G@city:winnipeg':
    - match: compound
    - settings.apt.apt-proxy.debian
```

INST AND CONFIG

```
 /srv/salt/settings/ntp/debian.sls
ntp:
  pkg:
    - installed
  service:
    - running
    - require:
      - pkg: ntp
    - watch:
      - file: /etc/ntp.conf

/etc/ntp.conf:
  file:
    - managed
    - source: salt://settings/ntp/ntp.conf
    - require:
      - pkg: ntp
```

CONFIG (CONT)

```
 /srv/salt/settings/ntp/ntp.conf
driftfile /var/lib/ntp/ntp.drift
statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable
server time.mbix.ca iburst
server ntp.torix.ca iburst
server 2.debian.pool.ntp.org iburst
server 3.debian.pool.ntp.org iburst
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery
restrict 127.0.0.1
restrict ::1
```

CMD.RUN

- `salt -G apt:true cmd.run 'apt-get -s dist-upgrade`

```
ns2.henchman21.net:
    Reading package lists...
    Building dependency tree...
    Reading state information...
    0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ns0.ciscodude.net:
    Reading package lists...
    Building dependency tree...
    Reading state information...
    0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
...
```

MASS UPGRADES

MASS UPGRADES

- Safe, systematic way:
    - `salt '*' pkg.refresh_db`
    - `salt '*' cmd.run 'apt-get -s dist-upgrade'`
    - `salt '*' pkg.upgrade`
- Or just one specific package:
    - This was handy for HeartBleed and Bash
    - `salt '*' pkg.install bash refresh=True`
    - `salt '*' pkg.install openssl refresh=True`
    - `salt '*' pkg.install libc6 refresh=True`
    - `salt '*' service.restart nginx`

1 SYSTEM

```
salt secure.ciscodude.net pkg.upgrade
```

```
secure.ciscodude.net:
    ----------
    changes:
        ----------
        prosody:
            ----------
            new:
                0.9.7-1~wheezy1
            old:
                0.9.6-1~wheezy2
    comment:

    result:
        True
```

CONCLUSION

THE END

- TRY IT!
- Presentation source/download available at github