Configuration Management with SaltStack

Act II

Arnold Bechtoldt

Berlin, 14.05.14

Topics

inovex

1. Configuration Management Systems
2. SaltStack Fundamentals
3. SaltStack Inside
4. Conclusions
5. Showcase/ Walkthrough

About me

arnold.bechtoldt@inovex.de

inovex

- Linux-Systems Engineer at inovex GmbH
- Develop lots of features for (Open Source) Datacenter Management
- Provisioning of physical & virtual infrastructure
- SaltStack user since December, 2012 (~ v0.10.x)

About inovex

inovex.de

inovex

- Provides a wide set of IT services:
  - Application Development
  - Mobile Development
  - Business Intelligence
  - IT Engineering & Operations
  - Consulting
  - Trainings

- Cool projects with great Open Source Software
- Teams of high-experienced engineers
- We have excellent job offers in Karlsruhe, Cologne, Munich and Pforzheim!

inovex

Configuration Management Systems

(a.k.a. CMS)

Configuration Management Systems

Definition

inovex

- Support **building** a defined infrastructure
- Support **managing** a defined infrastructure
- Definition of infrastructure in code (“Infrastructure as code”)
- *Configuration Management requires Software Development*

Configuration Management Systems

Traditional

inovex

1. Create a user (if needed): *postfix*
2. Install a package (or more): *postfix, postfix-pcre*
3. Create or change a file (configure a service): */etc/postfix/main.cf*
4. Enable and start the service: *chkconfig postfix on; service postfix start*

Configuration Management Systems

Limitations

1. User Management
2. Package Management
3. File Management
4. Service Management

What if we need m o r e ?

SaltStack Fundamentals

Why Salt?

inovex

Salt …

1. … is extremely flexible.
2. … is very easy to use.
3. … has lots of exciting features.
4. … is fast.
5. … makes sysadmin’s life easier.

Why Salt? (2)

inovex

- It's all about (simple) data
- Central place for configuration
- Asynchronous (send commands to 10,000 server at a time in seconds)
- Configuration management
- Remote execution
- Core functions are available as execution modules
- Hundreds of state + execution modules
- Easy to extend
- Separate data and code easily with pillars

inovex

SaltStack Inside

SaltStack Inside

Terminology

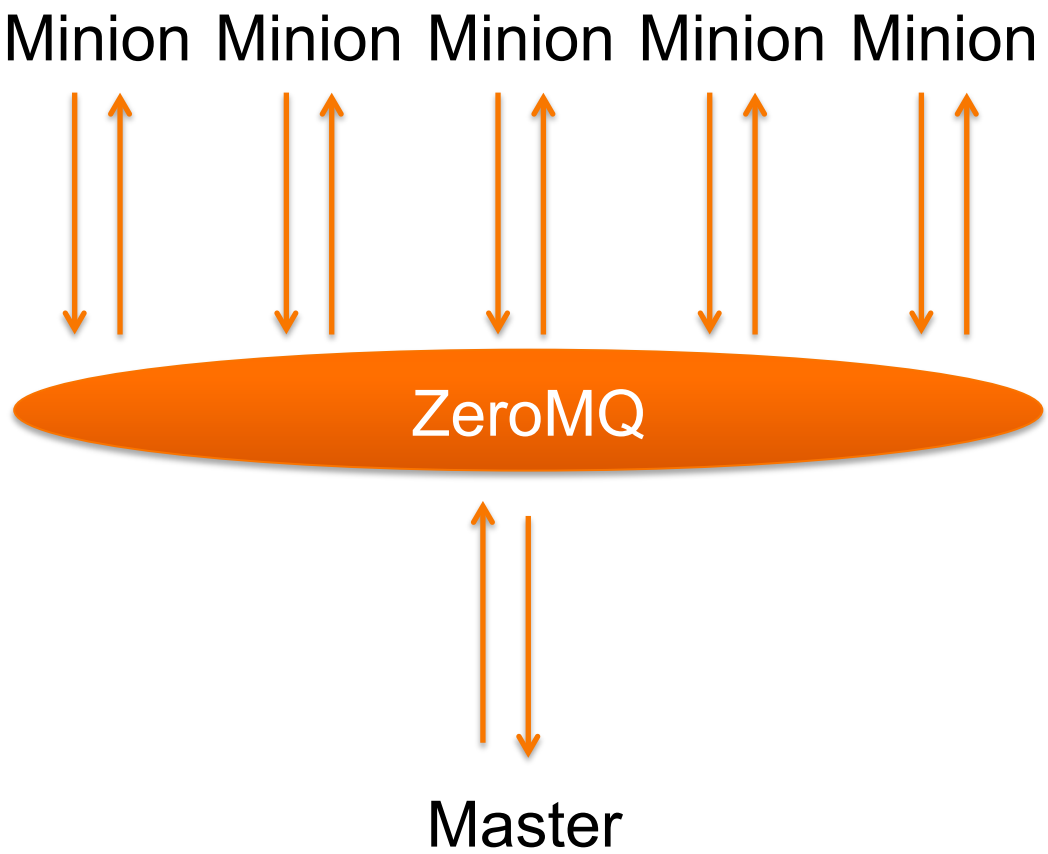
inovex

Different software, different names:

- Minion: The client itself
- Master: Manages minions
- Grains: Standard set of client system information
- Pillars: User-defined set of information
- State: User-defined description of a state of a file, package, …
- Formulas: Collection of user-defined states
- State Module: Set of state functions for files, packages, LVM, MySQL, …
- Execution Module: Predefined commands executed on the minions
- Jinja: Default template renderer

SaltStack Inside

Default Architecture



SaltStack Inside

Minion Targeting

inovex

You specify minion targeting to apply states, pillars or commands to a desired set of minions:

- Globbing: feweb*.domain.local, *.domain.local, feweb[1-3].domain.local
- PCRE: fe(web|mail)1.domain.local
- Grains: ‘os:CentOS’, ‘saltversion:2014.1.1’
- Pillars: ‘role:mailserver’, ‘cluster_name:fehomepage’
- Lists: feweb1.domain…, feweb2.domain…, feweb3.domain…
- Nodegroups: Predefined list of minions
- Compound (Mix): Mix of the above targeting types (operators: and, or, not)
- Batch Size: 4, 10% (execute on X minions at a time)

The Top Files

inovex

Components using a top file:

- States
- Pillars

What they do:

- Map minions with states
- Map minions with pillars
- Map minions with environments

The Top Files (2)

inovex

Top of States

```
dev:
  'mailserver*dev*':
    - postfix.satellite
qa:
  'mailserver*qa*':
    - postfix.satellite
prod:
  'mailserver*prod*':
    - postfix.satellite
    - monitoring
```

Top of Pillars

```
dev:
  'mailserver*dev*':
    - postfix.dev
qa:
  'mailserver*qa*':
    - postfix.qa
prod:
  'mailserver*prod*':
    - postfix.prod
    - monitoring.prod
```

SaltStack Inside

States

```
postfix:
    pkg:
        - installed
        - names:
            - postfix
            - postfix-pcre
    service:
        - running
        - watch:
            - file: /etc/postfix/main.cf
```

Dict/ Hash:	State ID
List/ Array:	State Module
Any:	Parameters

States (2)

```
/etc/postfix/main.cf:
    file:
        - managed
        - source: salt://postfix/files/satellite.main.cf
        - user: root
        - group: postfix
        - mode: 640
        - template: jinja
```

SaltStack Inside

Pillars

```
type: satellite
relayhost: smtp.domain.local
inet_protocols:
  - ipv4
soft_bounce: True
postscreen:
  - greylisting
  - pregreet
  - dnsbl
mynetworks: 127.0.0.0/8 [::ffff:127...
```

```
postscreen_dnsbl_sites:
  - zen.spamhaus.org*2
  - ix.dnsbl.manitu.net*2
  - dnsbl.sorbs.net=127.0.0.[2;3;5;6;7;9;10]
  - list.dnswl.org=127.0.[0..255].0*-1
  - list.dnswl.org=127.0.[0..255].[2..3]*-3
any:
  generic:
    list:
      - foo: oof
        bar: rab
```

SaltStack Inside

Fileserver Backends

inovex

Store top files, states (formulas), templates, custom modules, pillars, etc. on

- Local filesystems
- Git Repositories
- SVN Repositories
- Mercurial Repositories
- MinionFS (distributed over several hosts)
- Amazon S3

Separate them by

- Environments/ teams
- Projects
- Pillars
- …

SaltStack Inside

Data Access

inovex

Access data by:

- Pillars: {{ salt['pillar.get']('inet_protocols', ['ipv4', 'ipv6']) }}
- Grains: {{ salt['grains.get']('os_family') }}
- Peer Publish: {{ salt['publish.publish']('web*', 'grains.item', 'fqdn') }}
- Mine: {{ salt['pillar.item']('mine_functions:network.interfaces') }}
- Local env variables: {% set foo = 'bar' %} {{ foo }}
- Deserializing: load_json('file.json') / load_yaml('file.yaml') / …
- …

These are available in:

- Top Files
- State Files
- Template Files
- Pillar Files
- …

SaltStack Inside

Configuration Management + Remote Execution

inovex

One tool to rule them all:

- $ salt '*' state.sls ferm saltenv=prod
- $ salt '*' state.highstate test=False
- $ salt '*' gem.install foreman_provision
- $ salt '*' hadoop.dfs ls /
- $ salt '*' lxc.unfreeze bigfoot
- $ salt '*' network.traceroute inovex.de
- $ salt '*' pkg.install openssl refresh=True
  $ salt '*' service.restart nginx
- $ salt '*' dockerio.pull index.docker.io:MyRepo/image foo

- $ salt '*' tomcat.deploy_war salt://application.war /api yes http://localhost:8080/

- $ salt –C 'I@role:mailserver and (P@os:Debian or S@192.168.42.0/24)' …

SaltStack Inside

Configuration Management + Remote Execution (2)

```
feweb1.domain.local:
----------
          ID: ferm
    Function: pkg.installed
      Result: True
     Comment: All specified packages are already installed.
     Changes:
----------
          ID: ferm
    Function: file.managed
        Name: /etc/ferm/ferm.conf
      Result: True
     Comment: File /etc/ferm/ferm.conf updated
     Changes:
              ----------
              diff:
---
@@ -1,33 +1,31 @@
...
----------
          ID: ferm
    Function: service.running
      Result: True
     Comment: Service restarted
     Changes:
              ----------
              ferm:
                  True
Summary
------------
Succeeded: 3
Failed:    0
------------
Total:     3
```

Conclusions

Conclusions

IMHO

inovex

1. Choose the CMS which fits to your project, everyone is different
2. If you spend more time creating automation instead of saving it, something is wrong
3. Salt can help you managing large and complex infrastructures
4. SaltStack can do even more than CM: Salt-Cloud, Salt-Virt, Salt SSH, Salt Proxy, …
5. Salt can help you making your customers and yourself happy

Showcase/ Walkthrough

Code at Github.com

inovex

Basic configuration:

- /bechtoldt/network-formula
- /bechtoldt/time-formula

DNS/ DHCP:

- /bechtoldt/binddns-formula
- /bechtoldt/iscdhcp-formula

Lifecycle Management (physical + virtual servers):

- Foreman: /bechtoldt/foreman-formula

Cloud management:

- OpenNebula: /bechtoldt/opennebula-formula
- OpenStack: /EntropyWorks/salt-openstack/tree/formula

Thank You

Questions?

inovex

Contact

Arnold Bechtoldt
Linux-Systems Engineer

inovex GmbH
Office Karlsruhe
Ludwig-Erhard-Allee 6
D-76131 Karlsruhe

+49 (0)173 31 81 117
arnold.bechtoldt@inovex.de