Saltstack Workshop

e.mouzeli@logicea.net

August 19, 2016

Basic Packages

- Create file `/etc/salt/states/default/init.sls`:

```
thebase:
  pkg.installed:
    - install_recommends: False
    - pkgs:
      - wget:
      - curl:
      - sysstat:
      - ethtool:
      - screen:

openssh-server:
  pkg.installed

an_ssh_bla:
  pkg.installed:
    - name: openssh-server
```

- Run: `salt 'firefly' state.sls defaults`

Create pillar data

- Create file `/etc/salt/pillars/users/init.sls`

```
users:
  inara:
    fullname: Inara Serra
    email: inara@serenity.com
    home: /home/inara
    shell: /bin/bash
    groups:
      - root
    pub_keys:
      - ssh-rsa <snip>6Qd9dNjBpkEW0lJt1NXKQo3==
    enabled: True
```

- Run: `salt 'firefly' pillar.data`

Create pillar data

- Create file **/etc/salt/pillars/users/init.sls**

```
users:
  inara:
    fullname: Inara Serra
    email: inara@serenity.com
    home: /home/inara
    shell: /bin/bash
    groups:
      - root
    pub_keys:
      - ssh-rsa <snip>6Qd9dNjBpkEW0lJt1NXKQo3==
    enabled: True
```

- Run: `salt 'firefly' pillar.data`

- Create file **/etc/salt/pillars/top.sls**:

```
base:
  '*':
    - users
```

- Run again: `salt 'firefly' pillar.data`
- Go ahead and create another user

Create pillar data

- Create the following `location` pillar data and include it in `top.sls`:

```
location:
  name: firefly
  domain: serenity.com
  dns:
    - 8.8.8.8
    - 8.8.6.6
  root: /opt/serenity
```

Create pillar data

- Create the following `location` pillar data and include it in `top.sls`:

```
location:
  name: firefly
  domain: serenity.com
  dns:
    - 8.8.8.8
    - 8.8.6.6
  root: /opt/serenity
```

- Lastly create some `elasticsearch` pillar data and add it in `top.sls` only for your host:

```
elasticsearch:
  config:
    cluster.name: {{salt['grains.get']('elasticsearch:
    cluster', "kaylee")}}
    node.name: {{salt['grains.get']('fqdn')}}
    node.master: true
    node.data: true
    bootstrap.mlockall: true
    transport.tcp.compress: true
```

Running states - Create users

- Create and run user state, using the following:

```
{% for user,info in salt['pillar.get']('users').items() %}
{{user}}:
  user.present:
    - fullname: {{ info['fullname'] }}
    - shell: {{ info['shell']|default("/bin/true") }}
    - home: {{ info['home']|default("/home/%s" % user) }}
    - groups:
      {% for group in info['groups']|default([]) %}
      - {{ group }}
      {% endfor %}
```

Running states - Create users

- Create and run user state, using the following:

```
{% for user,info in salt['pillar.get']('users').items() %}
{{user}}:
  user.present:
    - fullname: {{ info['fullname'] }}
    - shell: {{ info['shell']|default("/bin/true") }}
    - home: {{ info['home']|default("/home/%s" % user) }}
    - groups:
      {% for group in info['groups']|default([]) %}
      - {{ group }}
      {% endfor %}
  {% if 'pub_keys' in info %}
  ssh_auth:
    - present
    - user: {{ user }}
    - names:
  {% for pub_ssh_key in info['pub_keys']|default([]) %}
      - {{ pub_ssh_key }}
  {% endfor %}
    - require:
      - user: {{ user }}
  {% endif %}
{% endfor %}
```

Running states - Create users (Notes)

- Salt's state system will first render all pillar data before rendering any *.sls files.

Running states - Create users (Notes)

- Salt's state system will first render all pillar data before rendering any *.sls files.
- `|default("some_value")` instructs the jinja renderer to apply this value if the variable we requested does not exist.

Running states - Create users (Notes)

- Salt's state system will first render all pillar data before rendering any *.sls files.
- `|default("some_value")` instructs the jinja renderer to apply this value if the variable we requested does not exist.
- `("/home/%s" % user)` works as well as python's `"/home/{0}".format(user)`

Running states - Create users (Notes)

- Salt's state system will first render all pillar data before rendering any *.sls files.
- `|default("some_value")` instructs the jinja renderer to apply this value if the variable we requested does not exist.
- `("/home/%s" % user)` works as well as python's `"/home/{0}".format(user)`
- `ssh_auth` is a salt module that will take care of creating an `authorized_keys` file under a user's `.ssh/` diretory, import any public keys found in the user's pillar data and lastly, apply the appropriate permissions `(600)`.

Running states - Install elasticsearch

- Create the elasticsearch state (**`init.sls`**) and use our elasticsearch pillar data as config options for **`elasticsearch.yml`**

```
elasticsearch.yml:
  file.managed:
    - name: /etc/elasticsearch/elasticsearch.yml
    - source: salt://elasticsearch/elasticsearch.yml.j2
    - template: jinja
    - context:
      elasticsearch: {{salt['pillar.get']('elasticsearch')}}
    - backup: minion
    - makedirs: True
```

Running states - Install elasticsearch

- Create the elasticsearch state (**`init.sls`**) and use our elasticsearch pillar data as config options for **`elasticsearch.yml`**

```
elasticsearch.yml:
  file.managed:
    - name: /etc/elasticsearch/elasticsearch.yml
    - source: salt://elasticsearch/elasticsearch.yml.j2
    - template: jinja
    - context:
      elasticsearch: {{salt['pillar.get']('elasticsearch')}}
    - backup: minion
    - makedirs: True
```

- Create the source file salt's templating engine will render to produce **`elasticsearch.yml`**. Check the file URI in the **`source`** parameter

Running states - Install elasticsearch

- Create the elasticsearch state (**init.sls**) and use our elasticsearch pillar data as config options for **elasticsearch.yml**

```
elasticsearch.yml:
  file.managed:
    - name: /etc/elasticsearch/elasticsearch.yml
    - source: salt://elasticsearch/elasticsearch.yml.j2
    - template: jinja
    - context:
      elasticsearch: {{salt['pillar.get']('elasticsearch')}}
    - backup: minion
    - makedirs: True
```

- Create the source file salt's templating engine will render to produce **elasticsearch.yml**. Check the file URI in the **source** parameter

```
{# Alternative syntax is elasticsearch['config'] #}
{%- for config, value in elasticsearch.config.items() %}
{{config}}: {{value}}
{%- endfor %}
```

Running states - Install elasticsearch

◦ Create the elasticsearch state (**`init.sls`**) and use our elasticsearch pillar data as config options for **`elasticsearch.yml`**

```
elasticsearch.yml:
  file.managed:
    - name: /etc/elasticsearch/elasticsearch.yml
    - source: salt://elasticsearch/elasticsearch.yml.j2
    - template: jinja
    - context:
      elasticsearch: {{salt['pillar.get']('elasticsearch')}}
    - backup: minion
    - makedirs: True
```

◦ Create the source file salt's templating engine will render to produce **`elasticsearch.yml`**. Check the file URI in the **`source`** parameter

```
{# Alternative syntax is elasticsearch['config'] #}
{%- for config, value in elasticsearch.config.items() %}
{{config}}: {{value}}
{%- endfor %}
```

◦ We need the file **`elasticsearch.yml`** to be present before the package's installation (it's a debian thing).

Running states - Install elasticsearch

- Continue working on the elasticsearch state:

```
elasticsearch:
  pkgrepo.managed:
    - humanname: Elasticsearch Official Debian Repository
    - name: deb http://packages.elasticsearch.org/
    elasticsearch/1.5/debian stable main
    - key_url: http://packages.elasticsearch.org/GPG-KEY-
    elasticsearch
    - file: /etc/apt/sources.list.d/elasticsearch.list
```

Running states - Install elasticsearch

- Continue working on the elasticsearch state:

```
elasticsearch:
  pkgrepo.managed:
    - humanname: Elasticsearch Official Debian Repository
    - name: deb http://packages.elasticsearch.org/
    elasticsearch/1.5/debian stable main
    - key_url: http://packages.elasticsearch.org/GPG-KEY-
    elasticsearch
    - file: /etc/apt/sources.list.d/elasticsearch.list
  pkg:
    - installed
    - hold: True
    - require:
      - pkgrepo: elasticsearch
      - file: elasticsearch.yml
```

Running states - Install elasticsearch

- Continue working on the elasticsearch state:

```
elasticsearch:
  pkgrepo.managed:
    - humanname: Elasticsearch Official Debian Repository
    - name: deb http://packages.elasticsearch.org/
    elasticsearch/1.5/debian stable main
    - key_url: http://packages.elasticsearch.org/GPG-KEY-
    elasticsearch
    - file: /etc/apt/sources.list.d/elasticsearch.list
  pkg:
    - installed
    - hold: True
    - require:
      - pkgrepo: elasticsearch
      - file: elasticsearch.yml
  service:
    - running
    - enable: True
    - require:
      - pkg: elasticsearch
    - watch:
      - file: elasticsearch*
```

Running states - Install elasticsearch (Notes)

- Two states are not permited to share the same name unless they call different salt modules.

Running states - Install elasticsearch (Notes)

- Two states are not permited to share the same name unless they call different salt modules.
- The `require` parameter in the `pkg` state will ensure that the `pkgrepo` state named “elasticsearch” and the `file` state named “elasticsearch.yml” will run before installing the “elasticsearch” package.

Running states - Install elasticsearch (Notes)

- Two states are not permited to share the same name unless they call different salt modules.
- The `require` parameter in the `pkg` state will ensure that the `pkgrepo` state named “elasticsearch” and the `file` state named “elasticsearch.yml” will run before installing the “elasticsearch” package.
- The `watch` parameter in the `service` state tells salt to restart the service named “elasticsearch” if any `file` state who’s name matches the expression `elasticsearch*`. In our example, if there are any changes on `elasticsearch.yml`, the elasticsearch service will restart.

Make your elasticsearch state great again

◦ Elasticsearch requires java, and we want to use Oracle Java. Go on and ensure that Oracle java is installed before installing elasticsearch. What you need is:

```
PPA: deb http://ppa.launchpad.net/webupd8team/java/ubuntu
    xenial main
Keyid:  EEA14886
keyserver: keyserver.ubuntu.com
Package name: oracle-java8-installer
```

◦ For reasons beyond the scope of this workshop, you must include the following salt module in your state:

```
accept_license:
  debconf.set:
    - data:
        'shared/accepted-oracle-license-v1-1': {'type': '
    boolean', 'value': True}
```

YOUR SALT
IS TOO. DAMN. SALTY
memegenerator.net