Using the power of Salt alongside SUSE Manager 3 – Lessons Learned

SUSECON19

Andreas Linden (COOP), Bo Jin (SUSE)
Nashville TN, USA, 1-5.04.2019
Agenda

• Who
• Salt basics
• Why we are doing this
• Our goals
• How we do it
• Configuration-Management
• Software-Management
• Risks
• Lessons Learned
whoami
About COOP

Number of employees

<table>
<thead>
<tr>
<th>Retail</th>
<th>Wholesale / Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>46,637</td>
<td>38,364</td>
</tr>
</tbody>
</table>

Number of stores and markets

<table>
<thead>
<tr>
<th>Retail</th>
<th>Wholesale / Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>2,250</td>
<td>226</td>
</tr>
</tbody>
</table>
Do you know Salt?
Salt

SALT...?!

– Central Systems Management

Salt is a new approach to infrastructure management built on a dynamic communication bus. Salt can be used for data-driven orchestration, remote execution for any infrastructure, configuration management for any app stack, and much more.
Salt

SALT...?!

- **Salt Master (SUSE Manager 3)**

  Central management system. This system is used to send commands and configurations to the Salt minion that is running on managed systems.
Salt

SALT...?!

- Salt Minion

Managed system. This system runs the Salt minion which receives commands and configuration from the Salt master.
Salt

- **Execution Modules**

Ad hoc commands executed from the command line against one or more managed systems. Useful for:

- Real-time monitoring, status, and inventory
- One-off commands and scripts
- Deploying critical updates

![Diagram showing Salt architecture with Master connecting to Minions across various operating systems: Linux, Solaris, FreeBSD, Windows, and OS X.](image-url)
Salt

SALT...?!

- **Formulas / States**

A declarative or imperative representation of a system configuration.
Salt

SALT...?!

- Grains

System variables. Grains are static information about the underlying managed system and include operating system, memory, and many other system properties. You can also define custom grains for any system.
Salt

SALT...?!

- The Top File

Matches formulas and Salt pillar data to Salt minions.
Why?!
Why we are doing this

Let me tell you a story…

– We needed to patch SLES Linux on IBM Power (no Novell ZCM agent for ppc64 available)
– Prevent configuration drift
– (Lessons Learned ) Automation Plattform
– (Lessons Learned ) Dokument/Reproduceability of komplex installations
Our goals
Goals

Benefits…

– System equality
– Prevent configuration drift
– Be flexible and scalable
– Automate complex installations
– Continuing quality
Suse Manager as salt-master with git

Challenges...

- Knowledge development (salt, jinja, python)
- Different approach
  - Think global
  - Niche solutions
  - No manual changes
- Collaboration with other departments
  - "Getting everyone on the same page"
- Salt-Version
  - Is dictated by SUSE → (Lessons Learned) → quite Actual now → Thank you Joe W.😊
How we do it
How we do it:

Suse Manager as Salt-Master with git …
How we do it

What you need...

- **Salt-Master ✓**
- **Install a git-client (GitPython or pygit2) (Lessons Learned pygit2!)**
  Configuration `/etc/salt/master`

```python
fileservr_backend:
  - git
  - roots

gitfs_provider: pygit2

gitfs_remotes:
  - https://git.hs.coop.ch/scm/salt/coopstack.git

# if self-signed certificate
gitfs_ssl_verify: False
```

- **RUN: salt-run fileservr.update**
How we do it

Why not using the Suse Manager "State Catalog" functionality …?

- There are a lot of things you can not do in the gui
- Transparent change system with GIT
- (Lessons Learned) Single point of Configuration
- (Lessons Learned) GitOps is awesome!
How we do it

Good to know…

Disabled runners:
- Add /your/_runners_dir to /etc/salt/master.d/susemanager.conf
- Salt-Warning: Topfile merging:
  - Change top_file_merging_strategy: same in /etc/salt/master
- Remove all top.sls from /srv/salt/ → on every update of the SuSE-Manager(…)

- (Lessons Learned) Change filesserver_limit_traversal: True in /etc/salt/master → if you have a lot of state files.
Configuration-Management
Configuration-Management

A greenfield project?!!…
Configuration-Management

Grains as a filter...

/etc/salt

grains

CORPORATION: COOP
CSBVERS: 3.1
DMZ: none
ENVIRONMENT: ENTW
FLAGS:
  - none
HIGHSTATE: true
LOCATION: basel
PILOT:
  - none
SOFTWAREGROUPS:
  - hybris_base
  - JVM_100
Configuration-Management

Topfile filtering...

- top.sls

/etc/salt/grains (minion)

---

```yaml
base:  
  # '*':  
  # - 01_GLOBAL.01_GLOBAL_STATES

  'CORPORATION:COOP':  
    - match: grain  
      - 02_COOP.01_COOP_STATES

  'G@CORPORATION:COOP and G@cpuarch:x86_64':  
    - match: compound  
      - 02_COOP.01_COOP_STATES.02_X86_64

  'G@CORPORATION:COOP and G@cpuarch:x86_64 and G@ENVIRONMENT:ENTW':  
    - match: compound  
      - 02_COOP.02_COOP_ENTW_STATES.02_X86_64
```
Configuration-Management

Topfile filtering...

```bash
*@
'G@CORPORATION:COOP'

'G@CORPORATION:COOP and G@cpuarch:ppc64'

'G@CORPORATION:COOP and G@cpuarch:x86_64'

'G@CORPORATION:COOP and G@cpuarch:ppc64 and G@ENVIRONMENT:ENTW'

'G@CORPORATION:COOP and G@cpuarch:ppc64 and G@ENVIRONMENT:TEST'

'G@CORPORATION:COOP and G@cpuarch:ppc64 and G@ENVIRONMENT:PROD'

'G@CORPORATION:COOP and G@cpuarch:x86_64 and G@ENVIRONMENT:ENTW'

'G@CORPORATION:COOP and G@cpuarch:x86_64 and G@ENVIRONMENT:TEST'

'G@CORPORATION:COOP and G@cpuarch:x86_64 and G@ENVIRONMENT:PROD'
```
Configuration-Management

Topfile filtering...

GIT://yourgit.internal.net/yourstack.git

- top.sls
- 01_GLOBAL
- 02_COOP
- 03_CERTS
- 04_CONFS
- _modules
- _scripts
- _states

- 01_PPC64
- 02_X86_64
- 03_PPC64LE

- init.sls
- CM_*_100.sls
- CM_*_100.sls
- CM_*_100.sls
- init.sls
- CM_*_100.sls
- init.sls
- CM_*_100.sls
- init.sls
- SW_*_100.sls
- init.sls
- SW_*_100.sls
- init.sls
- SW_*_100.sls

*template
Configuration-Management

include...
Configuration-Management

Running timed highstates...

- CM_salt_hs_100.sls

/etc/salt/grains

```python
{% if grains['HIGHSTATE'] == true %}
HighstateJOB:
  schedule.present:
    - function: state.highstate
    - minutes: 60
    - maxrunning: 1
{% else %}
HighstateJOB:
  schedule.absent: []
{% endif %}
```

- CORPORATION: COOP
- CSBVERS: 3.0
- DMZ: none
- ENVIRONMENT: TEST
- FLAGS:
  - none
- HIGHSTATE: true
- LOCATION: basel
- PILOT:
  - none
- SOFTWAREGROUPS:
  - hybris_base
Software-Management
Software-Management

Grain for software...

1. Activation grain

```yaml
CORPORATION: COOP
CSBVERS: 3.0
DMZ: none
ENVIRONMENT: TEST
FLAGS:
  - none
HIGHSTATE: true
LOCATION: basel
PILOT:
  - none
SOFTWAREGROUPS:
  - hybris_base
```

2. init.sls

```bash
include:
  # Include CM_* states here
  - .CM_SAMPLE_100
  # Include SW_* states here
  - .SW_hybris_101

3. Software state

```bash
# Hybris Base installation
{% if 'hybris_base' in grains['SOFTWAREGROUPS'] %}
include:
  - .JAVA_100
  - .hybris_101
{% endif %}
```
Risks (Buhhh...
Dave: Hallo, HAL. Do you read me, HAL?
HAL: Affirmative, Dave. I read you.
Dave: Open the pod bay doors, HAL.
HAL: I'm sorry, Dave. I'm afraid I can't do that.
Code of Conduct

You better watch it...!

- Use clear logical names for your structure
- Select minions in the order given by structure (top down)
- Clearly separate state file types by name (e.g. CM_* / SW_* )
- Document the state owner (header)
- Follow the correct integration path
Lessons Learned – Disk state
Lessons Learned

Disk state

Disk State configuration file

```
epaper_diskmgmt:
  epaper_appl:
    epaper_dev: '/dev/sdc'
    epaper_devnr: '1'
    epaper_lvpath: '/dev/app/epaper-appl'
    epaper_vgname: 'app'
    epaper_lvname: 'epaper-appl'
    epaper_lsize: '5G'
    epaper_mountp: '/usr/local/imagotag'
  epaper_log:
    epaper_dev: '/dev/sdc'
    epaper_devnr: '1'
    epaper_lvpath: '/dev/app/epaper-log'
    epaper_vgname: 'app'
    epaper_lvname: 'epaper-log'
    epaper_lsize: '2G'
    epaper_mountp: '/var/log/imagotag'
  epaper_import:
    epaper_dev: '/dev/sdc'
    epaper_devnr: '1'
    epaper_lvpath: '/dev/app/epaper-import'
    epaper_vgname: 'app'
    epaper_lvname: 'epaper-import'
    epaper_lsize: '2G'
    epaper_mountp: '/var/imagotag/import'
```
Lessons Learned

Disk state

Import the Config

{% set statepath = sls.split('.')[:-1] | join('/') %}
{% import_yaml statepath + '/files/disk_properties_coreservice.yml' as epaper_disks %}
Lessons Learned

Disk state

Check all needed parameters

```sql
{% for disks, devices in epaper_disks.iteritems() %}
  {% for key, value in devices.items() %}
    {% if value.get('epaper_dev') != False %}
    {% if value.get('epaper_devnr') != False %}
    {% if value.get('epaper_lvpath') != False %}
    {% if value.get('epaper_vgname') != False %}
    {% if value.get('epaper_lvname') != False %}
    {% if value.get('epaper_lvsize') != False %}
    {% if value.get('epaper_mountp') != False %}
```
Lessons Learned

Disk state

Create partition if not exists

```python
{% if not salt['partition.exists']('value.server_dev + value.server_devnr') %}
# NEW PARTED FIX
parted bugfix for {{ value.server_dev }} - {{ value.server_lxpath }}:
  module.run:
    - name: creapar.cr_table
    - device: {{ value.server_dev }}

Generating Partition for {{ value.server_lxpath }}:
  module.run:
    - name: partition.mkpart
    - device: {{ value.server_dev }}
    - part_type: primary
    - fs_type: ext2
    - start: 2560s
    - end: 100%

Set disk flag for {{ value.server_lxpath }}:
  module.run:
    - name: partition.set_id
    - device: {{ value.server_dev }}
    - minor: 1
    - system_id: {{ value.server_partid }}
{% endif %}
```
Lessons Learned

Disk state

Create PV – VG – LV if not exists

```plaintext
# pvcreate
{% if salt['cmd.retcode']['pvdisplay'] + value.server_dev + value.server_devnr) != 0 %}
Create pv for {{ value.server_lvpath }}:
  lvm.pv_present:
    - name: {{ value.server_dev }}{{ value.server_devnr }}
{% endif %}

# vgcreate
{% if salt['cmd.retcode']['vgdisplay'] + value.server_vgname) != 0 %}
Create vg for {{ value.server_lvpath }}:
  lvm.vg_present:
    - name: {{ value.server_vgname }}
    - devices: {{ value.server_dev }}{{ value.server_devnr }}
{% endif %}

# lvcreate
{% if salt['cmd.retcode']['lvdisplay'] + value.server_lvpath) != 0 %}
Create lv for {{ value.server_lvpath }}:
  lvm.lv_present:
    - name: {{ value.server_lvname }}
    - vgname: {{ value.server_vgname }}
    - size: {{ value.server_lvsize }}
{% endif %}
```
Lessons Learned

Disk state

Format and mount (incl. fstab entry) if not exists

```bash
# Create block device
{% if salt['disk.fstype']('value.server_lxpath') != 'xfs' %}
Configure blockdev for {{ value.server_lxpath }}:
  blockdev.formatted:
    - name: {{ value.server_lxpath }}
    - fs_type: {{ value.server_fstype }}
{% endif %}

# Check fstab for mounted devices of not found mount
{% if not salt['file.search']('/etc/fstab',value.server_lxpath) %}
Mount {{ value.server_lxpath }}:
  mount.mounted:
    - name: {{ value.server_mountp }}
    - device: {{ value.server_lxpath }}
    - fstype: xfs
    - mkmnt: true
    - persist: true
    - user: root
    - opts:
      - defaults
{% endif %}
```
Lessons Learned– self healing and Maintenance
Lessons Learned

Politics…

Selfhealing vs traditional detect and alert
Lessons Learned

Politics...

Is this a good idea?

```
/etc/ssh/sshd_config:
  file.managed:
    - user: root
    - group: root
    - mode: 0640
    - source: salt://02_COOP/AO_COOP_TEMPLATES/02_X86_64/LDAP_AUTH/CPF_sshd_config.template
      - template: jinja

/etc/ldap.conf:
  file.managed:
    - user: root
    - group: root
    - mode: 0644
    - source: salt://02_COOP/AO_COOP_TEMPLATES/02_X86_64/LDAP_AUTH/CPF_ldap.conf.template
      - template: jinja

sshd:
  pkg.installed:
    - name: openssh
  service.running:
    - enable: True
    - watch:
      - file: /etc/ssh/sshd_config
      - file: /etc/ldap.conf
      - require: pkg: openssh
```
Lessons Learned

Politics...

Or this?

```plaintext
{% if not salt['file.file_exists'](CorePath + value.name + "/no_start") %}
Hold {{ value.name }} Running:
  service.running:
    - name: {{ value.name }}.service
    - enable: True
{% endif %}
```
Lessons Learned

Politics...

What about highstates and changes during maintenance windows
Lessons Learned

Maintenance check...

```python
def patrol chk maint:
    Check if the minion is in Maintenance Mode.
    Returns True if the System is in Maintenance
    and False if not.

    salt '*' patrol.chk_maint

def patrol get maint info:
    Get information about the Maintenance in progress.
    returns the information string if Maintenance is set
    and 'No Maintenance set' if not.

    salt '*' patrol.get_maint_info
```

```bash
{% if not salt['patrol.chk_maint']('').%}
# Include CM states here
- .CM_filesystem_100
- .CM_system_cert_101
- .CM_dns_100
- .CM_ldap_auth_100
- .CM_local_auth_102
- .CM_system_files_100
- .CM_system_sysctl_100
- .CM_system_profile_103
- .CM_system_udev_100
- .CM_system_nfs_100
- .CM_system_additional_rpms_100
- .CM_system_coop_rpms_100
- .CM patrol_100
- .CM_system_mnom_100
- .CM system cfg2html_100
- .CM_system_mastercheck_100
- .CM_salt hs 101
- .CM_salt minion 100
# Include SW CM states here
{% else %}
- .report maintenance elastic_100
{% endif %}

{% set MAINTINFO = salt['patrol.get_maint info']('').%}
Report Maintenance to Elastic:
  cmd.run:
    - name: echo {{ MAINTINFO }}
```
Lessons Learned– Logging
Lessons Learned

Logging...

Salt "react-returner" aka Frankenturner
Lessons Learned

Logging...

```
Table | JSON
--- | ---
0 | 
1 | 
320 | 

```

```
@timestamp | version | _id | _index | _score | _type | a_stateid_change | a_stateid_error | case | change_count | data
--- | --- | --- | --- | --- | --- | --- | --- | --- | --- | ---
February 18th 2019, 16:11:19.423 | | 1 | | | | | | | | 
2019-02-18-02-42-33 | | | | | | | | | | 
```

```
cmd /usr/sbin/update-ca-certificates -r: run: 
----------
run_num: 10
state: 102
changes: 
comment: State was not run because none of the onchanges reqs changed 
result: True
```

```
cmd /usr/sbin/install-dirs for PatrolAgent -r: run: 
----------
run_num: 88
```

Q and A..?

What is going to happen…

I MUSTACHE YOU

A QUESTION
Unpublished Work of SUSE LLC. All Rights Reserved.

This work is an unpublished work and contains confidential, proprietary and trade secret information of SUSE LLC. Access to this work is restricted to SUSE employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of SUSE. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

General Disclaimer

This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. SUSE makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for SUSE products remains at the sole discretion of SUSE. Further, SUSE reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All SUSE marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.