Implementing SALT modules

Automate things where no-one has gone before

OpenSUSE Conference 2023, Nuremberg, 2023-05-27

Jochen Kellner <jochen@jochen.org>

Agenda

- Motivation
- Execution Modules
- State Modules
- Function signatures ~ Module parameters
- Examples
- Gain or pain?

Motivation for own modules (1)

- Configuration pain points
  - HANA parameters in ini-files
  - HDB userstore: credential store
- Goal: deploy and configure HANA databases
- SAP HANA: In-memory database – no community modules

Motivation for own modules (2)

- HANA parameters in ini-files
- Oh, there’s a SALT module for that!
- Works with file contents – needs HANA restart
- Other options:
  - HDB SQL: ALTER statements (needs DB credentials)
  - SAP script SetParameter.py as database OS user
- Both options can change (most) parameters dynamically

Motivation for own modules (3)

HANA Userstore: save credentials for login

Usage: hdbuserstore [options] command [arguments]

Set <KEY> <ENV>[@<DATABASE>] <USER> <PASSWORD>
Delete <KEY>
    Delete entry with key <KEY>.
List [<KEY> …]
    List entries of store. The passwords are not shown.

Execution Modules

- Get current configuration / Set desired configuration
- As simple as possible
- Use OS/DB commands/scripts/python modules as needed
- Use verbs as function names
- https://docs.saltproject.io/en/latest/ref/modules/index.html
- https://intothesaltmine.readthedocs.io/en/latest/chapters/chapters/development/writing-modules.html

Testing Execution Modules

- Stored in <file_root>/_modules
- Document with pydoc
- Call saltutils.sync_modules to sync to minions
- salt 'hostname*' module.function [parameters…]

Execution Module for HDB userstore

- Stored in <file_root>/_modules/hana.py
- Functions:
  - def hdbuserstore_set(sid, key, env, username, passwd, os_user=None):
  - def hdbuserstore_get(sid, key, os_user=None):
  - def hdbuserstore_delete(sid, key, os_user=None):
- There is no „hdbuserstore_list“ – the nodule works with keys

Execution Module call

```
hdbuserstore list
```

```
KEY BACKUPDB
  ENV: hostname:3nn13
  USER: BACKUP_ADMIN
  DATABASE: DB
```

```
salt 'hostname*' \
  hana.hdbuserstore_get SID \
  BACKUPDB
```

```
    ENV:
      hostname:3nn13@DB
    KEY:
      BACKUPDB
    USER:
      BACKUP_ADMIN
```

State Modules

- Desired State – calls execution modules to get work done
- Stored in <file_root>/_states
- Documentation in Pydoc
- Use adjectives to describe the desired state (present, absent...)
- Simple(?) python code and data structures
- Test-mode, diff
- https://docs.saltproject.io/en/latest/ref/states/writing.html

Function signatures ~ Module parameters

File: _state/hdbuserstore.py

```
def absent(sid, name,
           os_user=None):
```

```
my_hdbuserstore_absent:
  hdbuserstore.absent:
    sid: HDB
    name: MONITORING
    os_user: root
```

- Filename ↔ State Module
- Functionname ↔ Module function
- Function parameters ↔ State parameters (required, optional)

State Modules

Program flow:

1) Verify Parameters
2) Get current state, compare to desired state
3) Test-Mode: Return collected changes
4) Apply changes
5) Check the applied changes (again, like in 2)
6) Return collected changes

State Modules – return data

A state module returns the following information when applied:

```
ret["name"] = name
ret["result"] = True/False (success?)
ret["comment"] = "Descriptive text"

ret[name]["old"] = before_change
ret[name]["new"] = after_change
```

The dict is pretty printed when applying the state

Calling State Modules

```
salt 'hostname*' state.apply \
  hdbuserstore-test
```

```
ensure_userstore_absent:
  hdbuserstore.absent:
  - sid: SID
  - name: TEST
```

```
ID: ensure_userstore_absent
Function: hdbuserstore.absent
Name: TEST
Result: True
Comment: Delete userstore key TEST.
Changes:
  TEST:
    old:
     {'KEY': 'TEST', 'ENV': '...', 'USER': '...'}
```

Ensure HDB userstore

State to ensure a HDB userstore exists:

```
TSM:
  hdbuserstore.present:
    - sid: SID
    - env: hostname:30013
    - username: BACKUP_OPERATOR
    - password: strictly_confidential
    - verify: False
```

Ensure HDB userstore (2)

Used for inital deployment – first create user, then userstore

Usage for second day operations:

- Password change in HANA DB and userstore
- Move to FQDN (hostname → hostname.domain) and from hostname:port to hostname:3nn13@DB
  migrated more than 100*15*2 = 3,000 Userstore entries
  found 5 pre-existing errors (locked users)
  no breakage inflicted

Gain or pain?

- When there is no module, SALT can call commands
- But that’s not „desired state“ nor „idempotent“
- Need to think about „atomic/simple“ execution modules
- Need to define „desired state“ and how to describe that
- Gain: idempotency, test mode, validation, and changes displayed
- Pain: Code to write, test, and maintain (1k lines)
- Easy enough that it works quite well – we are happy with it

Questions and answers

Find the code
http://www.jochen.org/download/salt-hana-modules.tar.gz

Questions?