ansible-role-config-light Documentation
Release 1.4.0

Release 1.4.0

Vladimir Botka

Apr 11, 2020

TABLE OF CONTENTS:

USER'S GUIDE

Table of Contents

1.1 Introduction

- Ansible role: config_light

- Supported systems: FreeBSD, Ubuntu

- Requirements: None

The role installs packages, creates and configures files, services, and handlers. This provides a simple, but flexible framework to apply basic Ansible modules. Substantial part of the control-flow will be determined by the structure of the data. Some attributes of the dictionaries trigger Ansible modules to modify configuration files, configure services and create handlers.

The role can be used with any supported OS to install and configure arbitrary applications. The role is tested with supported releases of FreeBSD and Ubuntu. It can be expected that other BSD and Linux distributions, that support

the Ansible modules mentioned below, should work with minimal changes. Red Hat and Debian *ansible_os_family* should work out of the box.

Used Ansible modules comprise `package` to install Linux packages, and both `pkgng` and `portinstall` to install FreeBSD packages or ports.

Ansible modules `template`, `lineinfile`, `blockinfile` and `ini_file` are used to configure files. Module `service` is used to manage both Linux and FreeBSD services.

The directory `contrib` comprises examples of how to install and configure various applications, and how to create the handlers and templates.

The user of this role is expected to master at least the following Ansible topics:

* Basic Concepts
* Roles
* Working With Playbooks

Feel free to share your feedback and report issues. The contributions to the project are welcome.

1.2 Installation

The most convenient way how to install an Ansible role is to use Ansible Galaxy CLI `ansible-galaxy`. The utility comes with the standard Ansible package and provides the user with a simple interface to the Ansible Galaxy's services. For example take a look at the current status of the role

```
shell> ansible-galaxy info vbotka.config_light
```

and install it

```
shell> ansible-galaxy install vbotka.config_light
```

See also:

* To install specific versions from various sources see Installing content.
* Take a look at other roles `shell> ansible-galaxy search --author=vbotka`

1.3 Playbook

Below is simple playbook that applies this role at single host srv.example.com (2)

```
1  shell> cat config-light.yml
2  - hosts: srv.example.com
3    gather_facts: true
4    connection: ssh
5    remote_user: admin
6    become: yes
7    become_user: root
8    become_method: sudo
9    roles:
10     - vbotka.config_light
```

Note: `gather_facts:    true` (3) must be set to gather facts needed to evaluate OS specific options of the role. For example to install packages the variable *ansible_os_family* is needed to select the appropriate Ansible module.

See also:

- For details see Connection Plugins (4-5)

- and Understanding Privilege Escalation (6-8).

1.4 Debug

To see additional debug information enable debug output in the configuration

```
cl_debug: true
```

, or set the extra variable in the command:

```
shell> ansible-playbook config-light.yml -e 'cl_debug=true'
```

Note: The debug output of this role is optimized for the `yaml` callback plugin. Set this plugin for example in the environment `shell> export ANSIBLE_STDOUT_CALLBACK=yaml`.

See also:

- Playbook Debugger

1.5 Tags

The tags provide very useful tool to run selected tasks of the role. To see what tags are available list the tags of the role with the command:

```
1  shell> ansible-playbook config-light.yml --list-tags
2
3  playbook: config-light.yml
4
5    play #1 (srv.example.com): srv.example.com        TAGS: []
6    TASK TAGS: [always, cl_debug, cl_files, cl_packages, cl_sanity, cl_services, cl_
   ↪setup, cl_states, cl_vars]
```

For example display the list of the variables and their values with the tag `cl_debug` (when the debug is enabled `cl_debug:    true`). With this tag specified `-t cl_debug` all imported tasks before the task *debug.yml* will also run because of the tag `always` (when sanity testing is enabled `cl_sanity:    true` and setup is enabled `cl_setup:    true`). This is the default. See *main.yml*.

```
shell> ansible-playbook config-light.yml -t cl_debug
```

See what packages will be installed

```
shell> ansible-playbook config-light.yml -t cl_packages --check
```

Install packages and exit the play

```
shell> ansible-playbook config-light.yml -t cl_packages
```

1.6 Variables

In this chapter we describe role's default variables stored in the directory `defaults`.

See also:

- Ansible variable precedence: Where should I put a variable?

1.6.1 Default variables

Most of the variables are self-explaining. There are four very important variables `cl_handlers`, `cl_packages`, `cl_states`, `cl_services`, and `cl_files` (11-15). These dictionaries which comprise the configuration data of handlers, packages, services, and files will be explained in details. By default these dictionaries are empty.

Best practice is to provide the data either in *host_vars* and *group_vars* or as a files in the directories `cl_handlersd_dir`, `cl_packagesd_dir`, `cl_statesd_dir`, `cl_servicesd_dir`, and `cl_filesd_dir` (22-26). Both methods can be applied at the same time. The variables will be assembled and combined by the tasks `vars_handlers.yml`, `vars_packages.yml`, `vars_states.yml`, `vars_services.yml`, and `vars_files.yml`. The assembled dictionaries customized for each host in the play will be stored in the host-specific files `cl_packagesd`, `cl_statesd`, `cl_servicesd`, and `cl_filesd` (35-38). The variable `cl_handlers` is not host-specific because the handlers will be create at the controller (localhost). Assembled dictionary `cl_handlers` will be stored in the file `cl_handlersd` (34). Take a look at the directory `cl_dira` (33) to see assembled data.

By default the base of the directories is `role_path` (21). The user is expected to put the configuration data to more suitable directory, e.g., to `playbook_dir` directory.

[defaults/main.yml]

```
1  ---
2  # defaults for config_light
3
4  cl_sanity: true                # Import tasks/sanity.yml
5  cl_setup: true                 # Import tasks/setup.yml
6  cl_install: true               # Install packages or ports
7  cl_debug: false                # Print debug output
8  cl_backup: false               # Backup files
9
10 # Combine assembled data with these variables
11 cl_handlers: {}
12 cl_packages: {}
13 cl_services: {}
14 cl_files: {}
15 cl_states: {}
16
17 # Assemble data from these directories
18 # cl_dird_owner: root          # no default
19 # cl_dird_group: adm           # no default
20 cl_dird_dmode: "0775"          # default very permissive, restrict if necessary
21 cl_dird: "{{ role_path }}/files"
22 cl_handlersd_dir: "{{ cl_dird }}/handlers.d"
```

(continues on next page)

```
23  cl_packagesd_dir: "{{ cl_dird }}/packages.d"
24  cl_servicesd_dir: "{{ cl_dird }}/services.d"
25  cl_filesd_dir: "{{ cl_dird }}/files.d"
26  cl_statesd_dir: "{{ cl_dird }}/states.d"
27
28  # Assemble inventory_hostname data into these files
29  # cl_dira_owner: root        # no default
30  # cl_dira_group: adm         # no default
31  cl_dira_dmode: "0775"        # default very permissive, restrict if necessary
32  cl_dira_fmode: "0664"        # default very permissive, restrict if necessary
33  cl_dira: "{{ cl_dird }}/assemble"
34  cl_handlersd: "{{ cl_dira }}/handlersd"  # localhost; not inventory_hostname specific
35  cl_packagesd: "{{ cl_dira }}/packagesd.{{ inventory_hostname }}"
36  cl_servicesd: "{{ cl_dira }}/servicesd.{{ inventory_hostname }}"
37  cl_filesd: "{{ cl_dira }}/filesd.{{ inventory_hostname }}"
38  cl_statesd: "{{ cl_dira }}/statesd.{{ inventory_hostname }}"
39  cl_assemble_regexp: '^(.*)[^~]$'   # Any string but terminated with ~
40  # cl_assemble_validate: 'ansible-lint -x 205 %s'  # no default
41
42  # Handlers
43  cl_handlers_validate: 'ansible-lint %s'
44
45  # OS common
46  install_retries: 10
47  install_delay: 5
48
49  # FreeBSD
50  freebsd_install_method: packages
51  # freebsd_install_method: ports
52  freebsd_use_packages: true
53
54  # EOF
55  ...
```

> **Warning:** Defaults of the variables *cl_dira_dmode* (31) and *cl_dira_fmode* (32) are very permissive. These are
> the permissions to access the assembled dictionaries. Restrict the permissions if these dictionaries might comprise
> classified data.

<TODO: complete description of all default variables>

1.6.2 cl_handlers - dictionary with handlers

- *Synopsis*
- *Parameters*
- *Example*
- *See Also*
- *Notes*

Synopsis

The variable *cl_handlers* is a dictionary of the handlers. Structure of the dictionary depends on the template that is used to create the file with the handlers. For example, the structure below can be used with the template *handlers-auto1.yml.j2*.

Parameters

Parameter	Type	Comments
template	*string* `required`	Template filename
handler	*string* `required`	Name of the handler
module	*string* `required`	Ansible module in handler
params	*list* `required`	Ansible module parameters
conditions	*list*	List of conditions

Example

FreeBSD handlers for postfix

[contrib/postfix/conf-light/handlers.d/postfix-freebsd]

```
1  postfix_freebsd:
2    template: handlers-auto1.yml.j2
3    params:
4
5      - handler: 'enable and start postfix'
6        module: service
7        params:
8          - 'name: postfix'
9          - 'state: started'
10         - 'enabled: true'
```

(continues on next page)

```
11
12      - handler: 'disable and stop postfix'
13        module: service
14        params:
15          - 'name: postfix'
16          - 'state: stopped'
17          - 'enabled: false'
18
19      - handler: 'reload postfix'
20        module: service
21        params:
22          - 'name: postfix'
23          - 'state: reloaded'
24        conditions:
25          - '- cl_service_postfix_enable|bool'
26
27      - handler: 'restart postfix'
28        module: service
29        params:
30          - 'name: postfix'
31          - 'state: restarted'
32        conditions:
33          - '- cl_service_postfix_enable|bool'
34
35      - handler: 'postfix check'
36        module: command
37        params:
38          - 'cmd: /usr/local/sbin/postfix check'
39
40      - handler: 'newaliases'
41        module: command
42        params:
43          - 'cmd: /usr/bin/newaliases'
44
45 #    - handler: 'postmap smtp sasl passwords'
46 #      module: command
47 #      params:
48 #        - 'cmd: /usr/local/sbin/postmap {{ postfix_main_cf_smtp_sasl_password_maps }}
   ↪'
49
50 #    - handler: 'postmap virtual aliases'
51 #      module: command
52 #      params:
53 #        - cmd: /usr/local/sbin/postmap {{ postfix_virtual }}'
```

See Also

See also:

- See vars-handlers.yml how the variable *cl_handlers* is combined with the content of the directory *cl_handlersd_dir*.

- See setup.yml how the handlers are created.

- For details see the template handlers-auto1.yml.j2.

Notes

Note: The template *handlers-auto1.yml.j2* is available in the role's directory `templates`. The user is expected to create new templates when needed. Feel free to change the structure of the data and to create new templates that might fit the purpose better. Feel free to contribute new templates and configuration examples to the project.

1.6.3 cl_packages - dictionary with packages or BSD ports

- *Synopsis*
- *Parameters*
- *Example*
- *See Also*

Synopsis

The variable *cl_packages* is a dictionary of the packages, or BSD ports to be installed.

Parameters

Parameter	Type	Comments
name	*string* `required`	Package or BSD port

Example

FreeBSD package for postfix

[contrib/postfix/conf-light/packages.d/postfix]

```
1  postfix:
2    name: 'mail/postfix'
```

See Also

See also:

- See vars-packages.yml how the variable *cl_packages* is combined with the content of the directory *cl_packagesd_dir*.

- See packages.yml how the packages or BSD ports are installed.

1.6.4 cl_states - dictionary of files' states

- *Synopsis*
- *Parameters*
- *Example*
- *See Also*

Synopsis

The variable *cl_states* is a dictionary of the files' states.

Parameters

Parameter	Type	Comments
state	*string* `required`	State of the filename
path	*string* `required`	Path to file
owner	*string*	Owner of the file
group	*string*	Group of the file
mode	*string*	Mode of the file

<TODO: complete parameters. See tasks/states.yml>

Example

File's states

[contrib/lighttpd/conf-light/states.d/lighttpd-server-document-root]

```yaml
1  lighttpd_server_document_root:
2    state: directory
3    path: '{{ cl_lighttpd_server_document_root }}'
4    owner: '{{ cl_lighttpd_server_username }}'
5    group: '{{ cl_lighttpd_server_groupname }}'
6    mode: '0640'
```

See Also

See also:

- See vars-states.yml how the variable *cl_states* is combined with the content of the directory *cl_statesd_dir*.
- See states.yml how the file's states are set.

1.6.5 cl_services - dictionary with services

- *Synopsis*
- *Parameters*
- *Example*
- *See Also*

Synopsis

The variable *cl_services* is a dictionary with the services managed by this role.

Parameters

Parameter	*Type*	*Comments*
name	*string* `required`	Service
state	*string*	State of the service default: started
enabled	*boolean*	Start of boot default: true

Example

FreeBSD services for postfix

[contrib/postfix/conf-light/service.d/postfix]

```
1  postfix:
2    name: 'postfix'
3    state: '{{ cl_service_postfix_state }}'
4    enabled: '{{ cl_service_postfix_enable }}'
```

[contrib/postfix/conf-light/service.d/sendmail]

```
1  sendmail:
2    name: 'sendmail'
3    state: '{{ cl_service_sendmail_state }}'
4    enabled: '{{ cl_service_sendmail_enable }}'
```

See Also

See also:

- See vars-services.yml how the variable *cl_services* is combined with the content of the directory *cl_servicesd_dir*.
- See services.yml how the services are configured.

1.6.6 cl_files - dictionary with files

- *Synopsis*
- *See Also*
- *Parameters for template*
- *Example of template*
- *See Also*
- *Notes*
- *Parameters for lineinfile*
- *Example of lineinfile*
- *See Also*
- *Parameters for blockinfile*
- *Example of blockinfileinfile*
- *See Also*
- *Parameters for ini_file*
- *Example of ini_file*
- *See Also*

Synopsis

The variable *cl_files* is a dictionary of the files that shall be created or modified by this role. It's optional which Ansible module will be used to create or modify a file and more options can be applied to create and modify the same file. For example, it's possible to create a file by the Ansible module *template* and modify it with the module *lineinfile* later. Several options are available:

1. template: If the attribute *template* is defined in the dictionary
2. lineinfile: If the attribute *lines* is defined in the dictionary
3. blockinfile: If the attribute *blocks* is defined in the dictionary
4. ini_file: If the attribute *ini* is defined in the dictionary

Multiple options, when used to create or modify a file, will be applied in this order.

See Also

See also:

- See vars-files.yml how the variable *cl_files* is combined with the content of the directory *cl_filesd_dir*.
- See files.yml how the files are created and modified.
- See files-create-backup.yml how the backups are created (when enabled by *cl_backup*).
- See files-delete-backup.yml how the backup files are deleted when the files haven't been modified.

Parameters for template

Parameter	Type	Comments
path	*string* `required`	Path to file
template	*string* `required`	Template filename
owner	*string*	Owner of the file
group	*string*	Group of the file
mode	*string*	Mode of the file
force	*boolean*	Replace when different default: true
validate	*string*	Command to validate file
handlers	*list*	List of handlers

Example of template

File `/etc/mail/mailer.conf` for postfix

[contrib/postfix/conf-light/files.d/mailer-conf]

```yaml
1  mailerconf:
2    path: '/etc/mail/mailer.conf'
3    force: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    template: 'mailer.conf.j2'
```

See Also

See also:

- See files-template.yml how the files are modified or created by the Ansible module `template`.

Notes

Note: There are couple of templates ready to be used in the directory `templates`. The user is expected to create new templates when needed. Feel free to contribute new templates to the project.

Parameters for lineinfile

Parameter	Type	Comments
path	*string* `required`	Path to file
lines	*list* `required`	List of regexp and lines
owner	*string*	Owner of the file
group	*string*	Group of the file
mode	*string*	Mode of the file
create	*boolean*	Create if does not exist default: false
validate	*string*	Command to validate file
handlers	*list*	List of handlers

Example of lineinfile

File `/usr/local/etc/postfix/main.cf` for postfix

[contrib/postfix/conf-light/files.d/postfix-main.cf]

```
1  postfix_main_cf:
2    path: '/usr/local/etc/postfix/main.cf'
3    create: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    handlers:
8      - 'postfix_freebsd reload postfix'
9    lines:
10     - regexp: '^myhostname\s*=\s*(.*)$'
11       line: 'myhostname = {{ cl_myhostname }}'
```

See Also

See also:

- See files-lineinfile.yml how the files are modified or created by the Ansible module `lineinfile`.

Parameters for blockinfile

Parameter	Type	Comments
path	*string* `required`	Path to file
blocks	*list* `required`	List of blocks and markers
owner	*string*	Owner of the file
group	*string*	Group of the file
mode	*string*	Mode of the file
create	*boolean*	Create if does not exist default: false
validate	*string*	Command to validate file
handlers	*list*	List of handlers

Example of blockinfileinfile

<TODO: No example yet>

See Also

See also:

- See files-blockinfile.yml how the files are modified or created by the Ansible module `blockinfile`.

Parameters for ini_file

Parameter	Type	Comments
path	*string* `required`	Path to file
ini	*list* `required`	List of {section,option, value} dictionaries
owner	*string*	Owner of the file
group	*string*	Group of the file
mode	*string*	Mode of the file
create	*boolean*	Create if does not exist default: true
handlers	*list*	List of handlers

Example of ini_file

<TODO: No example yet>

See Also

See also:

- See files-inifile.yml how the files are modified or created by the Ansible module `ini_file`.

1.7 Best practice

Create variables. Take a look at directory `conf-light/assemble/` what files were created.

```
shell> ansible-playbook config-light.yml -t cl_vars
```

Test sanity. Then disable this task `cl_sanity:    false` to speedup the playbook.:

```
shell> ansible-playbook config-light.yml -t cl_sanity
```

Create handlers. Take a look at directory `roles/vbotka.config_light/handlers` what handlers were created. Run this task once to create the handlers. Then disable this task `cl_setup:    false` to speedup the playbook.

```
shell> ansible-playbook config-light.yml -t cl_setup
```

Display variables. Display the variables for debug if needed. Then disable this task `cl_debug:    false` to speedup the playbook.

```
shell> ansible-playbook config-light.yml -t cl_debug
```

Install packages. Then disable this task `cl_install:    false` to speedup the playbook.

```
shell> ansible-playbook config-light.yml -t cl_packages
```

Set files' states.

```
shell> ansible-playbook config-light.yml -t cl_states
```

Create and modify files

```
shell> ansible-playbook config-light.yml -t cl_files
```

Configure services

```
shell> ansible-playbook config-light.yml -t cl_services
```

The role and the configuration data in the examples are idempotent. Once the application is installed and configured there should be no changes reported by *ansible-playbook* when running the playbook repeatedly. Disable setup, sanity, debug, and install to speedup the playbook

```
shell> ansible-playbook config-light.yml
```

ANNOTATED SOURCE CODE

2.1 Tasks

2.1.1 main.yml

Synopsis: Tasks of the playbook.

Description of the task.

[main.yml]

```yaml
---
# tasks for config_light

- import_tasks: setup.yml
  when: cl_setup|bool
  delegate_to: localhost
  run_once: true
  tags: [cl_setup, always]

- import_tasks: vars.yml
  tags: [cl_vars, always]

- import_tasks: sanity.yml
  when: cl_sanity|bool
  tags: [cl_sanity, always]

- import_tasks: debug.yml
  when: cl_debug|bool
  tags: [cl_debug, always]

- import_tasks: packages.yml
  when: cl_install|bool
  tags: cl_packages

- import_tasks: states.yml
  tags: cl_states

- import_tasks: files.yml
  tags: cl_files

- import_tasks: services.yml
  tags: cl_services

# EOF
...
```

2.1.2 vars.yml

Synopsis: Combine dictionaries of packages, services and files.

Description of the task.

[vars.yml]

```yaml
---

```

```yaml
3    - name: "vars: Debug"
4      vars:
5        msg: |
6          cl_handlersd_dir [{{ cl_handlersd_dir }}]
7          cl_packagesd_dir [{{ cl_packagesd_dir }}]
8          cl_statesd_dir [{{ cl_statesd_dir }}]
9          cl_servicesd_dir [{{ cl_servicesd_dir }}]
10         cl_filesd_dir [{{ cl_filesd_dir }}]
11         cl_dira [{{ cl_dira }}]
12         cl_handlersd [{{ cl_handlersd }}]
13         cl_packagesd [{{ cl_packagesd }}]
14         cl_statesd [{{ cl_statesd }}]
15         cl_servicesd [{{ cl_servicesd }}]
16         cl_filesd [{{ cl_filesd }}]
17     debug:
18       msg: "{{ msg.split('\n') }}"
19
20   - name: "vars: Packages"
21     import_tasks: vars-packages.yml
22
23   - name: "vars: States"
24     import_tasks: vars-states.yml
25
26   - name: "vars: Services"
27     import_tasks: vars-services.yml
28
29   - name: "vars: Files"
30     import_tasks: vars-files.yml
31
32   # EOF
33   ...
```

2.1.3 vars-packages.yml

Synopsis: Combine dictionaries of packages.

Description of the task.

[vars-packages.yml]

```yaml
1    ---
2
3    - name: "vars-packages: Assemble packages to {{ cl_packagesd }}"
4      assemble:
5        regexp: "{{ cl_assemble_regexp }}"
6        src: "{{ cl_packagesd_dir }}"
7        dest: "{{ cl_packagesd }}"
8        owner: "{{ cl_dira_owner|default(omit) }}"
9        group: "{{ cl_dira_group|default(omit) }}"
10       mode: "{{ cl_dira_fmode }}"
11       validate: "{{ cl_assemble_validate|default(omit) }}"
12     delegate_to: localhost
13
14   - name: "vars-packages: Include files from {{ cl_packagesd }} to cl_packagesd_items"
15     include_vars:
16       file: "{{ cl_packagesd }}"
```

```
17      name: cl_packagesd_items
18
19  - name: "vars-packages: Combine cl_packages with cl_packagesd_items"
20    set_fact:
21      cl_packages: "{{ cl_packages|combine(cl_packagesd_items|default({})) }}"
22
23  - name: "vars-packages: Debug"
24    debug:
25      var: cl_packages
26    when: cl_debug|bool
27
28  # EOF
29  ...
```

2.1.4 vars-states.yml

Synopsis: Combine dictionaries of files' states.

Description of the task.

[vars-states.yml]

```
1   ---
2
3   - name: "vars-states: Assemble states to {{ cl_states }}"
4     assemble:
5       regexp: "{{ cl_assemble_regexp }}"
6       src: "{{ cl_statesd_dir }}"
7       dest: "{{ cl_statesd }}"
8       owner: "{{ cl_dira_owner|default(omit) }}"
9       group: "{{ cl_dira_group|default(omit) }}"
10      mode: "{{ cl_dira_fmode }}"
11      validate: "{{ cl_assemble_validate|default(omit) }}"
12    delegate_to: localhost
13
14  - name: "vars-states: Include files from {{ cl_statesd }} to cl_statessd_items"
15    include_vars:
16      file: "{{ cl_statesd }}"
17      name: cl_statesd_items
18
19  - name: "vars-states: Combine cl_statess with cl_statesd_items"
20    set_fact:
21      cl_states: "{{ cl_states|combine(cl_statesd_items|default({})) }}"
22
23  - name: "vars-states: Debug"
24    debug:
25      var: cl_states
26    when: cl_debug|bool
27
28  # EOF
29  ...
```

2.1.5 vars-services.yml

Synopsis: Combine dictionaries of services.

Description of the task.

[vars-services.yml]

```yaml
---

- name: "vars-services: Assemble services to {{ cl_servicesd }}"
  assemble:
    regexp: "{{ cl_assemble_regexp }}"
    src: "{{ cl_servicesd_dir }}"
    dest: "{{ cl_servicesd }}"
    owner: "{{ cl_dira_owner|default(omit) }}"
    group: "{{ cl_dira_group|default(omit) }}"
    mode: "{{ cl_dira_fmode }}"
    validate: "{{ cl_assemble_validate|default(omit) }}"
  delegate_to: localhost

- name: "vars-services: Include files from {{ cl_servicesd }} to cl_cervicesd_items"
  include_vars:
    file: "{{ cl_servicesd }}"
    name: cl_servicesd_items

- name: "vars-services: Combine cl_services with cl_servicesd_items"
  set_fact:
    cl_services: "{{ cl_services|combine(cl_servicesd_items|default({})) }}"

- name: "vars-services: Debug"
  debug:
    var: cl_services
  when: cl_debug|bool

# EOF
...
```

2.1.6 vars-files.yml

Synopsis: Combine dictionaries of files.

Description of the task.

[vars-files.yml]

```yaml
---

- name: "vars-files: Assemble files to {{ cl_filesd }}"
  assemble:
    regexp: "{{ cl_assemble_regexp }}"
    src: "{{ cl_filesd_dir }}"
    dest: "{{ cl_filesd }}"
    owner: "{{ cl_dira_owner|default(omit) }}"
    group: "{{ cl_dira_group|default(omit) }}"
    mode: "{{ cl_dira_fmode }}"
    validate: "{{ cl_assemble_validate|default(omit) }}"
  delegate_to: localhost
```

```yaml
13
14  - name: "vars-files: Include files from {{ cl_filesd }} to cl_filesd_items"
15    include_vars:
16      file: "{{ cl_filesd }}"
17      name: cl_filesd_items
18
19  - name: "vars-files: Combine cl_files with cl_filesd_items"
20    set_fact:
21      cl_files: "{{ cl_files|combine(cl_filesd_items|default({})) }}"
22
23  - name: "vars-files: Debug"
24    debug:
25      var: cl_files
26    when: cl_debug|bool
27
28  # EOF
29  ...
```

2.1.7 sanity.yml

Synopsis: Test sanity.

Description of the task.

[sanity.yml]

```yaml
1   ---
2
3   - name: "sanity: Directories to assemble data from must exist"
4     block:
5       - name: "sanity: Directories to assemble data from must exist"
6         debug:
7           msg: "{{ msg.split('\n')[:-1] }}"
8         vars:
9           msg: |
10            Directories to assemble data from do not exist. End of host.
11            Hint: Double check existence of the directories
12            {{ cl_handlersd_dir }}
13            {{ cl_packagesd_dir }}
14            {{ cl_statesd_dir }}
15            {{ cl_servicesd_dir }}
16            {{ cl_filesd_dir }}
17      - name: "sanity: End of host"
18        meta: end_host
19    when:
20      - cl_handlersd_dir is not exists or
21        cl_packagesd_dir is not exists or
22        cl_statesd_dir is not exists or
23        cl_servicesd_dir is not exists or
24        cl_filesd_dir is not exists
25
26  - name: "sanity: Check mode not possible without assembled data"
27    block:
28      - name: "sanity: Check mode not possible without assembled data"
29        debug:
30          msg: "{{ msg.split('\n')[:-1] }}"
```

```
31          vars:
32            msg: |
33              Check mode not possible without assembled data. End of host.
34              Hint: Assemble the variables first.
35              Run: ansible-playbook playbook.yml -t cl_vars
36        - name: "sanity: End of host"
37          meta: end_host
38      when:
39        - ansible_check_mode
40        - cl_packagesd is not exists or
41          cl_statesd is not exists or
42          cl_servicesd is not exists or
43          cl_filesd is not exists
44
45   # EOF
46   ...
```

2.1.8 setup.yml

Synopsis: Setup. Create dirs. Create handlers.

Description of the task.

[setup.yml]

```
1    ---
2
3    # directories
4    - name: "setup: Create directories in {{ cl_dird }}"
5      file:
6        state: directory
7        path: "{{ item }}"
8        owner: "{{ cl_dird_owner|default(omit) }}"
9        group: "{{ cl_dird_group|default(omit) }}"
10       mode: "{{ cl_dird_dmode }}"
11     loop:
12       - "{{ cl_dird }}"
13       - "{{ cl_handlersd_dir }}"
14       - "{{ cl_packagesd_dir }}"
15       - "{{ cl_servicesd_dir }}"
16       - "{{ cl_filesd_dir }}"
17       - "{{ cl_statesd_dir }}"
18
19   - name: "setup: Create directory {{ cl_dira }}"
20     file:
21       state: directory
22       path: "{{ cl_dira }}"
23       owner: "{{ cl_dira_owner|default(omit) }}"
24       group: "{{ cl_dira_group|default(omit) }}"
25       mode: "{{ cl_dira_dmode }}"
26
27   # handlers
28   - name: "setup: Assemble handlers"
29     import_tasks: vars-handlers.yml
30
31   - name: "setup: Create handlers"
```

```
32      template:
33        dest: "{{ role_path }}/handlers/handlers-auto-{{ item.key }}.yml"
34        src: "{{ item.value.template }}"
35        validate: "{{ cl_handlers_validate|default(omit) }}"
36        backup: "{{ cl_backup }}"
37      loop: "{{ cl_handlers|dict2items }}"
38      loop_control:
39        label: "{{ role_path }}/handlers/handlers-auto-{{ item.key }}.yml"
40
41    - name: "setup: Include handlers"
42      lineinfile:
43        path: "{{ role_path }}/handlers/main.yml"
44        line: "- import_tasks: handlers-auto-{{ item.key }}.yml"
45        validate: "{{ cl_handlers_validate|default(omit) }}"
46        backup: "{{ cl_backup }}"
47        create: true
48      loop: "{{ cl_handlers|dict2items }}"
49      loop_control:
50        label: "{{ role_path }}/handlers/handlers-auto-{{ item.key }}.yml"
51
52    # EOF
53    ...
```

2.1.9 vars-handlers.yml

Synopsis: Combine dictionaries of handlers.

Description of the task.

[vars-handlers.yml]

```
1    ---
2
3    - name: "vars-handlers: Assemble handlers to {{ cl_handlersd }}"
4      assemble:
5        regexp: "{{ cl_assemble_regexp }}"
6        src: "{{ cl_handlersd_dir }}"
7        dest: "{{ cl_handlersd }}"
8        owner: "{{ cl_dira_owner|default(omit) }}"
9        group: "{{ cl_dira_group|default(omit) }}"
10       mode: "{{ cl_dira_fmode }}"
11       validate: "{{ cl_assemble_validate|default(omit) }}"
12     delegate_to: localhost
13
14   - name: "vars-handlers: Include files from {{ cl_handlersd }} to cl_handlersd_items"
15     include_vars:
16       file: "{{ cl_handlersd }}"
17       name: cl_handlersd_items
18
19   - name: "vars-handlers: Combine cl_handlers with cl_handlersd_items"
20     set_fact:
21       cl_handlers: "{{ cl_handlers|combine(cl_handlersd_items|default({})) }}"
22
23   - name: "vars-handlers: Debug"
24     debug:
25       var: cl_handlers
```

```
26    when: cl_debug|bool
27
28  # EOF
29  ...
```

2.1.10 debug.yml

Synopsis: Display variables.

Description of the task.

[debug.yml]

```yaml
1   ---
2
3   - name: "debug: Config Light"
4     vars:
5       msg: |
6         ansible_os_family [{{ ansible_os_family }}]
7         ansible_distribution [{{ ansible_distribution }}]
8         ansible_distribution_major_version [{{ ansible_distribution_major_version }}]
9         ansible_distribution_version [{{ ansible_distribution_version }}]
10        ansible_distribution_release [{{ ansible_distribution_release }}]
11        ansible_python_version [{{ ansible_python_version }}]
12
13        cl_sanity [{{ cl_sanity }}]
14        cl_setup [{{ cl_setup }}]
15        cl_install [{{ cl_install }}]
16        cl_backup [{{ cl_backup }}]
17
18        cl_handlers
19        {{ cl_handlers|to_nice_yaml }}
20        cl_packages
21        {{ cl_packages|to_nice_yaml }}
22        cl_services
23        {{ cl_services|to_nice_yaml }}
24        cl_files
25        {{ cl_files|to_nice_yaml }}
26        cl_states
27        {{ cl_states|to_nice_yaml }}
28
29        cl_dird [{{ cl_dird }}]
30        cl_dird_owner [{{ cl_dira_owner|default('UNDEFINED') }}]
31        cl_dird_group [{{ cl_dira_group|default('UNDEFINED') }}]
32        cl_dird_dmode [{{ cl_dira_dmode }}]
33
34        cl_handlersd_dir [{{ cl_handlersd_dir }}]
35        cl_packagesd_dir [{{ cl_packagesd_dir }}]
36        cl_servicesd_dir [{{ cl_servicesd_dir }}]
37        cl_filesd_dir [{{ cl_filesd_dir }}]
38        cl_statesd_dir [{{ cl_statesd_dir }}]
39
40        cl_dira [{{ cl_dira }}]
41        cl_dira_owner [{{ cl_dira_owner|default('UNDEFINED') }}]
42        cl_dira_group [{{ cl_dira_group|default('UNDEFINED') }}]
43        cl_dira_dmode [{{ cl_dira_dmode }}]
```

```
44          cl_dira_fmode [{{ cl_dira_fmode }}]
45          cl_assemble_regexp [{{ cl_assemble_regexp }}]
46
47          cl_handlersd [{{ cl_handlersd }}]
48          cl_packagesd [{{ cl_packagesd }}]
49          cl_servicesd [{{ cl_servicesd }}]
50          cl_filesd [{{ cl_filesd }}]
51          cl_statesd [{{ cl_statesd }}]
52
53          cl_assemble_validate [{{ cl_assemble_validate|default('UNDEFINED') }}]
54          cl_handlers_validate [{{ cl_handlers_validate|default('UNDEFINED') }}]
55
56          install_retries [{{ install_retries }}]
57          install_delay [{{ install_delay }}]
58
59          freebsd_install_method [{{ freebsd_install_method }}]
60          freebsd_use_packages [{{ freebsd_use_packages }}]
61
62    debug:
63      msg: "{{ msg.split('\n') }}"
64
65  # EOF
66  ...
```

2.1.11 packages.yml

Synopsis: Install packages.

Description of the task.

[packages.yml]

```
1   ---
2
3   # FreeBSD ----------------------------------------------------------------
4   - name: "packages: Install packages FreeBSD"
5     block:
6       - name: "packages: Install packages FreeBSD"
7         pkgng:
8           name: "{{ item.value.name }}"
9           state: "{{ item.value.state|default(omit) }}"
10          annotation: "{{ item.value.annotation|default(omit) }}"
11          autoremove: "{{ freebsd_pkgng_autoremove|default(omit) }}"
12          cached: "{{ freebsd_pkgng_cached|default(omit) }}"
13          chroot: "{{ freebsd_pkgng_chroot|default(omit) }}"
14          jail: "{{ freebsd_pkgng_jail|default(omit) }}"
15          pkgsite: "{{ freebsd_pkgng_pkgsite|default(omit) }}"
16          rootdir: "{{ freebsd_pkgng_rootdir|default(omit) }}"
17        loop: "{{ cl_packages|dict2items }}"
18        loop_control:
19          label: "{{ item.key }}"
20        register: result
21        until: result is succeeded
22        retries: "{{ install_retries }}"
23        delay: "{{ install_delay }}"
24      - name: "packages: Debug FreeBSD packages"
```

```
25          debug:
26            var: result
27          when: cl_debug|bool
28      when:
29        - ansible_os_family == "FreeBSD"
30        - freebsd_install_method|lower == "packages"
31
32  - name: "packages: Install ports FreeBSD"
33    block:
34        - name: "packages: Install ports FreeBSD"
35          portinstall:
36            name: "{{ item.value.name }}"
37            state: "{{ item.value.state|default(omit) }}"
38            use_packages: "{{ item.value.use_packages|default(freebsd_use_packages) }}"
39          loop: "{{ cl_packages|dict2items }}"
40          loop_control:
41            label: "{{ item.key }}"
42          register: result
43          until: result is succeeded
44          retries: "{{ install_retries }}"
45          delay: "{{ install_delay }}"
46        - name: "packages: Debug FreeBSD ports"
47          debug:
48            var: result
49          when: cl_debug|bool
50      when:
51        - ansible_os_family == "FreeBSD"
52        - freebsd_install_method|lower == "ports"
53
54  # Linux ----------------------------------------------------------------
55  - name: "packages: Install packages Linux"
56    block:
57        - name: "packages: Install packages Linux"
58          package:
59            name: "{{ item.value.name }}"
60            state: "{{ item.value.state|default('present') }}"
61            use: "{{ item.value.use|default('auto') }}"
62          loop: "{{ cl_packages|dict2items }}"
63          loop_control:
64            label: "{{ item.key }}"
65          register: result
66          until: result is succeeded
67          retries: "{{ install_retries }}"
68          delay: "{{ install_delay }}"
69        - name: "packages: Debug Linux"
70          debug:
71            var: result
72          when: cl_debug|bool
73      when: ansible_os_family == "RedHat" or
74            ansible_os_family == "Debian"
75
76  # EOF
77  ...
```

2.1.12 states.yml

Synopsis: Configure states of files.

Description of the task.

[states.yml]

```yaml
1   ---
2
3   - name: "states: Apply states"
4     file:
5       state: "{{ item.value.state }}"
6       path: "{{ item.value.path }}"
7       src: "{{ item.value.src|default(omit) }}"
8       owner: "{{ item.value.owner|default(omit) }}"
9       group: "{{ item.value.group|default(omit) }}"
10      mode: "{{ item.value.mode|default(omit) }}"
11      attributes: "{{ item.value.attributes|default(omit) }}"
12      recurse: "{{ item.value.recurse|default(omit) }}"
13      force: "{{ item.value.force|default(omit) }}"
14      follow: "{{ item.value.follow|default(omit) }}"
15      access_time: "{{ item.value.access_time|default(omit) }}"
16      access_time_format: "{{ item.value.access_time_format|default(omit) }}"
17      modification_time: "{{ item.value.modification_time|default(omit) }}"
18      modification_time_format: "{{ item.value.modification_time_format|default(omit) }}
   →"
19      unsafe_writes: "{{ item.value.unsafe_writes|default(omit) }}"
20    loop: "{{ cl_states|dict2items }}"
21    loop_control:
22      label: "{{ item.value.path }}"
23
24  # EOF
25  ...
```

2.1.13 files.yml

Synopsis: Create or modify files.

Description of the task.

[files.yml]

```yaml
1   ---
2
3   - name: "files: Create backup"
4     import_tasks: files-create-backup.yml
5     when: cl_backup|bool
6
7   - name: "files: Template"
8     import_tasks: files-template.yml
9
10  - name: "file: Lineinfile"
11    import_tasks: files-lineinfile.yml
12
13  - name: "file: Blockinfile"
14    import_tasks: files-blockinfile.yml
15
```

```yaml
16    - name: "files: INI file"
17      import_tasks: files-inifile.yml
18
19    - name: "file: Delete backup"
20      import_tasks: files-delete-backup.yml
21      when:
22        - cl_backup|bool
23        - not ansible_check_mode
24
25    # EOF
26    ...
```

2.1.14 files-create-backup.yml

Synopsis: Create backup files.

Description of the task.

[files-create-backup.yml]

```yaml
1     ---
2
3     - name: "file-create-backup: Create time-stamp"
4       set_fact:
5         cl_timestamp: "{{ '%Y-%m-%d_%H_%M_%S'|strftime }}"
6
7     - name: "file-create-backup: Stat {{ item.path }}"
8       stat:
9         path: "{{ item.path }}"
10      loop: "{{ cl_files.values() }}"
11      loop_control:
12        label: "{{ item.path }}"
13      register: result
14
15    - name: "file-create-backup: Debug result"
16      debug:
17        var: result
18      when: cl_debug|bool
19
20    - name: "file-create-backup: Create backup files"
21      copy:
22        remote_src: true
23        src: "{{ item.item.path }}"
24        dest: "{{ item.item.path }}_{{ cl_timestamp }}.bak"
25        mode: "preserve"
26      loop: "{{ result.results }}"
27      loop_control:
28        label: "{{ item.item.path }}"
29      when: item.stat.exists
30      changed_when: false
31
32    # EOF
33    ...
```

2.1.15 files-delete-backup.yml

Synopsis: Delete backup files if the files haven't been modified.

Description of the task.

[files-delete-backup.yml]

```yaml
---

- name: "files-delete-backup: Delete backup files that did not change"
  when: cl_backup|bool
  file:
    state: absent
    path: "{{ item }}_{{ cl_timestamp }}.bak"
  loop: "{{ cl_files.values()|json_query('[].path')|
            difference(
            cl_results_template.results|default([])|
            json_query('[?changed==`true`].invocation.module_args.path'))|unique|
            difference(
            cl_results_lines.results|default([])|
            json_query('[?changed==`true`].invocation.module_args.path'))|unique|
            difference(
            cl_results_blocks.results|default([])|
            json_query('[?changed==`true`].invocation.module_args.path'))|unique|
            difference(
            cl_results_inifile.results|default([])|
            json_query('[?changed==`true`].invocation.module_args.path'))|unique
            }}"
  changed_when: false

# EOF
...
```

2.1.16 files-template.yml

Synopsis: Create or modify files by Ansible module `template`.

Description of the task.

[files-template.yml]

```yaml
---
# TODO: Complete parameters
- name: "files-template: Template"
  template:
    dest: "{{ item.path }}"
    src: "{{ item.template }}"
    owner: "{{ item.owner|default(omit) }}"
    group: "{{ item.group|default(omit) }}"
    mode: "{{ item.mode|default(omit) }}"
    force: "{{ item.force|default(omit) }}"
    validate: "{{ item.validate|default(omit) }}"
    # backup: "{{ cl_backup }}"
  loop: "{{ cl_files.values()|selectattr('template', 'defined')|list }}"
  loop_control:
    label: "{{ item.path }}"
  notify: "{{ item.handlers|default(omit) }}"
```

```yaml
17      register: cl_results_template
18
19  - name: "files-template: Debug"
20      block:
21          - name: Debug cl_results_template
22            debug:
23              var: cl_results_template
24          - name: Debug changed template path
25            debug:
26              msg: "{{ cl_results_template|default([])|
27                      json_query('[?changed==`true`].invocation.module_args.path')
28                      }}"
29      when: cl_debug|bool
30
31  # EOF
32  ...
```

2.1.17 files-lineinfile.yml

Synopsis: Create or modify files by Ansible module `lineinfile`.

Description of the task.

[files-lineinfile.yml]

```yaml
1   ---
2   # TODO: Complete parameters
3   - name: "files-lineinfile: Lineinfile"
4     lineinfile:
5       path: "{{ item.0.path }}"
6       regexp: "{{ item.1.regexp }}"
7       line: "{{ item.1.line }}"
8       owner: "{{ item.0.owner|default(omit) }}"
9       group: "{{ item.0.group|default(omit) }}"
10      mode: "{{ item.0.mode|default(omit) }}"
11      create: "{{ item.0.create|default(omit) }}"
12      validate: "{{ item.0.validate|default(omit) }}"
13      # backup: "{{ cl_backup }}"
14    loop: "{{ cl_files.values()|subelements('lines', skip_missing=true) }}"
15    loop_control:
16      label: "{{ item.0.path }}"
17    notify: "{{ item.0.handlers|default(omit) }}"
18    register: cl_results_lines
19
20  - name: "files-lineinfile: Debug"
21    block:
22        - name: Debug cl_results_lines
23          debug:
24            var: cl_results_lines
25        - name: Debug changed lines paths
26          debug:
27            msg: "{{ cl_results_lines.results|default([])|
28                    json_query('[?changed==`true`].invocation.module_args.path')
29                    }}"
30    when: cl_debug|bool
31
```

```
32   # EOF
33   ...
```

2.1.18 files-blockinfile.yml

Synopsis: Create or modify files by Ansible module `blockinfile`.

Description of the task.

[files-blockinfile.yml]

```
1    ---
2    # TODO: Complete parameters
3    - name: "files-blockinfile: Blockinfile"
4      blockinfile:
5        path: "{{ item.0.path }}"
6        marker: "# {mark} ANSIBLE MANAGED BLOCK {{ item.1.marker }}"
7        block: "{{ item.1.block }}"
8        owner: "{{ item.0.owner|default(omit) }}"
9        group: "{{ item.0.group|default(omit) }}"
10       mode: "{{ item.0.mode|default(omit) }}"
11       create: "{{ item.0.create|default(omit) }}"
12       validate: "{{ item.0.validate|default(omit) }}"
13       # backup: "{{ cl_backup }}"
14     loop: "{{ cl_files.values()|subelements('blocks', skip_missing=true) }}"
15     loop_control:
16       label: "{{ item.0.path }}"
17     notify: "{{ item.0.handlers|default(omit) }}"
18     register: cl_results_blocks
19
20   - name: "files-blockinfile: Debug"
21     block:
22       - name: Debug cl_results_blocks
23         debug:
24           var: cl_results_blocks
25       - name: Debug changed blocks paths
26         debug:
27           msg: "{{ cl_results_blocks.results|default([])|
28                    json_query('[?changed==`true`].invocation.module_args.path')
29                    }}"
30     when: cl_debug|bool
31
32   # EOF
33   ...
```

2.1.19 files-inifile.yml

Synopsis: Create or modify files by Ansible module `ini_infile`.

Description of the task.

[files-inifile.yml]

```yaml
 1  ---
 2  # TODO: Complete parameters
 3  - name: "files-inifile: INI files"
 4    ini_file:
 5      path: "{{ item.0.path }}"
 6      section: "{{ item.1.section }}"
 7      option: "{{ item.1.option }}"
 8      value: "{{ item.1.value }}"
 9      owner: "{{ item.0.owner|default(omit) }}"
10      group: "{{ item.0.group|default(omit) }}"
11      mode: "{{ item.0.mode|default(omit) }}"
12      create: "{{ item.0.create|default(omit) }}"
13      # backup: "{{ cl_backup }}"
14    loop: "{{ cl_files.values()|subelements('ini', skip_missing=true) }}"
15    loop_control:
16      label: "{{ item.0.path }}"
17    notify: "{{ item.0.handlers|default(omit) }}"
18    register: cl_results_ini
19
20  - name: "files-inifile: Debug"
21    block:
22      - name: Debug cl_results_ini
23        debug:
24          var: cl_results_ini
25      - name: Debug changed ini paths
26        debug:
27          msg: "{{ cl_results_ini.results|default([])|
28                   json_query('[?changed==`true`].invocation.module_args.path')
29                   }}"
30    when: cl_debug|bool
31
32  # EOF
33  ...
```

2.1.20 services.yml

Synopsis: Configure services.

Description of the task.

[services.yml]

```yaml
 1  ---
 2
 3  # FreeBSD --------------------------------------------------------------
 4  - name: "services: FreeBSD"
 5    block:
 6
 7      - name: "services: Enable service in rc.conf FreeBSD"
 8        lineinfile:
```

(continues on next page)

```yaml
 9          dest: "/etc/rc.conf"
10          regexp: "^\\s*{{ item.value.name }}_enable\\s*="
11          line: "{{ item.value.name }}_enable=\"YES\""
12          backup: "{{ cl_backup }}"
13        loop: "{{ cl_services|dict2items }}"
14        loop_control:
15          label: "{{ item.key }}"
16        when: "item.value.enabled|default(true)|bool"
17
18      - name: "services: Disable service in rc.conf FreeBSD"
19        lineinfile:
20          dest: "/etc/rc.conf"
21          regexp: "^\\s*{{ item.value.name }}_enable\\s*="
22          line: "{{ item.value.name }}_enable=\"NO\""
23          backup: "{{ cl_backup }}"
24        loop: "{{ cl_services|dict2items }}"
25        loop_control:
26          label: "{{ item.key }}"
27        when: "not item.value.enabled|default(true)|bool"
28
29    when: "ansible_os_family == 'FreeBSD'"
30
31  # All ---------------------------------------------------------------------
32  - name: "services: Manage services"
33    block:
34
35      - name: "services: Manage services"
36        service:
37          name: "{{ item.value.name }}"
38          state: "{{ item.value.state|default('started') }}"
39          enabled: "{{ item.value.enabled|default(true) }}"
40        loop: "{{ cl_services|dict2items }}"
41        loop_control:
42          label: "{{ item.key }}"
43
44    when: "ansible_os_family == 'RedHat' or
45           ansible_os_family == 'Debian' or
46           ansible_os_family == 'FreeBSD'"
47
48  # EOF
49  ...
```

EXAMPLES

Table of Contents

3.1 FreeBSD Postfix

3.1.1 Handlers

contrib/postfix/conf-light/handlers.d/postfix-freebsd

Synopsis: Create handlers for Postfix.

Use template (2) to create handlers.

[contrib/postfix/conf-light/handlers.d/postfix-freebsd]

```
1  postfix_freebsd:
2    template: handlers-auto1.yml.j2
3    params:
4
5      - handler: 'enable and start postfix'
6        module: service
7        params:
8          - 'name: postfix'
9          - 'state: started'
10         - 'enabled: true'
11
12     - handler: 'disable and stop postfix'
13       module: service
14       params:
15         - 'name: postfix'
16         - 'state: stopped'
17         - 'enabled: false'
18
19     - handler: 'reload postfix'
20       module: service
21       params:
22         - 'name: postfix'
23         - 'state: reloaded'
24       conditions:
25         - '- cl_service_postfix_enable|bool'
26
27     - handler: 'restart postfix'
28       module: service
29       params:
30         - 'name: postfix'
31         - 'state: restarted'
32       conditions:
33         - '- cl_service_postfix_enable|bool'
34
35     - handler: 'postfix check'
36       module: command
37       params:
38         - 'cmd: /usr/local/sbin/postfix check'
39
40     - handler: 'newaliases'
41       module: command
42       params:
43         - 'cmd: /usr/bin/newaliases'
44
45 #     - handler: 'postmap smtp sasl passwords'
```

(continues on next page)

```
46   #       module: command
47   #       params:
48   #         - 'cmd: /usr/local/sbin/postmap {{ postfix_main_cf_smtp_sasl_password_maps }}
     ↪'
49
50   #     - handler: 'postmap virtual aliases'
51   #       module: command
52   #       params:
53   #         - cmd: /usr/local/sbin/postmap {{ postfix_virtual }}'
```

See also:

See *setup.yml* how the handlers are created.

contrib/postfix/conf-light/handlers.d/sendmail-freebsd

Synopsis: Create handlers for Sendmail.

Use template (2) to create handlers.

[contrib/postfix/conf-light/handlers.d/sendmail-freebsd]

```
1    sendmail_freebsd:
2      template: handlers-auto1.yml.j2
3      params:
4
5        - handler: 'enable and start sendmail'
6          module: service
7          params:
8            - 'name: sendmail'
9            - 'state: started'
10           - 'enabled: true'
11
12       - handler: 'disable and stop sendmail'
13         module: service
14         params:
15           - 'name: sendmail'
16           - 'state: stopped'
17           - 'enabled: false'
18
19       - handler: 'reload sendmail'
20         module: service
21         params:
22           - 'name: sendmail'
23           - 'state: reloaded'
24         conditions:
25           - '- cl_service_sendmail_enable|bool'
26
27       - handler: 'restart sendmail'
28         module: service
29         params:
30           - 'name: sendmail'
31           - 'state: restarted'
32         conditions:
33           - '- cl_service_sendmail_enable|bool'
34
35       - handler: 'start sendmail'
```

```
36        module: service
37        params:
38          - 'name: sendmail'
39          - 'state: started'
40
41     - handler: 'stop sendmail'
42        module: service
43        params:
44          - 'name: sendmail'
45          - 'state: stopped'
```

See also:

See *setup.yml* how the handlers are created.

3.1.2 Packages

contrib/postfix/conf-light/packages.d/postfix

Synopsis: Install Postfix.

Use package or port (2) to install Postfix.

[contrib/postfix/conf-light/packages.d/postfix]

```
1  postfix:
2    name: 'mail/postfix'
```

See also:

See *packages.yml* how the FreeBSD packages or ports are installed.

3.1.3 Services

contrib/postfix/conf-light/services.d/postfix

Synopsis: Configure Postfix service.

Set service (2) state (3). Run the service on boot (4).

[contrib/postfix/conf-light/services.d/postfix]

```
1  postfix:
2    name: 'postfix'
3    state: '{{ cl_service_postfix_state }}'
4    enabled: '{{ cl_service_postfix_enable }}'
```

See also:

See custom Postfix variables *contrib/postfix/config-light-postfix.yml*. See *services.yml* how the services are configured.

ансible-role-config-light Documentation, Release 1.4.0

contrib/postfix/conf-light/services.d/sendmail

Synopsis: Configure Sendmail service.

Set service (2) state (3). Do not run the service on boot (4).

[contrib/postfix/conf-light/services.d/sendmail]

```
1  sendmail:
2    name: 'sendmail'
3    state: '{{ cl_service_sendmail_state }}'
4    enabled: '{{ cl_service_sendmail_enable }}'
```

See also:

See custom Postfix variables *contrib/postfix/config-light-postfix.yml*.

3.1.4 Files

contrib/postfix/config-light-postfix.yml

Synopsis: Custom variables for Postfix.

Put the host-specific variables (6) into the `host_vars`. Optionally other variables might be put into the `group_vars`.

[contrib/postfix/config-light-postfix.yml]

```
1  ---
2
3  # 28.4.2. Replace the Default MTA
4  # https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/mail-changingmta.html
5
6  cl_myhostname: host99.region9.example.com
7
8  # conf-light/files.d/mailer-conf
9  # Execute the Postfix sendmail program, named /usr/local/sbin/sendmail
10 cl_mailerconf:
11   - 'sendmail      /usr/local/sbin/sendmail'
12   - 'send-mail     /usr/local/sbin/sendmail'
13   - 'mailq         /usr/local/sbin/sendmail'
14   - 'newaliases    /usr/local/sbin/sendmail'
15
16 # conf-light/files.d/rc-rconf
17 cl_rcconf_postfix_enable: 'YES'
18 cl_rcconf_sendmail_enable: 'NO'
19 cl_rcconf_sendmail_submit_enable: 'NO'
20 cl_rcconf_sendmail_outbound_enable: 'NO'
21 cl_rcconf_sendmail_msp_queue_enable: 'NO'
22
23 # conf-light/files.d/periodic-conf
24 cl_periodicconf_daily_clean_hoststat_enable: "NO"
25 cl_periodicconf_daily_status_mail_rejects_enable: "NO"
26 cl_periodicconf_daily_status_include_submit_mailq: "NO"
27 cl_periodicconf_daily_submit_queuerun: "NO"
28
29 # Services
30 cl_service_sendmail_enable: false
```

(continues on next page)

```
31  cl_service_sendmail_state: 'stopped'
32  cl_service_postfix_enable: true
33  cl_service_postfix_state: 'started'
34
35  # If you are using SASL, you need to make sure that postfix has access
36  # to read the sasldb file.  This is accomplished by adding postfix to
37  # group mail and making the /usr/local/etc/sasldb* file(s) readable by
38  # group mail (this should be the default for new installs).
39
40  # EOF
41  ...
```

contrib/postfix/conf-light/files.d/mailer-conf

Synopsis: Create file.

Create file (2) from the template (7).

[contrib/postfix/conf-light/files.d/mailer-conf]

```
1  mailerconf:
2    path: '/etc/mail/mailer.conf'
3    force: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    template: 'mailer.conf.j2'
```

contrib/postfix/conf-light/files.d/periodic-conf

Synopsis: Modify file.

Modify file (2) with the lines (7).

[contrib/postfix/conf-light/files.d/periodic-conf]

```
1  periodic_conf:
2    path: '/etc/periodic.conf'
3    create: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    lines:
8      - regexp: '^daily_clean_hoststat_enable(.*)$'
9        line: 'daily_clean_hoststat_enable="{{ cl_periodicconf_daily_clean_hoststat_
   →enable }}"'
10     - regexp: '^daily_status_mail_rejects_enable(.*)$'
11       line: 'daily_status_mail_rejects_enable="{{ cl_periodicconf_daily_status_mail_
   →rejects_enable }}"'
12     - regexp: '^daily_status_include_submit_mailq(.*)$'
13       line: 'daily_status_include_submit_mailq="{{ cl_periodicconf_daily_status_
   →include_submit_mailq }}"'
14     - regexp: '^daily_submit_queuerun(.*)$'
15       line: 'daily_submit_queuerun="{{ cl_periodicconf_daily_submit_queuerun }}"'
```

contrib/postfix/conf-light/files.d/postfix-main-cf

Synopsis: Modify file and notify handlers.

Modify file (2) with the lines (9) and notify handlers (7).

[contrib/postfix/conf-light/files.d/postfix-main-cf]

```
1  postfix_main_cf:
2    path: '/usr/local/etc/postfix/main.cf'
3    create: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    handlers:
8      - 'postfix_freebsd reload postfix'
9    lines:
10     - regexp: '^myhostname\s*=\s*(.*)$'
11       line: 'myhostname = {{ cl_myhostname }}'
```

contrib/postfix/conf-light/files.d/rc-conf

Synopsis: Modify file.

Modify file (2) with the lines (7).

[contrib/postfix/conf-light/files.d/rc-conf]

```
1  rcconf:
2    path: '/etc/rc.conf'
3    create: true
4    owner: 'root'
5    group: 'wheel'
6    mode: '0644'
7    lines:
8      - regexp: '^sendmail_enable(.*)$'
9        line: 'sendmail_enable="{{ cl_rcconf_sendmail_enable }}"'
10     - regexp: '^sendmail_submit_enable(.*)$'
11       line: 'sendmail_submit_enable="{{ cl_rcconf_sendmail_submit_enable }}"'
12     - regexp: '^sendmail_outbound_enable(.*)$'
13       line: 'sendmail_outbound_enable="{{ cl_rcconf_sendmail_outbound_enable }}"'
14     - regexp: '^sendmail_msp_queue_enable(.*)$'
15       line: 'sendmail_msp_queue_enable="{{ cl_rcconf_sendmail_msp_queue_enable }}"'
16     - regexp: '^postfix_enable(.*)$'
17       line: 'postfix_enable="{{ cl_rcconf_postfix_enable }}"'
```

3.2 Armbian Simple SMTP

3.2.1 Packages

contrib/ssmtp/conf-light/packages.d/ssmtp

Synopsis: Install Simple SMTP.

Use package (2) to install sSMTP.

[contrib/ssmtp/conf-light/packages.d/ssmtp]

```
1  ssmtp:
2    name: 'ssmtp'
```

See also:

See *packages.yml* how the Linux packages are installed.

3.2.2 Files

contrib/ssmtp/config-light-ssmtp.yml

Synopsis: Custom variables for sSMTP.

Put the host-specific variables (7) into the `host_vars`. Optionally other variables might be put into the `group_vars`.

[contrib/ssmtp/config-light-ssmtp.yml]

```
1   ---
2
3   # sSMTP - Simple SMTP
4   # https://wiki.debian.org/sSMTP
5
6   # linux-postinstall FQDN
7   lp_fqdn: "host99.region9.example.com"
8
9   # NEVER USE PLAINTEXT PASSWORD. USE VAULT INSTEAD
10  smtp_client_password_mail_example_com: "PASSWORD"
11
12  # conf-light/files.d/ssmtp-conf
13  cl_ssmtp_srv: "mail.example.com"
14  cl_ssmtp_srv_domain: "example.com"
15
16  cl_ssmtp_postmaster_address: "postmaster@{{ lp_fqdn }}"
17  cl_ssmtp_mailhub: "{{ cl_ssmtp_srv }}:587"
18  cl_ssmtp_rewriteDomain: "{{ cl_ssmtp_srv_domain }}"
19
20  cl_ssmtp_UseTLS: "Yes"
21  cl_ssmtp_UseSTARTTLS: "Yes"
22
23  cl_ssmtp_AuthUser: "smtp_client"
24  cl_ssmtp_AuthPass: "{{ smtp_client_password_mail_example_com }}"
25  cl_ssmtp_AuthMethod: "LOGIN"
26
27  cl_ssmtp_FromLineOverride: "yes"
28
29  # conf-light/files.d/revaliases
30  cl_ssmtp_revaliases:
31    - "root:root@{{ cl_ssmtp_srv_domain }}:{{ cl_ssmtp_srv }}:587"
32    - "admin:admin@{{ cl_ssmtp_srv_domain }}:{{ cl_ssmtp_srv }}:587"
33    - "user1:user1@{{ cl_ssmtp_srv_domain }}:{{ cl_ssmtp_srv }}:587"
34
35  # EOF
36  ...
```

contrib/ssmtp/conf-light/files.d/revaliases

Synopsis: Create file.

Create file (2) from the template (7).

[contrib/ssmtp/conf-light/files.d/revaliases]

```yaml
1  revaliases:
2    path: '/etc/ssmtp/revaliases'
3    force: true
4    owner: 'root'
5    group: 'mail'
6    mode: 'u=rw,g=r'
7    template: 'revaliases.j2'
```

See also:

See template revaliases.j2. See how files are created from template *files-template.yml*.

contrib/ssmtp/conf-light/files.d/ssmtp-conf

Synopsis: Create file.

Create file (2) from the template (7).

[contrib/ssmtp/conf-light/files.d/ssmtp-conf]

```yaml
1  ssmtp_conf:
2    path: '/etc/ssmtp/ssmtp.conf'
3    force: true
4    owner: 'root'
5    group: 'mail'
6    mode: 'u=rw,g=r'
7    template: 'ssmtp.conf.j2'
```

See also:

See template ssmtp.conf.j2. See how files are created from template *files-template.yml*.

FOUR

COPYRIGHT

Redistribution and use in source (RST DocUtils) and 'compiled' forms (XML, HTML, PDF, PostScript, RTF and so forth) with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code (RST Docutils) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in compiled form (transformed to other DTDs, converted to PDF, PostScript, RTF and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

LEGAL NOTICE

All product names, logos, and brands are property of their respective owners.

SIX

INDICES AND TABLES

- genindex
- modindex
- search