Configuration Ansible

Configuration Ansible

Ce document propose d'examiner les principales options de configuration de Ansible sur le noeud de contrôle. Son architecture sans agent laisse le soin à chacun de configurer finement et personnellement le comportement par défaut de la solution.

Objectifs de certification

RHCE EX294 (RHEL8)

Si vous préparez un examen de certification, ce document poursuit les objectifs suivant :

- **2. Maîtrise des composants de base d'Ansible**
  - 2.7. Fichiers de configuration
  - 2.8. Utiliser la documentation fournie pour trouver des informations spécifiques aux modules et commandes Ansible
- **3. Installation et configuration d'un nœud de contrôle Ansible**
  - 3.3. Créer un fichier de configuration
  - 3.5. Gérer les parallélismes

Le comportement d'Ansible peut être influencé de différentes manières, dans l'ordre de précédence :

1. En configurant des variables d'environnement ;
2. en passant directement les paramètres sur la ligne de commande `ansible` ou `ansible-playbook` ;
3. en définissant des fichiers de configuration `ansible.cfg` dont la précédence dépend de l'emplacement.

1. Commande ansible-config

La commande `ansible-config dump` donne la liste des variables de configuration chargées dans le système de contrôle. Voyez-vous même :

```
ansible-config dump
```

Beaucoup de ces variables sont dans un format `ANSIBLE_*` ou `DEFAULT_*`.

2. Fichier de configuration ansible.cfg

On peut changer ces variables de configuration en renseignant un fichier de configuration. Ansible cherchera dans l'ordre (le premier trouvé sera utilisé et les autres seront ignorés)[1] :

1. Le contenu de la variable d'environnement `ANSIBLE_CONFIG` (Si la variable d'environnement est valorisée)
2. L'emplacement `./ansible.cfg` (dans le dossier courant, le répertoire de travail)
3. L'emplacement `~/.ansible.cfg` (à la racine du dossier utilisateur comme fichier caché)
4. L'emplacement `/etc/ansible/ansible.cfg` (dans le dossier de configuration du logiciel)

La commande `ansible-config view` permet de visualiser le contenu du fichier de configuration courant.

Le dépot GitHub d'Ansible offre un exemple de fichier de configuration commenté. Il est directement disponible sur cet URL :

```
curl -s https://raw.githubusercontent.com/ansible/ansible/devel/examples/ansible.cfg | less
```

On y trouvera une dizaine de sections dans le format INI :

- `[defaults]`
- `[inventory]`
- `[privilege_escalation]`
- `[paramiko_connection]`
- `[ssh_connection]`
- `[persistent_connection]`
- `[accelerate]`
- `[selinux]`
- `[colors]`
- `[diff]`

3. Section [defaults]

La section `[defaults]` est la plus intéressante, le sigle `#` au début de chaque ligne mettant le paramètre en commentaire :

3.1. Valeurs habituelles

La section “defaults” définit des variables utiles.

```
[defaults]

#inventory      = /etc/ansible/hosts
#library        = /usr/share/my_modules/
#module_utils   = /usr/share/my_module_utils/
#remote_tmp     = ~/.ansible/tmp
#local_tmp      = ~/.ansible/tmp
#plugin_filters_cfg = /etc/ansible/plugin_filters.yml

#forks          = 5
#poll_interval  = 15

#sudo_user      = root
#ask_sudo_pass = True
#ask_pass      = True
#transport      = smart
#remote_port    = 22
#module_lang    = C
#module_set_locale = False
```

On retiendra les variables :

- `inventory`
- `forks`
- `sudo_user`
- `ask_sudo_pass`
- `ask_pass`
- `remote_port`

3.2. Récupération des “facts”

Toujours dans la section default, la récupération des “facts” est activée par défaut : `gathering = implicit`. On peut aussi définir le type de “facts” collectés (`gather_subset`).

```
# plays will gather facts by default, which contain information about
# the remote system.
#
# smart - gather by default, but don't regather if already gathered
# implicit - gather by default, turn off with gather_facts: False
# explicit - do not gather by default, must say gather_facts: True
#gathering = implicit

# This only affects the gathering done by a play's gather_facts directive,
# by default gathering retrieves all facts subsets
# all - gather all subsets
# network - gather min and network facts
# hardware - gather hardware facts (longest facts to retrieve)
# virtual - gather min and virtual facts
# facter - import facts from facter
# ohai - import facts from ohai
# You can combine them using comma (ex: network,virtual)
# You can negate them using ! (ex: !hardware,!facter,!ohai)
# A minimal set of facts is always gathered.
#gather_subset = all

# some hardware related facts are collected
# with a maximum timeout of 10 seconds. This
# option lets you increase or decrease that
# timeout to something more suitable for the
# environment.
# gather_timeout = 10

# Ansible facts are available inside the ansible_facts.* dictionary
# namespace. This setting maintains the behaviour which was the default prior
# to 2.5, duplicating these variables into the main namespace, each with a
# prefix of 'ansible_'.
# This variable is set to True by default for backwards compatibility. It
# will be changed to a default of 'False' in a future release.
# ansible_facts.
# inject_facts_as_vars = True
```

3.3. Vérification des clés SSH

Il n’est pas nécessaire d’agréer manuellement les clés SSH des hôtes à gérer. Par défaut, le `host_key_checking` SSH est désactivé.

```
# uncomment this to disable SSH key host checking
#host_key_checking = False
```

3.4. Callback plugins

On peut activer la configuration des sorties à la suite des exécutions des tâches : horodatage, envoi de courriel, etc.

```
# change the default callback, you can only have one 'stdout' type  enabled at a time.
#stdout_callback = skippy
## Ansible ships with some plugins that require whitelisting,
## this is done to avoid running all of a type by default.
## These setting lists those that you want enabled for your system.
## Custom plugins should not need this unless plugin author specifies it.

# enable callback plugins, they can output to stdout but cannot be 'stdout' type.
#callback_whitelist = timer, mail
```

3.5. Handlers manquants

```
# Controls if a missing handler for a notification event is an error or a warning
#error_on_missing_handler = True
```

3.6. Timeout SSH

```
# SSH timeout
#timeout = 10
```

3.7. Utilisateur cible par défaut

Par défaut, c’est l’utilisateur local qui agit du même nom sur la cible distante.

```
# default user to use for playbooks if user is not specified
# (/usr/bin/ansible will use current user as default)
#remote_user = root
```

3.8. Logging

Le logging est désactivé tant qu’un chemin n’est pas défini dans la variable `log_path`.

```
# logging is off by default unless this path is defined
# if so defined, consider logrotate
#log_path = /var/log/ansible.log
```

3.9. Extensions Jinja2

On définit ici l’extension reconnue par Ansible des fichiers de modèles (templates) Jinja2.

```
# list any Jinja2 extensions to enable here:
#jinja2_extensions = jinja2.ext.do,jinja2.ext.i18n
```

3.10. Fichier de mot de passe ansible-vault

Pour simplifier la gestion des secrets, on peut encoder son mot de passe ansible-vault dans un emplacement protégé.

```
# If set, configures the path to the Vault password file as an alternative to
# specifying --vault-password-file on the command line.
#vault_password_file = /path/to/vault_password_file
```

3.11. Affichage

Ces paramètres permettent de contrôler finement l’affichage dans l’exécution des tâches.

```
# by default, ansible-playbook will display "Skipping [host]" if it determines a task
# should not be run on a host.  Set this to "False" if you don't want to see these "Skipping"
# messages. NOTE: the task header will still be shown regardless of whether or not the
# task is skipped.
#display_skipped_hosts = True

# by default, if a task in a playbook does not include a name: field then
# ansible-playbook will construct a header that includes the task's action but
# not the task's args.  This is a security feature because ansible cannot know
# if the *module* considers an argument to be no_log at the time that the
# header is printed.  If your environment doesn't have a problem securing
# stdout from ansible-playbook (or you have manually specified no_log in your
# playbook on all of the tasks where you have secret information) then you can
# safely set this to True to get more informative messages.
#display_args_to_stdout = False

# by default (as of 1.3), Ansible will raise errors when attempting to dereference
# Jinja2 variables that are not set in templates or action lines. Uncomment this line
# to revert the behavior to pre-1.3.
#error_on_undefined_vars = False

# by default (as of 1.6), Ansible may display warnings based on the configuration of the
# system running ansible itself. This may include warnings about 3rd party packages or
# other conditions that should be resolved if possible.
# to disable these warnings, set the following value to False:
#system_warnings = True

# by default (as of 1.4), Ansible may display deprecation warnings for language
# features that should no longer be used and will be removed in future versions.
# to disable these warnings, set the following value to False:
#deprecation_warnings = True

# (as of 1.8), Ansible can optionally warn when usage of the shell and
# command module appear to be simplified by using a default Ansible module
```

```
# instead.  These warnings can be silenced by adjusting the following
# setting or adding warn=yes or warn=no to the end of the command line
# parameter string.  This will for example suggest using the git module
# instead of shelling out to the git command.
# command_warnings = False
```

3.12. Types et emplacements des plugins

```
# set plugin path directories here, separate with colons
#action_plugins     = /usr/share/ansible/plugins/action
#cache_plugins      = /usr/share/ansible/plugins/cache
#callback_plugins   = /usr/share/ansible/plugins/callback
#connection_plugins = /usr/share/ansible/plugins/connection
#lookup_plugins     = /usr/share/ansible/plugins/lookup
#inventory_plugins  = /usr/share/ansible/plugins/inventory
#vars_plugins       = /usr/share/ansible/plugins/vars
#filter_plugins     = /usr/share/ansible/plugins/filter
#test_plugins       = /usr/share/ansible/plugins/test
#terminal_plugins   = /usr/share/ansible/plugins/terminal
#strategy_plugins   = /usr/share/ansible/plugins/strategy
```

3.13. Stratégie

Ansible propose principalement deux “stratégies” d’exécution :

- `linear` par défaut : Par défaut, les jeux fonctionnent avec une stratégie linéaire, dans laquelle tous les hôtes exécutent chaque tâche avant qu’un hôte ne commence la tâche suivante, en utilisant le nombre de “forks” (5 par défaut) pour établir un parallélisme.
- `free` qui permet à chaque hôte d’exécuter chaque tâche aussi vite qu’il le peut jusqu’à la fin du jeu.

```
# by default, ansible will use the 'linear' strategy but you may want to try
# another one
#strategy = free
```

3.14. Callbacks

```
# by default callbacks are not loaded for /bin/ansible, enable this if you
# want, for example, a notification or logging callback to also apply to
# /bin/ansible runs
#bin_ansible_callbacks = False
```

3.15. Cows

```
# don't like cows?  that's unfortunate.
# set to 1 if you don't want cowsay support or export ANSIBLE_NOCOWS=1
#nocows = 1

# set which cowsay stencil you'd like to use by default. When set to 'random',
# a random stencil will be selected for each task. The selection will be filtered
# against the `cow_whitelist` option below.
#cow_selection = default
#cow_selection = random

# when using the 'random' option for cowsay, stencils will be restricted to this list.
# it should be formatted as a comma-separated list with no spaces between names.
# NOTE: line continuations here are for formatting purposes only, as the INI parser
#       in python does not support them.
#cow_whitelist=bud-frogs,bunny,cheese,daemon,default,dragon,elephant-in-snake,elephant,eyes,\
#              hellokitty,kitty,luke-koala,meow,milk,moofasa,moose,ren,sheep,small,stegosaurus,\
#              stimpy,supermilker,three-eyes,turkey,turtle,tux,udder,vader-koala,vader,www
```

3.16. Couleurs

```
# don't like colors either?
# set to 1 if you don't want colors, or export ANSIBLE_NOCOLOR=1
#nocolor = 1
```

3.17. Mise en cache des “facts”

```
# if set to a persistent type (not 'memory', for example 'redis') fact values
# from previous runs in Ansible will be stored.  This may be useful when
# wanting to use, for example, IP information from one group of servers
# without having to talk to them in the same playbook run to get their
# current IP information.
#fact_caching = memory

#This option tells Ansible where to cache facts. The value is plugin dependent.
#For the jsonfile plugin, it should be a path to a local directory.
#For the redis plugin, the value is a host:port:database triplet: fact_caching_connection = localhost:6379:0

#fact_caching_connection=/tmp
```

3.18. Fichier Retry

```
# retry files
# When a playbook fails by default a .retry file will be created in ~/
# You can disable this feature by setting retry_files_enabled to False
# and you can change the location of the files by setting retry_files_save_path

#retry_files_enabled = False
#retry_files_save_path = ~/.ansible-retry
```

3.19. Squash action

```
# squash actions
# Ansible can optimise actions that call modules with list parameters
# when looping. Instead of calling the module once per with_ item, the
# module is called once with all items at once. Currently this only works
# under limited circumstances, and only with parameters named 'name'.
#squash_actions = apk,apt,dnf,homebrew,pacman,pkgng,yum,zypper
```

3.20. Journalisation des tâches

```
# prevents logging of task data, off by default
#no_log = False

# prevents logging of tasks, but only on the targets, data is still logged on the master/controller
#no_target_syslog = False
```

3.21. Divers

```
# controls whether Ansible will raise an error or warning if a task has no
# choice but to create world readable temporary files to execute a module on
# the remote machine.  This option is False by default for security.  Users may
# turn this on to have behaviour more like Ansible prior to 2.1.x.  See
# https://docs.ansible.com/ansible/become.html#becoming-an-unprivileged-user
# for more secure ways to fix this than enabling this option.
#allow_world_readable_tmpfiles = False

# controls the compression level of variables sent to
# worker processes. At the default of 0, no compression
# is used. This value must be an integer from 0 to 9.
#var_compression_level = 9

# controls what compression method is used for new-style ansible modules when
# they are sent to the remote system.  The compression types depend on having
# support compiled into both the controller's python and the client's python.
# The names should match with the python Zipfile compression types:
# * ZIP_STORED (no compression. available everywhere)
# * ZIP_DEFLATED (uses zlib, the default)
# These values may be set per host via the ansible_module_compression inventory
# variable
#module_compression = 'ZIP_DEFLATED'

# This controls the cutoff point (in bytes) on --diff for files
# set to 0 for unlimited (RAM may suffer!).
#max_diff_size = 1048576

# This controls how ansible handles multiple --tags and --skip-tags arguments
```

```
# on the CLI.  If this is True then multiple arguments are merged together.  If
# it is False, then the last specified argument is used and the others are ignored.
# This option will be removed in 2.8.
#merge_multiple_cli_flags = True

# Controls showing custom stats at the end, off by default
#show_custom_stats = True

# Controls which files to ignore when using a directory as inventory with
# possibly multiple sources (both static and dynamic)
#inventory_ignore_extensions = ~, .orig, .bak, .ini, .cfg, .retry, .pyc, .pyo

# This family of modules use an alternative execution path optimized for network appliances
# only update this setting if you know how this works, otherwise it can break module execution
#network_group_modules=eos, nxos, ios, iosxr, junos, vyos

# When enabled, this option allows lookups (via variables like {{lookup('foo')}} or when used as
# a loop with `with_foo`) to return data that is not marked "unsafe". This means the data may contain
# jinja2 templating language which will be run through the templating engine.
# ENABLING THIS COULD BE A SECURITY RISK
#allow_unsafe_lookups = False

# set default errors for all plays
#any_errors_fatal = False
```

4. Section [inventory]

```
[inventory]
# enable inventory plugins, default: 'host_list', 'script', 'auto', 'yaml', 'ini', 'toml'
#enable_plugins = host_list, virtualbox, yaml, constructed

# ignore these extensions when parsing a directory as inventory source
#ignore_extensions = .pyc, .pyo, .swp, .bak, ~, .rpm, .md, .txt, ~, .orig, .ini, .cfg, .retry

# ignore files matching these patterns when parsing a directory as inventory source
```

```
#ignore_patterns=

# If 'true' unparsed inventory sources become fatal errors, they are warnings otherwise.
#unparsed_is_failed=False
```

5. Section [privilege_escalation]

```
[privilege_escalation]
#become=True
#become_method=sudo
#become_user=root
#become_ask_pass=False
```

6. Connexion SSH

```
[paramiko_connection]

# uncomment this line to cause the paramiko connection plugin to not record new host
# keys encountered.  Increases performance on new host additions.  Setting works independently of the
# host key checking setting above.
#record_host_keys=False

# by default, Ansible requests a pseudo-terminal for commands executed under sudo. Uncomment this
# line to disable this behaviour.
#pty=False

# paramiko will default to looking for SSH keys initially when trying to
# authenticate to remote devices.  This is a problem for some network devices
# that close the connection after a key failure.  Uncomment this line to
# disable the Paramiko look for keys function
#look_for_keys = False

# When using persistent connections with Paramiko, the connection runs in a
# background process.  If the host doesn't already have a valid SSH key, by
```

```
# default Ansible will prompt to add the host key.  This will cause connections
# running in background processes to fail.  Uncomment this line to have
# Paramiko automatically add host keys.
#host_key_auto_add = True
```

```
[ssh_connection]

# ssh arguments to use
# Leaving off ControlPersist will result in poor performance, so use
# paramiko on older platforms rather than removing it, -C controls compression use
#ssh_args = -C -o ControlMaster=auto -o ControlPersist=60s

# The base directory for the ControlPath sockets.
# This is the "%(directory)s" in the control_path option
#
# Example:
# control_path_dir = /tmp/.ansible/cp
#control_path_dir = ~/.ansible/cp

# The path to use for the ControlPath sockets. This defaults to a hashed string of the hostname,
# port and username (empty string in the config). The hash mitigates a common problem users
# found with long hostnames and the conventional %(directory)s/ansible-ssh-%%h-%%p-%%r format.
# In those cases, a "too long for Unix domain socket" ssh error would occur.
#
# Example:
# control_path = %(directory)s/%%h-%%r
#control_path =

# Enabling pipelining reduces the number of SSH operations required to
# execute a module on the remote server. This can result in a significant
# performance improvement when enabled, however when using "sudo:" you must
# first disable 'requiretty' in /etc/sudoers
#
# By default, this option is disabled to preserve compatibility with
# sudoers configurations that have requiretty (the default on many distros).
#
#pipelining = False

# Control the mechanism for transferring files (old)
#   * smart = try sftp and then try scp [default]
```

```
#   * True = use scp only
#   * False = use sftp only
#scp_if_ssh = smart

# Control the mechanism for transferring files (new)
# If set, this will override the scp_if_ssh option
#   * sftp  = use sftp to transfer files
#   * scp   = use scp to transfer files
#   * piped = use 'dd' over SSH to transfer files
#   * smart = try sftp, scp, and piped, in that order [default]
#transfer_method = smart

# if False, sftp will not use batch mode to transfer files. This may cause some
# types of file transfer failures impossible to catch however, and should
# only be disabled if your sftp version has problems with batch mode
#sftp_batch_mode = False

# The -tt argument is passed to ssh when pipelining is not enabled because sudo
# requires a tty by default.
#usetty = True

# Number of times to retry an SSH connection to a host, in case of UNREACHABLE.
# For each retry attempt, there is an exponential backoff,
# so after the first attempt there is 1s wait, then 2s, 4s etc. up to 30s (max).
#retries = 3
```

```
[persistent_connection]

# Configures the persistent connection timeout value in seconds.  This value is
# how long the persistent connection will remain idle before it is destroyed.
# If the connection doesn't receive a request before the timeout value
# expires, the connection is shutdown. The default value is 30 seconds.
#connect_timeout = 30

# The command timeout value defines the amount of time to wait for a command
# or RPC call before timing out. The value for the command timeout must
# be less than the value of the persistent connection idle timeout (connect_timeout)
# The default value is 30 second.
```

```
#command_timeout = 30

[accelerate]
#accelerate_port = 5099
#accelerate_timeout = 30
#accelerate_connect_timeout = 5.0

# The daemon timeout is measured in minutes. This time is measured
# from the last activity to the accelerate daemon.
#accelerate_daemon_timeout = 30

# If set to yes, accelerate_multi_key will allow multiple
# private keys to be uploaded to it, though each user must
# have access to the system via SSH to add a new key. The default
# is "no".
#accelerate_multi_key = yes
```

7. Section [selinux]

```
[selinux]
# file systems that require special treatment when dealing with security context
# the default behaviour that copies the existing context or uses the user default
# needs to be changed to use the file system dependent context.
#special_context_filesystems=nfs,vboxsf,fuse,ramfs,9p,vfat

# Set this to yes to allow libvirt_lxc connections to work without SELinux.
#libvirt_lxc_noseclabel = yes
```

8. Sections [colors] et [diff]

```
[colors]
#highlight = white
#verbose = blue
#warn = bright purple
```

```
#error = red
#debug = dark gray
#deprecate = purple
#skip = cyan
#unreachable = red
#ok = green
#changed = yellow
#diff_add = green
#diff_remove = red
#diff_lines = cyan
```

```
[diff]
# Always print diff when running ( same as always running with -D/--diff )
# always = no

# Set how many context lines to show in diff
# context = 3
```

9. Plugins de connexion

Connection Plugins

Les plug-ins de connexion permettent à Ansible de se connecter aux hôtes cibles afin d'exécuter des tâches sur ceux-ci. Ansible est livré avec de nombreux plugins de connexion, mais un seul peut être utilisé par hôte à la fois. Les plus utilisés sont les types de connexion Paramiko SSH, ssh natif (appelé simplement ssh) et local.

On peut utiliser `ansible-doc -t connection -l` pour voir la liste des plugins disponibles. `ansible-doc -t <nom du plug-in>` permet d'afficher une documentation détaillée et des exemples.

- buildah Interact with an existing buildah container
- chroot Interact with local chroot
- docker Run tasks in docker containers
- funcd Use funcd to connect to target
- httpapi Use httpapi to run command on network appliances
- iocage Run tasks in iocage jails
- jail Run tasks in jails
- kubectl Execute tasks in pods running on Kubernetes.
- libvirt_lxc Run tasks in lxc containers via libvirt
- local execute on controller

- lxc Run tasks in lxc containers via lxc python library
- lxd Run tasks in lxc containers via lxc CLI
- netconf Provides a persistent connection using the netconf protocol
- network_cli Use network_cli to run command on network appliances
- oc Execute tasks in pods running on OpenShift.
- paramiko_ssh Run tasks via python ssh (paramiko)
- persistent Use a persistent unix socket for connection
- psrp Run tasks over Microsoft PowerShell Remoting Protocol
- saltstack Allow ansible to piggyback on salt minions
- ssh connect via ssh client binary
- winrm Run tasks over Microsoft's WinRM
- zone Run tasks in a zone instance

Par défaut, c'est le plugion de connexion `ssh` sui est utilisé. Il se définit plus finement avec ses paramètres adaptés comme variable de l'hôte lui-même.

10. Automation système Linux

Voici un exemple de fichier de configuration utilisé dans le cadre de l'automation de serveurs Linux :

```
cat << EOF >> ansible.cfg
[defaults]
inventory = ./inventory.ini
host_key_checking = False
private_key_file = /root/.ssh/id_rsa
callback_whitelist = profile_tasks
forks = 20
#strategy = free
gathering = explicit
become = True
[callback_profile_tasks ]
task_output_limit = 100
EOF
```

On remarquera les paramètres suivants du fichier `ansible.cfg` :

- `inventory`: l'emplacement de l'inventaire utilisé
- `private_key_file`: l'emplacement de la clé publique pour les connexions SSH (ici le paramètre est commenté)

1. Ansible Configuration Settings ↩

Revision #1
Created 3 October 2021 21:19:04 by garfi
Updated 3 October 2021 21:19:38 by garfi