



Préparation à la certification **RHCE** **Red Hat Enterprise Linux 8**

EXAMEN EX294

17 Travaux pratiques
84 Questions réponses

INCLUS :
UN EXAMEN BLANC



Philippe PINCHON



Les éléments à télécharger sont disponibles à l'adresse suivante :

<http://www.editions-eni.fr>

Saisissez la référence ENI de l'ouvrage **CERHCE** dans la zone de recherche et validez.

Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Chapitre 1

	Introduction
A. À qui s'adresse ce livre ?	15
B. Les conditions de l'examen	15
C. Par où commencer ?	16
D. La certification RHCSA.	16
E. La certification RHCE.	18
1. Cursus RHEL 7	18
2. Cursus RHEL 8	19
F. Comment actualiser sa certification RHCE ?	20
G. Après la certification RHCE ?	20

Chapitre 2

	Création d'un laboratoire
A. Quels outils vous faut-il ?	26
1. Machines nécessaires	26
2. Solutions alternatives.	26
B. Le système d'hôte et l'hyperviseur	27
1. Installation du système d'hôte	27
a. Obtenir l'image ISO.	27
b. Générer une clé USB amorçable	27
c. Bouter le PC sur la clé USB.	30
d. Processus d'installation	31
e. Tâches post-installation.	48
2. L'hyperviseur	53
a. Présentation de KVM.	54
b. Installation de KVM	54
c. Configuration de KVM.	55
C. Machine virtuelle modèle	61
1. Création de la machine virtuelle modèle.	61
2. Installation du système invité.	70

D. Machines virtuelles de travail	75
1. Clonage manuel	75
2. Clonage par script	76
E. Test des machines virtuelles	78
1. Test de la VM server1	78
2. Test de la VM server2	80
3. Test de la VM server3	82

Chapitre 3

Présentation de Ansible

A. Qu'est-ce que Ansible ?	88
B. Architecture	88
1. Le nœud de contrôle	88
2. Les hôtes gérés	89
3. Play et playbook	89
4. Tâches et modules	89
5. Plug-ins	90
6. Ansible Tower et AWX	90
a. Console web	91
b. API REST	91
7. Ansible Vault	91
8. Ansible Galaxy	91
9. Configuration Management DataBase	91
10. Réseaux	92
C. Installation de Ansible	92
1. Prérequis	92
2. Processus d'installation sur RHEL 8.0	93
3. Processus d'installation sur CentOS 8.0	93
4. Tâches post-installation	95
a. Afficher la version de Ansible	95
b. Afficher la configuration	95
c. Tester la communication	96
D. Hôtes gérés Unix et Linux	100
E. Hôtes gérés Microsoft Windows	101
F. Hôtes gérés réseau	101
G. Validation des acquis : questions/réponses	101

H. Travaux pratiques	103
1. Installer Ansible	103

Chapitre 4	Déploiement
A. Gestion des inventaires	108
1. Inventaire statique	108
a. Fichier d'inventaire au format INI	108
b. Fichier d'inventaire au format YAML	111
c. Manipulation de fichiers d'inventaire	120
2. Inventaire dynamique	121
a. Utilisation de scripts fournis	121
b. Utilisation de vos propres scripts	122
c. Exécution de scripts d'inventaire dynamiques	123
3. Inventaires multiples	123
B. Configuration de Ansible	124
1. Emplacement du fichier ansible.cfg	124
2. Directives du fichier de configuration	125
a. Section [defaults]	126
b. Section [privilege_escalation]	126
C. Utilisation des commandes ad hoc	126
1. Module de fichiers	127
2. Modules pour gérer les paquets logiciels	128
3. Modules du système	131
4. Module shell	134
5. Collecter les faits	135
D. Validation des acquis : questions/réponses	135
E. Travaux pratiques	139
1. Les fichiers d'inventaire	139
2. Configurer ansible	140
3. Utilisation des commandes ad hoc	141

Chapitre 5	Playbooks
A. Définition	146
B. Écriture d'un playbook	146
C. Exécution de playbooks	147
1. Principe de fonctionnement	147
2. Verbosité des playbooks	148
3. Vérification de la syntaxe	148
a. Vérification avec Ansible-playbook	148
b. Vérification avec yamllint	150
4. Exécution à blanc d'un playbook	150
D. Validation des acquis : questions/réponses	151
E. Travaux pratiques	152
1. Installation de Apache HTTP Server	152
2. Déployer et configurer mariADB	153
Chapitre 6	Variables Ansible
A. Définition d'une variable	158
1. Nom d'une variable	158
2. Étendue des variables	158
3. Types de variables	164
a. Types primitifs	165
b. Dictionnaires	165
B. Utilisation de variables Ansible	166
C. Récupérer le résultat d'une commande	168
D. Variables externes	172
E. Variables définies sur la ligne de commande	173
F. Chiffrer les variables	175
1. Présentation de Ansible Vault	175
2. Gestion d'un fichier chiffré	176
a. Création	176
b. Consultation	177
c. Modification	177
d. Chiffrer un fichier existant	178
e. Déchiffrer un fichier existant	178
f. Changer le mot de passe d'un fichier chiffré	179
g. Exécution d'un playbook	179

G. Validation des acquis : questions/réponses	180
H. Travaux pratiques	183
1. Manipulation de variables	183
2. Manipulation de variables chiffrées	185

Chapitre 7	Gestion des faits
-------------------	--------------------------

A. Présentation des faits de Ansible	190
1. Collecte des faits	190
2. Variable ansible_facts	192
3. Activer ou désactiver la collecte	193
B. Faits personnalisés	193
1. Stockage des faits	193
2. Faits statiques	195
3. Faits dynamiques	195
C. Module set_fact	199
D. Variables magiques	200
1. Variable magique hostvars	200
2. Variable magique groups	201
3. Variable magique group_names	202
4. Variable magique inventory_hostname	202
E. Validation des acquis : questions/réponses	203
F. Travaux pratiques	205
1. Gestion des faits de Ansible et des faits personnalisés	205

Chapitre 8	Contrôle de tâches
-------------------	---------------------------

A. Itération de tâches	210
1. Itération sur une simple liste	210
2. Itération sur une liste de hachage	212
3. Itération sur un dictionnaire	215
4. Capturer le résultat d'une tâche en boucle	217
B. Tâches conditionnelles	219
1. Conditions simples	219
2. Conditions multiples	230
a. Conditions combinées avec le mot-clé and	230
b. Conditions combinées avec le mot-clé or	231

C. Gestionnaires Ansible	231
D. Gestion des erreurs de tâche	232
1. Ignorer l'échec d'une tâche	232
2. Forcer l'exécution des gestionnaires	234
3. Spécifier les conditions d'échec d'une tâche	236
a. failed_when	236
b. fail	237
4. Spécifier une tâche signalant un résultat « Changed »	238
5. Gérer les erreurs au sein des blocs Ansible	239
E. Validation des acquis : questions/réponses	241
F. Travaux pratiques	243
1. Boucles et tâches conditionnelles	243
2. Gestionnaires Ansible	243
3. Échecs de tâche	244

Chapitre 9**Manipulation de fichiers sur les hôtes**

A. Introduction	250
B. Présentation des modules de fichiers	250
C. Appliquer les ACL de fichiers	251
D. Modifier l'horodatage des fichiers	252
E. Copie de fichiers	254
1. Module copy	254
2. Module fetch	257
F. Suppression de fichiers	259
G. Déplacement et renommage des fichiers	259
H. Recherche de fichiers	260
I. Informations sur l'état d'un fichier	263
J. Manipulation d'archives	264
1. Création d'une archive	264
2. Extraction d'une archive	265
K. Modification de fichiers	266
1. Module lineinfile	266
2. Module blockinfile	267
3. Module replace	268

L. Modifications de contexte de fichier SELinux	269
M. Synchronisation de fichiers	271
N. Modèles JINJA2	272
1. Présentation de Jinja2	272
2. Conception et application d'un modèle	272
3. Structures de contrôle	274
a. Boucles	274
b. Conditions	274
c. Opérateurs	275
4. Filtres	278
a. Filtres de variable	278
b. Filtres utilisés avec des conditions	279
5. Déployer un fichier personnalisé	280
O. Validation des acquis : questions/réponses	281
P. Travaux pratiques	283
1. Gestion de fichiers	283
2. Archivage	284
3. Apache	285

Chapitre 10**Gestion des rôles Ansible**

A. Présentation des rôles	292
B. Structure des rôles	292
C. Variables	293
1. Variables de rôle	293
2. Variables par défaut	293
D. Utilisation de rôles	294
1. Fonctionnement des rôles	294
2. Contrôle de l'ordre d'exécution	294
E. Rôles système	301
1. Présentation des rôles système	301
a. Rôles système avec support	301
b. Rôles système en préversion technologique	301
c. Rôles système en cours de développement	301
2. Installation de rôles système	302
3. Accès à la documentation des rôles système	304

4. Utilisation des rôles système	305
a. rhel-system-roles.kdump	305
b. rhel-system-roles.network	307
c. rhel-system-roles.selinux	308
d. rhel-system-roles.storage	312
e. rhel-system-roles.timesync	314
F. Création de rôles	314
1. Création de la structure du dossier	314
2. Définition du contenu du rôle	317
G. Déploiement de rôle	321
1. Ansible Galaxy	321
a. Présentation de Ansible Galaxy	321
b. Aide et documentation sur Ansible Galaxy	321
c. Rechercher des rôles	323
2. Commande ansible-galaxy	324
a. Recherche de rôles	324
b. Information d'un rôle	324
c. Installation de rôles	325
d. Gestion des rôles locaux	327
H. Validation des acquis : questions/réponses	328
I. Travaux pratiques	330
1. Utilisation d'un rôle système	330
2. Création d'un rôle	332

Chapitre 11**Dépannage**

A. Introduction	338
B. Outils de dépannage	338
1. Système de journalisation	338
2. Syntaxe YAML	339
a. Configuration de Vim	339
b. Utilitaire yamllint	341
3. Commande ansible-playbook	342
a. Option --syntax-check	342
b. Option --check	342
c. Option --step	343
d. Option --start-at-task	344
4. Débogage des tâches	344

5. Fonctionnement de Ansible	346
C. Problèmes de connexion avec les hôtes gérés	346
1. Configuration réseau	347
2. Identification et authentification	347
3. Utilisation de commandes ad hoc	350
D. Résolution de problèmes concernant les playbooks	350
1. Analyser la sortie à l'écran	350
2. Utilisation du mode check	352
a. Prise en charge du mode check	353
b. Activation ou désactivation du mode check	353
c. Voir les modifications apportées	353
E. Quelques bonnes pratiques	358
1. Définir le nom des plays et des tâches	358
a. Nom d'un play	358
b. Nom d'une tâche	358
2. Commentaire et documentation	359
3. Rédaction du code	359
F. Validation des acquis : questions/réponses	360

Chapitre 12**Examen blanc**

A. Préparation	365
1. Les objectifs de l'examen	365
2. Élaboration de l'environnement	365
3. Quelques conseils	365
B. Examen blanc	366
1. Installer et configurer Ansible	366
2. Commandes Ad-hoc	366
3. Modifier le fichier motd	367
4. Configurer le serveur SSH	367
5. Utiliser Ansible Vault	367
6. Gestion des utilisateurs	367
7. Tâches programmées	368
8. Créer un rôle pour MariaDB	368
9. Apache	369
10. Utiliser un rôle de Ansible Galaxy	369
11. SELinux	369
12. Créer une archive	370

13. Gestion de paquets	370
14. Systemd	370
15. Configurer les paramètres du noyau	370
16. Modèle Jinja2	370
17. Ansible Facts.	371
C. Correction de l'examen blanc	371
1. Correction : installation et configuration de Ansible	371
2. Correction : commandes Ad-hoc	373
3. Correction : modifier le fichier motd.	374
4. Correction : configurer le serveur SSH	375
5. Correction : utiliser Ansible Vault	376
6. Correction : gestion des utilisateurs	377
7. Tâche 7 : tâches programmées.	378
8. Correction : créer un rôle pour MariaDB	379
9. Correction : Apache.	383
10. Correction : utiliser un rôle de Ansible Galaxy	385
11. Correction : SELinux.	386
12. Correction : créer une archive	387
13. Correction : gestion de paquets	387
14. Correction : systemd	388
15. Correction : configurer les paramètres du noyau.	389
16. Correction : modèle Jinja2.	389
17. Correction : Ansible Facts	390

Chapitre 13**Corrections des travaux pratiques**

A. Corrections du TP chapitre Présentation de Ansible	395
1. Installer Ansible	395
B. Corrections du TP chapitre Déploiement	401
1. Les fichiers d'inventaire	401
2. Configurer ansible	404
3. Utilisation des commandes ad hoc	406
C. Corrections du TP chapitre Playbooks.	408
1. Installation de Apache HTTP Server	408
2. Déployer et configurer mariaDB	412
D. Corrections du TP chapitre Variables Ansible	416
1. Manipulation de variables	416
2. Manipulation de variables chiffrées	419

E. Corrections du TP chapitre Gestion des faits	424
1. Gestion des faits de Ansible et des faits personnalisés	424
F. Corrections du TP chapitre Contrôle de tâches	428
1. Boucles et tâches conditionnelles	428
2. Gestionnaires Ansible	430
3. Échecs de tâche	434
G. Corrections du TP du chapitre Gestion de fichiers sur les hôtes	436
1. Gestion de fichiers	436
2. Archivage	439
3. Apache	442
H. Corrections du TP du chapitre Gestion des rôles Ansible	448
1. Utilisation d'un rôle système	448
2. Création d'un rôle	455
 Tableau des objectifs	461
Index	465

Prérequis



Avoir étudié les chapitres Présentation de Ansible et Déploiement.

Objectifs

- Présenter le playbook de Ansible.
- Appréhender la syntaxe YAML pour concevoir un playbook.
- Vérifier la syntaxe d'un playbook.
- Exécuter un playbook.

A. Définition

Il peut s'avérer qu'exécuter une à une les différentes commandes ad hoc devienne assez rapidement rébarbatif. D'autant plus qu'elles exécutent une tâche simple ponctuellement sur un ou plusieurs hôtes gérés.

S'il faut lancer des tâches plus élaborées sur des hôtes et pouvoir les reproduire facilement, il est alors possible d'utiliser les playbooks, lesquels représentent un ensemble de scripts d'automatisation que l'on appelle « plays ».

Les playbooks définissent, donc, les tâches de gestion de configuration à accomplir sur les hôtes gérés.

B. Écriture d'un playbook

Vous pouvez concevoir des playbooks de bout en bout ou à partir de modules disponibles au sein de la plateforme, voire depuis la communauté d'utilisateurs.

Le format des fichiers playbooks est YAML. Nous avons déjà rencontré ce langage dans le chapitre Présentation de Ansible pour la rédaction des inventaires. Vous allez de nouveau utiliser l'éditeur de texte vi avec des paramètres spécifiques à YAML dans le fichier de configuration `~/.vimrc`.

Exemple

Avant de commencer, créez un répertoire de travail nommé **workspace** :

```
root@server1 ~]# mkdir workspace ; cd workspace
[root@server1 workspace]#
```

Sur les hôtes `server2` et `server3`, qui sont des plateformes Linux, créez un utilisateur avec les caractéristiques suivantes :

Nom : `john`

Groupe primaire : `users`

Groupes supplémentaires : `adm` et `sys`.

Vous pouvez appliquer la commande ad hoc ci-dessous pour effectuer cette tâche :

```
[root@server1 workspace]# ansible -m user -a "name=john \
> group=users groups=adm,sys state=present" all
[root@server1 workspace]#
```

Le module `user` de Ansible gère les comptes utilisateurs des plateformes Linux, macOS et Unix. Le module à utiliser, quant aux plateformes Microsoft Windows, est `win_user`.

Passez des arguments au module `user` avec le commutateur `-a` :

Paramètres	Valeurs
<code>name</code>	Nom de l'utilisateur à créer, supprimer ou modifier.
<code>group</code>	Nom du groupe primaire.
<code>groups</code>	Noms des groupes supplémentaires.
<code>state</code>	Si le compte doit exister ou non, prendre des mesures si l'état est différent de ce qui est indiqué. La valeur peut être « <code>present</code> » ou « <code>absent</code> ».

Le mot `all` définit que l'action est destinée à tous les hôtes de l'inventaire.

Cette commande peut être aussi écrite en tant que « `play` » dans un fichier playbook que l'on va nommer `user_john.yml` :

```
[root@server1 workspace]# cat user_john.yml
---
- name: Création de l'utilisateur john
  hosts: all
  tasks:
    - name: john Doe
      user:
        name: john
        group: users
        groups: adm,sys
        state: present
[root@server1 workspace]#
```

La balise `name` spécifie le nom du playbook Ansible. N'importe quel nom logique peut être donné.

La balise `hosts` spécifie la liste des hôtes ou le groupe d'hôtes à gérer.

La balise `vars` permet de définir les variables que vous souhaitez utiliser dans le playbook. Leur utilisation est similaire aux variables rencontrées dans n'importe quel langage de programmation.

La balise `tasks` décrit la liste d'actions à effectuer. Tous les playbooks doivent contenir des tâches ou une liste de tâches à exécuter. Chaque tâche est liée à un module Ansible, qui est exécuté avec les arguments.

C. Exécution de playbooks

Sur le nœud de contrôle, la commande `ansible-playbook` exécute les playbooks. Il faut lui fournir en tant qu'argument le nom du playbook à utiliser.

1. Principe de fonctionnement

La définition des attributs `name` dans un playbook qui contient plusieurs plays et tâches permet de surveiller la progression de leur exécution.

Gathering Facts est une tâche particulière qui, généralement, est exécutée automatiquement par le module `setup` au début d'un play. Ce sujet est abordé dans le chapitre Gestion des faits.

Les plays et les tâches sont traités dans l'ordre dans lequel ils sont écrits au sein du playbook. Lors de leur exécution, la sortie générée indique les résultats de chaque tâche effectuée.

Exécution du playbook user_john.yml

```
[root@server1 workspace]# ansible-playbook user_john.yml
PLAY [Création de l'utilisateur john]*****
TASK [Gathering Facts]*****
ok: [172.16.32.1]
ok: [172.16.32.2]

TASK [john Doe]*****
```

```

changed: [172.16.32.1]
changed: [172.16.32.2]

PLAY RECAP*****
172.16.32.1  : ok=2  changed=1  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
172.16.32.2  : ok=2  changed=1  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

[root@server1 workspace]#

```

Vous pouvez, sans crainte, exécuter un playbook plusieurs fois du fait que les tâches au sein de celui-ci sont idempotentes.

2. Verbosité des playbooks

La commande `ansible-playbook` fournit, par défaut, peu d'informations sur l'exécution d'une tâche. Pour obtenir un retour plus détaillé, il existe quatre niveaux de verbosité :

Commutateurs	Description
<code>-v</code>	Afficher les résultats de la tâche.
<code>-vv</code>	Afficher les résultats et la configuration de la tâche.
<code>-vvv</code>	Afficher les résultats et la configuration de la tâche et aussi les informations sur les connexions aux hôtes cibles.
<code>-vvvv</code>	Idem que <code>-vvv</code> avec des informations détaillées supplémentaires telles que l'activation du débogage de connexion.

3. Vérification de la syntaxe

Vous disposez de deux outils pour effectuer la vérification syntaxique :

- `ansible-playbook` avec le commutateur `--syntax-check` ;
- `yamllint`.

 La commande qu'il est conseillé d'utiliser le jour de l'examen est `ansible-playbook`. Elle est, de fait, disponible sur le nœud de contrôle tandis que `yamllint` doit être installée depuis l'EPEL comme nous l'avons vu dans le chapitre Déploiement.

a. Vérification avec Ansible-playbook

Il est préférable de réaliser une vérification syntaxique avant d'exécuter un playbook et de s'assurer ainsi que son contenu est correct. Il faut, pour cela, utiliser le commutateur `--syntax-check` de la commande `ansible-playbook` :

```

[root@server1 workspace]# ansible-playbook --syntax-check \
> user_john.yml -vv
ansible-playbook 2.8.5
  config file = /root/playbook/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules',
  '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.6/site-packages/ansible
  executable location = /usr/bin/ansible-playbook
  python version = 3.6.8 (default, Oct  7 2019, 17:58:22)

```

```
[GCC 8.2.1 20180905 (Red Hat 8.2.1-3)]
Using /root/playbook/ansible.cfg as config file
1 plays in user_john.yml

playbook: user_john.yml
[root@server1 workspace]#
```

Il est aussi possible de combiner la vérification syntaxique et la verbosité comme vous pouvez le constater.

La détection d'une erreur de syntaxe se représente de cette façon :

```
[root@server1 workspace]# ansible-playbook --syntax-check \
> user_john.yml -vv
ansible-playbook 2.8.5
  config file = /root/playbook/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules',
  '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.6/site-packages/ansible
  executable location = /usr/bin/ansible-playbook
  python version = 3.6.8 (default, Oct 7 2019, 17:58:22)
[GCC 8.2.1 20180905 (Red Hat 8.2.1-3)]
Using /root/playbook/ansible.cfg as config file
ERROR! Syntax Error while loading YAML.
  could not find expected ':'
```

The error appears to be in '/root/playbook/user_john.yml': line 7, column 13, but may be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

```
user
  name: john
    ^ here
```

```
[root@server1 workspace]#
```

L'erreur se situe approximativement sur la ligne 7 qui est pointée par un curseur « ^ here ». Généralement, elle se situe sur une ligne située avant. Pour vous en assurer, consultez le contenu du fichier *user_john.yml*:

```
[root@server1 workspace]# cat -n user_john.yml
 1      ---
 2      - name: Création de l'utilisateur john
 3        hosts: all
 4        tasks:
 5          - name: john Doe
 6            user
 7              name: john
 8              group: users
 9              groups: adm,sys
```

```
10           state: present
[root@server1 workspace]#
```

L'erreur est en réalité sur la ligne 6. Un caractère « : » est absent derrière le mot-clé user.

b. Vérification avec yamllint

En reprenant le même fichier erroné que précédemment :

```
[root@server1 workspace]# yamllint user_john.yml -vv
user_john.yml
 7:13      error      syntax error: could not find expected ':' (syntax)

[root@server1 workspace]#
```

L'erreur se situe sur la ligne 7, colonne 13, c'est-à-dire sur le caractère « : » situé après le mot-clé name : Yamllint s'est arrêté également au même endroit que la commande ansible-playbook.

4. Exécution à blanc d'un playbook

Les commutateurs -C ou --check effectuent l'exécution à blanc d'un playbook. Durant son exécution, les modifications qui seraient réalisées sont affichées sur l'écran sans réaliser le moindre changement sur les hôtes à gérer :

```
[root@server1 workspace]# ansible-playbook -C user_john.yml

PLAY [Création de l'utilisateur john]*****
TASK [Gathering Facts]*****
ok: [10.14.8.2]
ok: [10.14.8.1]

TASK [john Doe] *****
ok: [10.14.8.2]
ok: [10.14.8.1]

PLAY RECAP *****
10.14.8.1 : ok=2  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
10.14.8.2 : ok=2  changed=0  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

[root@server1 workspace]#
```

D. Validation des acquis : questions/réponses

Questions

Le but est de valider les points essentiels que nous avons évoqués dans ce chapitre.

- 1 Dans quel langage peut être écrit un fichier playbook ?
 - 2 Dans quel ordre sont exécutés les plays et les tâches dans un playbook ?
 - 3 Comment pouvez-vous vérifier la syntaxe du fichier site_fr.yml ?
 - 4 Quel est le commutateur qui permet d'afficher les résultats et la configuration de la tâche d'un playbook ?
 - 5 Comment identifier la configuration qui est utilisée par Ansible ?
 - 6 Comment devez-vous exécuter à blanc le fichier playbook.yml ?
 - 7 Comment pouvez-vous concevoir des playbooks ?
 - 8 Quel est le nom de la tâche qui est exécutée automatiquement par le module setup au début d'un play ?
-

Résultats

Référez-vous aux pages suivantes pour contrôler vos réponses. Pour chacune de vos réponses, compétez un point.

Nombre de points : /8

Votre score minimum doit être : 6/8

Réponses

- 1 Dans quel langage peut être écrit un fichier playbook ?

Les playbooks de Ansible utilisent le langage YAML parce qu'il est plus facile à lire et à écrire pour les humains que d'autres formats de données courants tels que XML ou JSON.

- 2 Dans quel ordre sont exécutés les plays et les tâches dans un playbook ?

Les plays et les tâches sont exécutés dans l'ordre dans lequel ils sont écrits au sein du fichier playbook.

- 3 Comment pouvez-vous vérifier la syntaxe du fichier site_fr.yml ?

Pour vérifier la syntaxe du fichier site_fr.yml, nous disposons de deux outils. Le premier est ansible-playbook :

ansible-playbook --syntax-check site_fr.yml

Le second est yamllint :

yamllint site_fr.yml