

Ansible

Ansible???????????????????????????????????????????????????????????????????? Ansible ???
Michael DeHaan ????? 2015 ??Red Hat?????????????? Unix???Microsoft Windows ?????

- Ansible GUI
- Playbooks
- Automation Tools

Ansible GUI

??

Ansible???????????????????????????????????????????????????????????????????? Ansible ???
Michael DeHaan ?????? 2015 ??Red Hat???????????????? Unix???Microsoft Windows ?????

Links

- Ansible Community
 - [AWX for Docker](#)
 - [Documentation](#)
- [How to Test Ansible Roles with Molecule and Docker](#)
- Red Hat Ansible Automation Platform
 - [Download](#)
 - [Trial License](#)
 - [Ansible Lightspeed with Watson Code Assistant](#)
?? IBM Watson Code Assistant ??? playbooks
 - [Setting up VSCode and Ansible Lightspeed on Ubuntu 22.04](#)
- [Ansible Galaxy](#)

Commands

```
# Look up the plugins
# Usage:
# motd_value: "{ { lookup('file', '/etc/motd') } }"
ansible-doc -l -t lookup
```

Ansible Semaphore

Links

- <https://www.ansible-semaphore.com/>
- Github
- Docs
- [Video] [This web UI for Ansible is so damn useful!](#)

Install with Docker

Create the directory

```
mkdir playbooks
mkdir config
chown 1001:1001 config
```

docker-compose.yml:

```
---
volumes:
  semaphore-mysql:
    driver: local
services:
  mysql:
    image: mysql:8.0
    hostname: mysql
    volumes:
      - semaphore-mysql:/var/lib/mysql
    environment:
      - MYSQL_RANDOM_ROOT_PASSWORD=yes
      - MYSQL_DATABASE=semaphore
      - MYSQL_USER=semaphore
      - MYSQL_PASSWORD=secret-password # change!
    restart: unless-stopped
  semaphore:
    container_name: ansiblesemaphore
    image: semaphoreui/semaphore:v2.8.90
    user: 1001:1001 # change if needed
    ports:
      - 3000:3000
    environment:
      - SEMAPHORE_DB_USER=semaphore
      - SEMAPHORE_DB_PASS=secret-password # change!
      - SEMAPHORE_DB_HOST=mysql
      - SEMAPHORE_DB_PORT=3306
      - SEMAPHORE_DB_DIALECT=mysql
      - SEMAPHORE_DB=semaphore
      - SEMAPHORE_ADMIN_PASSWORD=secret-admin-password # change!
```

```
- SEMAPHORE_ADMIN_NAME=admin
- SEMAPHORE_ADMIN_EMAIL=admin@localhost
- SEMAPHORE_ADMIN=admin
- SEMAPHORE_ACCESS_KEY_ENCRYPTION= # add to your access key encryption !
- ANSIBLE_HOST_KEY_CHECKING=false # (optional) change to true if you want to enable host key checking

volumes:
- ./inventory:/inventory:ro
- ./authorized-keys:/authorized-keys:ro
- ./config:/etc/semaphore:rw
- ./playbooks:/playbooks:ro

restart: unless-stopped

depends_on:
- mysql
```

You must specify following confidential variables:

- `MYSQL_PASSWORD` and `SEMAPHORE_DB_PASS` — password for the MySQL user.
- `SEMAPHORE_ADMIN_PASSWORD` — password for the Semaphore's admin user.
- `SEMAPHORE_ACCESS_KEY_ENCRYPTION` — key for encrypting access keys in database. It must be generated by using the following command: `head -c32 /dev/urandom | base64`.

Get Started

1. Create New Project
2. New Keys:
 - Name1: None
 - Type1: None
 - -----
 - Name2: ssh_alang
 - Type2: SSH Key
 - Username2: alang
 - Private key2: <Key-String>
 - -----
 - Name3: sudo_alang
 - Type3: Login with password
 - Login3: alang
 - Password3: <password>
3. New Repository:
 - Name: Local
 - Path: /playbooks
 - Access Key: None

AWX

RedHat 8.7

?????

```
dnf update
reboot
dnf install ansible-core openssl-libs
dnf group install "Development Tools"
dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
dnf install python39-setuptools_scm
```

AWX

```
git clone -b 22.3.0 https://github.com/ansible/awx.git
cd awx

# [REDACTED]
#vi tools/docker-compose/inventory

make docker-compose-build
cp Makefile{.,.orig}
sed -i 's/^\(DOCKER_COMPOSE ?=\).*\1 docker compose/' Makefile
make docker-compose
```

????????????????????????????????????

AWX Web UI: <https://server.ip.adress:8043/>

??????????

```
<% if (process.env.NODE_ENV === 'production') { %> <% } %> <% if
(process.env.NODE_ENV === 'production') { %> <% } else { %> <% } %>
<% if (process.env.NODE_ENV === 'production') { %>
```

Clean and build the UI

```
docker exec tools_awx_1 make clean-ui ui-devel
```

??????????

Creating an optimized production build...
Browserslist: caniuse-lite is outdated. Please run:
npx update-browserslist-db@latest
Why you should do it regularly: <https://github.com/browserslist/update-db#readme>

Ctrl + C ??

```
docker exec -it tools_awx_1 bash
> cd /awx_devel/awx/ui
> npx update-browserslist-db@latest
> exit
```

?????

```
docker exec tools_awx_1 make clean-ui ui-devel
```

????????????????

The project was built assuming it is hosted at ./.
You can control this with the homepage field in your package.json.

The build folder is ready to be deployed.

Find out more about deployment here:

<https://cra.link/deployment>

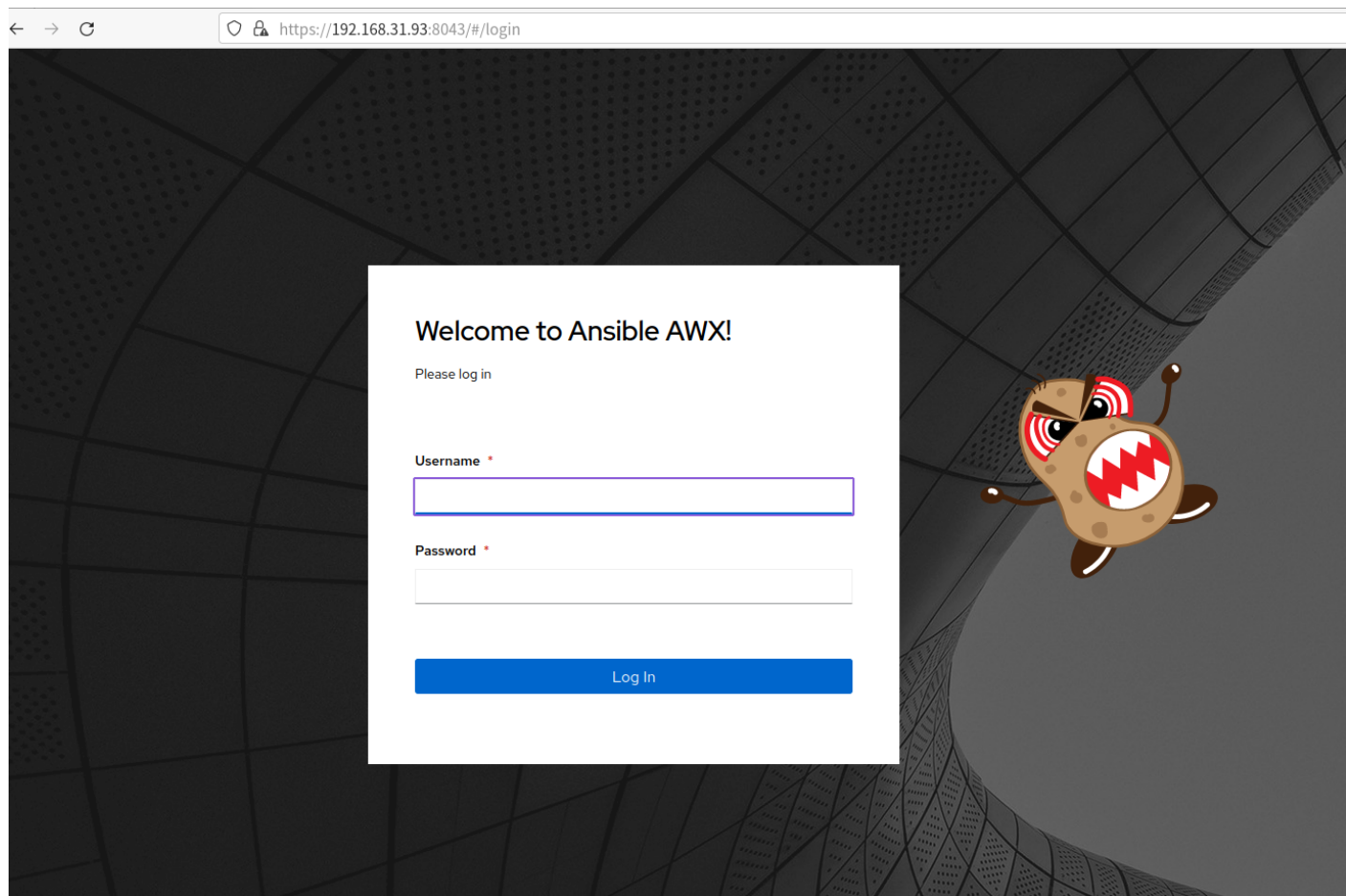
```
touch awx/ui/.ui-built
make[1]: Leaving directory '/awx_devel'
```

?? AWX ?????

? Ctrl + C ?? container????

```
make docker-compose
```

????? https://server.ip.adress:8043/



???admin ?????{? log ?}

Q & A

No match for argument: rsyslog-8.2102.0-106.el9

Solution:

```
cp tools/ansible/roles/dockerfile/templates/Dockerfile.j2{,.orig}
sed -i 's/rsyslog-8.[0-9a-z\.-]*rsyslog/g' tools/ansible/roles/dockerfile/templates/Dockerfile.j2
```

AWX Commands

?? AWX

```
# 1111
cd awx-repo/
make docker-compose

# 1111
make docker-compose COMPOSE_UP_OPTS=-d
```

?? AWX

```
docker stop tools_awx_1 tools_postgres_1 tools_redis_1
```

Create an admin user

```
docker exec -ti tools_awx_1 awx-manage createsuperuser
```


Playbooks

- [Github] [Ansible for DevOps Examples](#)
- [AI] [Welcome to the Ansible Lightspeed with IBM Watson Code Assistant Technical Preview | Ansible Collaborative](#)
- [AI] [Red Hat Ansible Lightspeed | Red Hat Developer](#)

Include a variables file

```
---  
  
- hosts: all  
  become: true  
  
  vars_files:  
    - vars.yml
```

vars.yml

```
---  
  
download_dir: /tmp  
solr_dir: /opt/solr  
solr_version: 8.6.0  
solr_checksum: sha512:6b0d618069e37215f305d9a61a3e65be2b9cfc32a3689ea6a25\  
be2f220b1ecc96a644ecc31c81e335a2dfa0bc8b7d0f2881ca192c36fd435cdd832fd7309a9ddb
```

Installs Apache on a RHEL/CentOS server

```
---  
  
- hosts: all  
  become: yes  
  
  tasks:  
    - name: Install Apache.  
      dnf:  
        name:  
          - httpd
```

```
- httpd-devel
state: present

- name: Copy configuration files.
copy:
  src: "{{ item.src }}"
  dest: "{{ item.dest }}"
  owner: root
  group: root
  mode: 0644
with_items:
  - src: httpd.conf
    dest: /etc/httpd/conf/httpd.conf
  - src: httpd-vhosts.conf
    dest: /etc/httpd/conf/httpd-vhosts.conf

- name: Make sure Apache is started now and at boot.
service:
  name: httpd
  state: started
  enabled: yes
```

Deploy Node.js app

```
---
- hosts: all
  become: yes

  vars:
    node_apps_location: /usr/local/opt/node

  tasks:
    - name: Install EPEL repo.
      dnf: name=epel-release state=present
    - name: Import Remi GPG key.
      rpm_key:
        key: "https://rpms.remirepo.net/RPM-GPG-KEY-remi2018"
        state: present
```

```

- name: Install Remi repo.
  dnf:
    name: "https://rpms.remirepo.net/enterprise/remi-release-8.rpm"
    state: present

- name: Ensure firewalld is stopped (since this is for testing).
  service: name=firewalld state=stopped

- name: Install Node.js and npm.
  dnf: name=npm state=present enablerepo=epel

- name: Install Forever (to run our Node.js app).
  npm: name=forever global=yes state=present

```

□

```

- name: Ensure Node.js app folder exists.
  file: "path={{ node_apps_location }} state=directory"

- name: Copy example Node.js app to server.
  copy: "src=app dest={{ node_apps_location }}"

- name: Install app dependencies defined in package.json.
  npm: path={{ node_apps_location }}/app

- name: Check list of running Node.js apps.
  command: /usr/local/bin/forever list
  register: forever_list
  changed_when: false

- name: Start example Node.js app.
  command: "/usr/local/bin/forever start {{ node_apps_location }}/app/app.js"
  when: "forever_list.stdout.find(node_apps_location + '/app/app.js') == -1"

```

Basic LAMP server setup

```

tasks:
- name: Get software for apt repository management.
  apt:
    state: present
    name:
      - python3-apt

```

- python3-pycurl

- name: Add Ondrej repository for later versions of PHP.

apt_repository: repo='ppa:ondrej/php' update_cache=yes

- name: "Install Apache, MySQL, PHP, and other dependencies."

apt:

state: present

name:

- - acl

- git

- curl

- unzip

- sendmail

- apache2

- php8.2-common

- php8.2-cli

- php8.2-dev

- php8.2-gd

- php8.2-curl

- php8.2-opcache

- php8.2-xml

- php8.2-mbstring

- php8.2-pdo

- php8.2-mysql

- php8.2-apcu

- libpcre3-dev

- libapache2-mod-php8.2

- python3-mysqldb

- mysql-server

-

- name: Disable the firewall (since this is for local dev only).

service: name=ufw state=stopped

- name: "Start Apache, MySQL, and PHP."

service: "name={{ item }} state=started enabled=yes"

with_items:

- apache2

- mysql

Configure Apache

```
- name: Enable Apache rewrite module (required for Drupal).
  apache2_module: name=rewrite state=present
  notify: restart apache

- name: Add Apache virtualhost for Drupal.
  template:
    src: "templates/drupal.test.conf.j2"
    dest: "/etc/apache2/sites-available/{{ domain }}.test.conf"
    owner: root
    group: root
    mode: 0644
  notify: restart apache

- name: Enable the Drupal site.
  command: >
    a2ensite {{ domain }}.test
    creates=/etc/apache2/sites-enabled/{{ domain }}.test.conf
  notify: restart apache

- name: Disable the default site.
  command: >
    a2dissite 000-default
    removes=/etc/apache2/sites-enabled/000-default.conf
  notify: restart apache
```

Template: drupal.test.conf.j2

```
<VirtualHost *:80>
  ServerAdmin webmaster@localhost
  ServerName {{ domain }}.test
  ServerAlias www.{{ domain }}.test
  DocumentRoot {{ drupal_core_path }}/web
  <Directory "{{ drupal_core_path }}/web">
    Options FollowSymLinks Indexes
    AllowOverride All
  </Directory>
</VirtualHost>
```

Configure PHP with *lineinfile*

```
- name: Adjust OpCache memory setting.  
lineinfile:  
  dest: "/etc/php/8.2/apache2/conf.d/10-opcache.ini"  
  regexp: "^opcache.memory_consumption"  
  line: "opcache.memory_consumption = 96"  
  state: present  
  notify: restart apache
```

Configure MySQL

```
- name: Create a MySQL database for Drupal.  
mysql_db: "db={{ domain }}" state=present"  
  
- name: Create a MySQL user for Drupal.  
mysql_user:  
  name: "{{ domain }}"  
  password: "1234"  
  priv: "{{ domain }}.*:ALL"  
  host: localhost  
  state: present
```

Install Composer with *get_url*

```
- name: Download Composer installer.  
get_url:  
  url: https://getcomposer.org/installer  
  dest: /tmp/composer-installer.php  
  mode: 0755  
  
- name: Run Composer installer.  
command: >  
  php composer-installer.php  
  chdir=/tmp  
  creates=/usr/local/bin/composer  
  
- name: Move Composer into globally-accessible location.  
command: >  
  mv /tmp/composer.phar /usr/local/bin/composer
```

```
creates=/usr/local/bin/composer
```

Create a Drupal project with Composer

- name: Ensure Drupal directory exists.

file:

path: "{{ drupal_core_path }}"

state: directory

owner: www-data

group: www-data

- name: Check if Drupal project already exists.

stat:

path: "{{ drupal_core_path }}/composer.json"

register: drupal_composer_json

- name: Create Drupal project.

composer:

command: create-project

arguments: drupal/recommended-project "{{ drupal_core_path }}"

working_dir: "{{ drupal_core_path }}"

no_dev: true

become_user: www-data

when: not drupal_composer_json.stat.exists

Decompress the tar file

- name: Download Solr.

get_url:

url: "https://archive.apache.org/dist/lucene/solr/\

{{ solr_version }}/solr-{{ solr_version }}.tgz"

dest: "{{ download_dir }}/solr-{{ solr_version }}.tgz"

checksum: "{{ solr_checksum }}"

- name: Expand Solr.

unarchive:

src: "{{ download_dir }}/solr-{{ solr_version }}.tgz"

dest: "{{ download_dir }}"

remote_src: true

creates: "{{ download_dir }}/solr-{{ solr_version }}/README.txt"

Automation Tools

With SSH

- [Fabric](#) is a high level Python (2.7, 3.4+) library designed to execute shell commands remotely over SSH, yielding useful Python objects in return. (?????????????)
 - [Fabric - Run Shell Commands Remotely Over SSH in Linux \(tecmint.com\)](#)
- sshpass (??? SHELL ????????)
 - [ssh | BookOSSLab](#)
 - [sshpass - Login to ssh server with a password using a shell script - nixCraft \(cyberciti.biz\)](#)