Ansible Minecraft Role

unknown

May 17, 2023

TABLE OF CONTENTS

This role installs Minecraft or Spigot and configures it to run under systemd or Supervisor. Its recommended to use the `systemd` process management

Note:

For all steps (Development,starting the Server and executing tests) the User must accept the Minecraft EULA, by own configured properly!
by example setting a envierment property like `export mc_accept_eula=true`

This Documentation should be describe how you use and develop this Ansible Role. You find a list of possible role configurations at *Role variables*.

CHAPTER

ONE

ANSIBLE-MINECRAFT

This is a fork from the https://github.com/devops-coop/ansible-minecraft/ Project, thanks for the basement!!!

chat on gitter

This role installs Minecraft or Spigot and configures it to run under systemd or Supervisor.

1.1 Features

- supports vanilla Minecraft and Spigot
- supports Debian >9, Ubuntu 14.04, Ubuntu 16.04, Ubuntu 18.04, CentOS 7, Fedora 29
- safely stops the server using stop when running under **systemd**
- uses Docker and Molecule to run integration tests
- manages user ACLs
- manages Bukkit/Spigot Plugins
- manages `server.properties`
- hooks: include arbitrary tasks at specific stages during execution

1.1.1 Out of Role Scop

- executing backups and recovery
- healthy checks like Minecraft-Region-Fixer
- handle utility services like filebeat or prometheus
- install additional Tools like rcon-cli.

All of this is needed but not a part of this role!, *you will find examples at*nolte/minecraft-infrastructure.

1.2 Usage

By default this role will be install a Vanilla Minecraft Server.

1.2.1 Install

```
ansible-galaxy install nolte.minecraft
```

or add this to your `requirements.yml`

```
- name: nolte.minecraft
  version: v5.0.12.dev
```

and execute `ansible-galaxy install -r requirements.yml`

1.2.2 Use

```
- hosts: minecraft
  roles:
     - { role: nolte.minecraft, minecraft_whitelist: ["jeb_", "dinnerbone"]}
```

1.3 Requirements

- Python 3.x on the Ansible control machine to generate user ACLs
- Ansible 2.7.0+ on the control machine to fetch the Minecraft version

1.4 Contributing

The best way to contribute is to use this role to deploy your own Minecraft server! We really appreciate bug

CHAPTER

TWO

ADVANCED USAGE

2.1 Configure the Role

This role should fix two Problems, firstly *Configure the Server* and secondly *Install Plugins*.

2.1.1 Configure the Server

This Role will be install by default a vanilla server to the configured *Server Directory*. You will find a full list of configuration attributes on *Role variables*.

Example

```
- hosts: minecraft
  roles:
     - { role: nolte.minecraft, minecraft_whitelist: ["jeb_", "dinnerbone"]}
```

2.1.2 Install Plugins

The plugins will installed to the Configured *Plugins Location* into a Release subfolder like `plugins/releases/{pluginsets}/*.jar` and finaly link to `plugins/shared`.

The `plugins/shared` Directory will be linked to `server/shared/plugins` all Plugin Runtime-data of your server will be stored under `plugins/shared`, see *FileSystem Structure*.

Listing 1: Example Plugin Source Config file

```
      minecraft_plugins_set_version: "minimal"
      minecraft_plugin_sets:
        minimal:
          vault:
            src: https://media.forgecdn.net/files/2615/750/Vault.jar
          permissionsEx:
            src: https://media.forgecdn.net/files/909/154/PermissionsEx-1.23.4.jar
            dest: PermissionsEx.jar
            config:
              - src: "{{ playbook_dir }}/molecule/resources/playbooks/templates/config_
↪permissionex.yml.j2"
                dest: PermissionsEx/config.yml
```

(continues on next page)

(continued from previous page)

```
        multiverseCore:
          src: https://ci.onarandombox.com/job/Multiverse-Core/lastSuccessfulBuild
          dest: Multiverse-Core.jar
          type: "jenkins_latest"
          jenkins_artefact_path: "/artifact/target"
          force: true
          validate_certs: false
        tne:
          src: https://github.com/TheNewEconomy/TNE-Bukkit/releases/download/0.1.1.8/
↪TNE-0.1.1.8.zip
          type: "archive"
```

Configure Plugin Download Source

Directly Download a `*.jar` from a Webserver, like `media.forgecdn.net`.

`type` (*optional*) default direct jar

> `"jenkins_latest"` used for load the latest successful build.
>
> `"archive"` used for load and unpack some Archive from remote.

`src` The Download Source from the Plugin.

`dest` (*optional*) The local jar name, like `PermissionsEx.jar`

`force` (*optional*) overwrite allways existing plugins, (default: `false`).

`validate_certs` (*optional*) If `false`, SSL certificates will not be validated, look (Ansible Doc, validate_certs) (default: `true`).

`jenkins_artefact_path` (*optional*)

> system group Minecraft runs as (default: `/artifact/target`)
> only usable with `type: "jenkins_latest"`

`config` (*optional*)

> To automatical configure a plugin create a `Jinja` templatefile at your Playbook `templates` folder, and add a `config:` entry.
> The `dest:` path is relative to the `plugins/shared` folder.

Listing 2: You can set a list of dict`s like:

```
...
config:
  - src: "{{ playbook_dir }}/templates/config_permissionex.yml.j2"
    dest: PermissionsEx/config.yml
...
```

2.2 Role variables

The following variable defaults are defined in `defaults/main.yml`.

minecraft_version Minecraft version to install (default: `latest`)

Examples:

```
minecraft_version: latest
minecraft_version: 1.10
minecraft_version: 1.9.1
minecraft_version: 16w21a
```

minecraft_eula_accept accept the Minecraft eula License, must accepted by the Role User (default: `false`)

minecraft_url Minecraft download URL (default: `https://s3.amazonaws.com/Minecraft.Download/versions`)

minecraft_user system user Minecraft runs as (default: `{{ minecraft_server }}`)

minecraft_group system group Minecraft runs as (default: `{{ minecraft_server }}`)

minecraft_basedir directory base variable for the Minecraft installation (default: `/opt/minecraft`)

minecraft_home directory to install Minecraft Server to (default: `{{minecraft_basedir}}/server`)

minecraft_plugins directory to install Minecraft Plugins to (default: `{{minecraft_basedir}}/plugins`)

minecraft_max_memory Java max memory (`-Xmx`) to allocate (default: `1024M`)

minecraft_initial_memory Java initial memory (`-Xms`) to allocate (default: `1024M`)

minecraft_service_name systemd service name or Supervisor program name (default: `minecraft`)

minecraft_supervisor_name DEPRECATED: Supervisor program name (default: `{{ minecraft_service_name }}`)

minecraft_whitelist list of Minecraft usernames to whitelist (default: `[]`)

minecraft_ops list of Minecraft usernames to make server ops (default: `[]`)

minecraft_banned_players list of Minecraft usernames to ban (default: `[]`)

minecraft_banned_ips list of IP addresses to ban (default: `[]`)

minecraft_server_properties dictionary of server.properties entries (e.g. `server-port: 25565`) to set (default: `{}`)

minecraft_server choose between `minecraft` or `spigot` (default: `minecraft`)

minecraft_server_java_ops additional java ops like remote debug `-Xdebug -Xnoagent -Djava.compiler=NONE -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=5005` (default: *none*)

2.2.1 Logging

`minecraft_external_log_conf` (***optional***) **type Dict** handle a external Log4j2 Config used RollingRandomAccessFileAppender, controlling LogRotate, Maximal LogFile Size, and maximum keeped logs.

Examples:

```
minecraft_external_log_conf:
  conf_file: log4j2.xml
  template: log4j2.xml.j2
  fileName: /var/log/minecraft/server.log
  filePattern: /var/log/minecraft/server_%d{yyyy-MM-dd}.log.gz
  rollover: 5
  sizeBased: 10MB
```

2.2.2 Hooks and run stages

ansible-minecraft organizes execution into a number of run stages:

`setup`

- install prerequisites (e.g., Java)
- create Minecraft user and group

`download`

- fetch the latest version of from the launcher API
- download Minecraft

`install`

- symlink version to `minecraft_server.jar`
- agree to EULA

`acl`

- configure server ACLs (whitelist, banned players, etc.)

`configure`

- set `server.properties`

`start`

- (re)start server

You can execute custom tasks before or after specific stages. Simply specify a task include file using the relevant role variable:

```
- hosts: minecraft
  roles:
    - role: devops-coop.minecraft
      minecraft_hook_before_start: "{{ playbook_dir }}/download-world-from-s3.yml"
```

The available hooks are:

`minecraft_hook_before_setup` run before `setup` tasks

`minecraft_hook_after_setup` run after `setup` tasks

minecraft_hook_before_download run before download tasks

minecraft_hook_after_download run after download tasks

minecraft_hook_before_install run before install tasks

minecraft_hook_after_install run after install tasks

minecraft_hook_before_start run before start tasks

minecraft_hook_after_start run after start tasks

2.3 Maintenance

2.3.1 Useful commands

command	description
`sudo service spigot restart`	server restart
`tail -f /opt/minecraft/server/shared/logs/latest.log`	show current logs

2.3.2 Version updates

TBD

2.3.3 FileSystem Structure

```
[vagrant@localhost minecraft]$ tree -L 4
.
├── plugins
│   ├── current -> /opt/minecraft/plugins/releases/minimal
│   ├── releases
│   │   └── minimal
│   │       ├── Multiverse.jar
│   │       ├── PermissionsEx.jar
│   │       ├── report-jenkinsbuild.yml
│   │       ├── report-permissionsEx.yml
│   │       ├── report-vault.yml
│   │       ├── TNE.jar
│   │       └── Vault.jar
│   └── shared
│       ├── Multiverse.jar -> /opt/minecraft/plugins/releases/minimal/Multiverse.jar
│       ├── PermissionsEx
│       │   └── config.yml
│       ├── PermissionsEx.jar -> /opt/minecraft/plugins/releases/minimal/PermissionsEx.
↪jar
│       ├── TheNewEconomy
│       ├── TNE.jar -> /opt/minecraft/plugins/releases/minimal/TNE.jar
│       └── Vault.jar -> /opt/minecraft/plugins/releases/minimal/Vault.jar
└── server
    ├── current -> /opt/minecraft/server/releases/1.13.2
    ├── releases
```

(continues on next page)

(continued from previous page)

```
    │   └── 1.13.2
    │       ├── ...
    │       ├── spigot-1.13.2.jar
    │       └── work
    └── shared
        ├── plugins -> /opt/minecraft/plugins/shared
        ├── server.properties
        └── spigot.jar -> /opt/minecraft/server/current/spigot-1.13.2.jar
```

For First usage of the Role crate a Ansible Playbook Project, with a structure like this:

```
.
├── inventories
│   ├── prod
│   │   └── hosts.yml
│   └── test
│       └── hosts.yml
├── pluginlist.yml
├── provision-minecraft-master.yml
├── requirements.yml
└── Vagrantfile
```

(The `Vagrantfile` *is only for a local TestEnv needet)*

List this `role` under the `requirements.yml` file.

```
...
- name: nolte.minecraft
  version: 5.0.12.dev
...
```

> **Warning:** Please when you host the Minecraft Server at the internet configure somethink like `firewalld` for a minimal portection, and don`t publish the `rcon.port` to the public space!!

CHAPTER

THREE

DEVELOPMENT

This section shoud be describe how the development process *(coding, testing, releasing and publishing)* works.

3.1 Building

As build script we use Tox, so it`s easy to execute the different kind of build commands like, generate docs or execute tests.

```
tox -e spigot
```

Possible Tox Envs

env	Description
`docs`	generates the sphinx documentation page (generated to `.tox/docs/tmp/html/`)
`default`	Execute an Molecule tests for the classic vanilla Minecraft server (Tested `CentOS7`, `Ubuntu1604`, `Ubuntu18`, `DebianJessie`)
`spigot`	Execute the Molecule tests for a spigot server.

3.1.1 Versioning

This project follows semantic versioning.

In the context of semantic versioning, consider the role contract to be defined by the role variables.

- Changes that require user intervention will increase the **major** version. This includes changing the default value of a role variable.
- Changes that do not require user intervention, but add backwards-compatible features, will increase the **minor** version.
- Bug fixes will increase the **patch** version.

Handling Version

For handle the version number in the different files we use the bumpversion tool.
The updateable files are listed at `.bumpversion.cfg` placed in the project root directory.

Update project minor version

Call bumpversion on the commandline like:

```
bumpversion minor
```

for update the **minor** version of this project.

3.1.2 Releasing

3.2 Testing

Testing can be done using the provided Vagrantfile or by installing Docker and use Molecule locally.

For execute the molecule test you can use the Docker Image described at Molecule page.

```
docker run --rm -it \
    -v $(pwd):/tmp/$(basename "${PWD}"):ro \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -e mc_accept_eula=${mc_accept_eula} \
    -w /tmp/$(basename "${PWD}") \
    retr0h/molecule:latest \
    sudo molecule test --all
```

after execute drink a pot of tee, coffee or some beer, all molecule scenarios will be run more than 40 minute

3.2.1 Testing with Molecule

The Tests are impemented with Molecule

```
molecule test -s spigot
```

Molecule Tips

For the development and debugging it is easyer and faster to execute the Molecule sequenzes step by step.
First you must start the container with `molecule create`, after the containers started, you can execute the Role/Playbook `molecule converge`.
Now, when all the steps are finished, you can execute the Integration Tests with `molecule verify`

Note: For Debugging the role take a look into the container with `docker exec -t -i centos7 /bin/bash`

3.2.2 Testing with Vagrant

This role includes a Vagrantfile used for Exploratory testing.
If you want to use this vagrant machine follow this steps:

1. Install Vagrant and VirtualBox.
1.1. Accept the Minecraft EULA with setting a Environment Property like: `export mc_accept_eula=true`
2. Run `vagrant up` from the same directory as the Vagrantfile in this repository.

Note: Now, you can start the game and conntecting again our server e.g. `localhost:25565` and test the changes.

3. for manual lookups you can connect over SSH into the VM with: `vagrant ssh`

Note: If the Vagrant box allways exists, you can reexecute the Playbook with `vagrant rsync && vagrant provision`

3.3 Used Public Services

For the Development we use a set of public services:

GitHub for SoureCode and issue Tracking.

TravisCI for simple tests execution an Ansible galaxy update notification.

CircleCI for complex workflows, like releasing and publishing documentation.

3.4 Branch Modell

As Branchmodel we use a mix of Gitflow and pull-requests. Gitflow is used for the Release Process, the `master` branch present the latest Published Release. PullRequests are used for integrate external changes and `feature` branches into the `develop` branch.

The `develop` branch contains the latest unrelesed version from the role, mostly stable ;)
New features will be develop in feature branches like `feature/integrate-cuberite`, it`s not recommended to use this on PRODUCTION!!!.
The `master` present the latest published release.

For the Continuous Integration we use *GitHub Workflows* as service.

CHAPTER FOUR

STRUCTURE

A finished installation can be looks like:

```
[vagrant@localhost minecraft]$ tree -L 4
.
├── plugins
│   ├── current -> /opt/minecraft/plugins/releases/minimal
│   ├── releases
│   │   └── minimal
│   │       ├── Multiverse.jar
│   │       ├── PermissionsEx.jar
│   │       ├── report-jenkinsbuild.yml
│   │       ├── report-permissionsEx.yml
│   │       ├── report-vault.yml
│   │       ├── TNE.jar
│   │       └── Vault.jar
│   └── shared
│       ├── Multiverse.jar -> /opt/minecraft/plugins/releases/minimal/Multiverse.jar
│       ├── PermissionsEx
│       │   └── config.yml
│       ├── PermissionsEx.jar -> /opt/minecraft/plugins/releases/minimal/PermissionsEx.
↪jar
│       ├── TheNewEconomy
│       ├── TNE.jar -> /opt/minecraft/plugins/releases/minimal/TNE.jar
│       └── Vault.jar -> /opt/minecraft/plugins/releases/minimal/Vault.jar
└── server
    ├── current -> /opt/minecraft/server/releases/1.13.2
    ├── releases
    │   └── 1.13.2
    │       ├── ...
    │       ├── spigot-1.13.2.jar
    │       └── work
    └── shared
        ├── plugins -> /opt/minecraft/plugins/shared
        ├── server.properties
        └── spigot.jar -> /opt/minecraft/server/current/spigot-1.13.2.jar
```