# Managing Facts

# What are Facts?

Ansible facts are variables that are automatically discovered by Ansible on a managed host.

# Managing Facts

Facts contain host-specific information that can be used just like regular variables in **plays**, **conditionals**, **loops**, or any other statement that depends on a value collected from a managed host.

# Some of the facts gathered for a managed host must include:

- The host name
- The kernel version
- The network interfaces
- The IP addresses
- The version of the operating system
- Various environment variables
- The number of CPUs
- The available or free memory
- The available disk space

# Use case of Facts

Facts are convenient way to retrieve the state of managed host and to determine what action to take on that state. For Example

- A Server can be restarted by a conditional task which is run based on a fact containing the managed hosts current kernel version.
- The MySQL configurationfile can be customized depending on the available memory reported by a fact.
- The IPv4 address used in a configuration file can be set based on the value of a fact.

# Example of facts

- Normally, every play runs the setup module automatically before the first task in order to gather facts. By default, you do not need to have a task to run setup in your play. It is normally run automatically for you.
- One way to see what facts are gathered for your managed hosts is to run a short playbook that gathers facts and uses the debug module to print the value of the ansible_facts variable.

```
- name: Fact dump
 hosts: all
 tasks:
   - name: Print all facts
     debug:
       var: ansible_facts
```

- When you run the playbook, the facts are displayed in the job output:

```
[root@ansimaster ~]# ansible-playbook facts.yml

PLAY [Fact dump] ***************************************************************

TASK [Gathering Facts] *********************************************************
ok: [demo1.example.com]

TASK [Print all facts] *********************************************************
ok: [demo1.example.com] => {
    "ansible_facts": {
        "all_ipv4_addresses": [
            "172.25.250.10"
        ],
        "all_ipv6_addresses": [
            "fe80::5054:ff:fe00:fa0a"
        ],
        "ansible_local": {},
        "apparmor": {
            "status": "disabled"
        },
        "architecture": "x86_64",
        "bios_date": "01/01/2011",
        "bios_version": "0.5.1",
        "cmdline": {
            "BOOT_IMAGE": "/boot/vmlinuz-3.10.0-327.el7.x86_64",
            "LANG": "en_US.UTF-8",
            "console": "ttyS0,115200n8",
            "crashkernel": "auto",
            "net.ifnames": "0",
            "no_timer_check": true, "ro": true,
            "root": "UUID=2460ab6e-e869-4011-acae-31b2e8c05a3b"
         },
...output omitted...
```

# Example of Ansible Facts:

| Fact | Variable |
|---|---|
| short Host Name | ansible_facts[ ‘hostname’ ] |
| Fully qualified domain name | ansible_facts[ ‘fqdn’ ] |
| Main IPv4 address (based on routing) | ansible_facts['default_ipv4']['address'] |
| List of the names of all network interfaces | ansible_facts['interfaces'] |
| Size of the /dev/vda1 disk partition | ansible_facts['devices']['vda']['partitions']['vda1']['size'] |
| List of DNS servers | ansible_facts['dns']['nameservers'] |
| Version of the currently running kernel | ansible_facts['kernel'] |

# Ansible Facts Injected as Variables

- Before Ansible 2.5, facts were injected as individual variables prefixed with the string ansible_ instead of being part of the **ansible_facts** variable.
- For example, the ansible_facts['distribution'] fact would have been called ansible_distribution.
- We can use an ad hoc command to run the setup module to print the value of all facts in this form. In the following example, an ad hoc command is used to run the setup module on the managed host demo1.example.com:

```
[root@ansimaster ~]# ansible demo1.example.com -m setup
demo1.example.com | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.25.250.10"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::5054:ff:fe00:fa0a"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "01/01/2011",
        "ansible_bios_version": "0.5.1",
        "ansible_cmdline": {
             "BOOT_IMAGE": "/boot/vmlinuz-3.10.0-327.el7.x86_64",
             "LANG": "en_US.UTF-8",
             "console": "ttyS0,115200n8",
             "crashkernel": "auto",
             "net.ifnames": "0",
             "no_timer_check": true,
             "ro": true,
             "root": "UUID=2460ab6e-e869-4011-acae-31b2e8c05a3b"
        }
...output omitted...
```

The following table compares the old and new fact names.

- Comparison of Selected Ansible Fact Names

| ansible_facts form Old fact variable form | ansible_facts form Old fact variable form |
|---|---|
| ansible_facts['hostname'] ansible_hostname | ansible_facts['hostname'] ansible_hostname |
| ansible_facts['fqdn'] ansible_fqdn | ansible_facts['fqdn'] ansible_fqdn |
| ansible_facts['default_ipv4'] | ansible_facts['default_ipv4'] |
| ansible_facts['interfaces'] ansible_interfaces | ansible_facts['interfaces'] ansible_interfaces |
| ansible_facts['devices']['vda'] ['partitions']['vda1']['size'] | ansible_devices['vda'] ['partitions']['vda1']['size'] |
| ansible_facts['dns'] ['nameservers'] | ansible_dns['nameservers'] |
| ansible_facts['kernel'] | ansible_kernel |

# Turning Off Fact Gathering

If you do not want to gather facts for your play.

There are a couple of reasons why this might be the case.

- It might be that you are not using any facts and want to speed up the play or reduce load caused by the play on the managed hosts.
- It might be that the managed hosts cannot run the setup module for some reason, or need to install some prerequisite software before gathering facts.

To disable fact gathering for a play, set the gather_facts keyword to no:

```
---
- name: This play gathers no facts automatically
  hosts: large_farm
  ga9ther_facts: no
```

Even if gather_facts: no is set for a play, you can manually gather facts at any time by running a task that uses the setup module:

```
tasks:
  - name: Manually gather facts
    setup:
...output omitted...
```

# Creating Custom Facts

- Administrators can create custom facts which are stored locally on each managed host.
- These facts are integrated into the list of standard facts gathered by the setup module when it runs on the managed host.
- These allow the managed host to provide arbitrary variables to Ansible which can be used to adjust the behavior of plays.
- Custom facts can be defined in a static file, formatted as an INI file or using JSON. They can also be executable scripts which generate JSON output.
- Custom facts allow administrators to define certain values for managed hosts, which plays might use to populate configuration files or conditionally run tasks.
- Dynamic custom facts allow the values for these facts, or even which facts are provided, to be determined programmatically when the play is run.

- By default, the setup module loads custom facts from files and scripts in each managed host's /etc/ansible/facts.d directory.
- The name of each file or script must end in .fact in order to be used. Dynamic custom fact scripts must output JSON-formatted facts and must be executable.
- This is an example of a static custom facts file written in INI format. An INI-formatted custom facts file contains a top level defined by a section, followed by the key-value pairs of the facts to define:

```
[packages]
web_package = httpd
db_package = mariadb-server

[users]
user1 = joe
user2 = jane
```

# Creating Custom Facts

- The same facts could be provided in JSON format. The following JSON facts are equivalent to the facts specified by the INI format in the preceding example. The JSON data could be stored in a static text file or printed to standard output by an executable script:

```
{
"packages": {
  "web_package": "httpd",
  "db_package": "mariadb-server"
 },
 "users": {
  "user1": "joe",
  "user2": "jane"
 }
}
```

- Custom facts are stored by the setup module in the ansible_facts['ansible_local']variable. Facts are organized based on the name of the file that defined them.

- For example, assume that the preceding custom facts are produced by a file saved as /etc/ ansible/facts.d/custom.fact on the managed host. In that case, the value of ansible_facts['ansible_local']['custom']['users']['user1'] is joe.
- You can inspect the structure of your custom facts by running the setup module on the managed hosts with an ad hoc command.

- Custom facts can be used the same way as default facts in playbooks:

```
[root@ansimaster ~]# cat playbook.yml
---
- hosts: all
  tasks:
  - name: Prints various Ansible facts
    debug:
      msg: >
           The package to install on {{ ansible_facts['fqdn'] }}
           is {{ ansible_facts['ansible_local']['custom']['packages']
['web_package'] }}

[user@demo ~]$ ansible-playbook playbook.yml
PLAY ***************************************************************

TASK [Gathering Facts] *********************************************
ok: [demo1.example.com]

TASK [Prints various Ansible facts] ********************************
ok: [demo1.example.com] => {
    "msg": "The package to install on demo1.example.com is httpd"
}

PLAY RECAP *********************************************************
demo1.example.com       : ok=2   changed=0       unreachable=0    failed=0
```