链滴

ansible 的 /etc/ansible/hosts 主机清单目录

作者：SmiteLi
原文链接：https://ld246.com/article/1538270225393
来源网站：链滴

一、概览

Ansible works against multiple systems in your infrastructure at the same time. It does this by selecting portions of systems listed in Ansible’ s inventory, which defaults to being saved in t e location /etc/ansible/hosts. You can specify a different inventory file using the -i option on he command line.

Not only is this inventory configurable, but you can also use multiple inventory files at the sa e time and pull inventory from dynamic or cloud sources or different formats (YAML, ini, etc), s described in Working With Dynamic Inventory. Introduced in version 2.4, Ansible has invent ry plugins to make this flexible and customizable.

二、配置文件详解

2.1 Hosts and Groups

The inventory file can be in one of many formats, depending on the inventory plugins you ha e. For this example, the format for /etc/ansible/hosts is an INI-like (one of Ansible’ s defaults) and looks like this:

```
mail.example.com

[webservers]
foo.example.com
bar.example.com

[dbservers]
one.example.com
two.example.com
```

原文链接：ansible 的 /etc/ansible/hosts 主机清单目录

```
three.example.com
```

The headings in brackets are group names, which are used in classifying systems and deciding what systems you are controlling at what times and for what purpose.

A YAML version would look like:

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```

It is ok to put systems in more than one group, for instance a server could be both a webserv r and a dbserver. If you do, note that variables will come from all of the groups they are a m mber of. Variable precedence is detailed in a later chapter.

If you have hosts that run on non-standard SSH ports you can put the port number after he hostname with a colon. Ports listed in your SSH config file won’t be used with the pa amiko connection but will be used with the openssh connection.

To make things explicit, it is suggested that you set them if things are not running on the def ult port:

```
badwolf.example.com:5309
```

Suppose you have just static IPs and want to set up some aliases that live in your host file, or ou are connecting through tunnels. You can also describe hosts via variables:

In INI:

```
jumper ansible_port=5555 ansible_host=192.0.2.50
```

In YAML:

```
...
  hosts:
    jumper:
      ansible_port: 5555
      ansible_host: 192.0.2.50
```

In the above example, trying to ansible against the host alias “jumper” (which may not even be a real hostname) will contact 192.0.2.50 on port 5555.

Note that this is using a feature of the inventory file to define some special variables. Generall speaking, this is not the best way to define variables that describe your system policy, but w ’ ll share suggestions on doing this later.

Note:

Values passed in the INI format using the key=value syntax are not interpreted as Python liter l structure (strings, numbers, tuples, lists, dicts, booleans, None), but as a string. For example ar=FALSE would create a string equal to ‘FALSE’. Do not rely on types set during definition, always make sure you specify type with a filter when needed when consuming the variable.

If you are adding a lot of hosts following similar patterns, you can do this rather than listing e ch hostname:

```
[webservers]
www[01:50].example.com
```

For numeric patterns, leading zeros can be included or removed, as desired. Ranges are inclus ve. You can also define alphabetic ranges:

```
[databases]
db-[a:f].example.com
```

You can also select the connection type and user on a per host basis:

```
[targets]
localhost              ansible_connection=local
other1.example.com     ansible_connection=ssh       ansible_user=mpdehaan
other2.example.com     ansible_connection=ssh       ansible_user=mdehaan
```

As mentioned above, setting these in the inventory file is only a shorthand, and we’ ll discuss how to store them in individual files in the ‘host_vars’ directory a bit later on.

2.2 Host Variables

As described above, it is easy to assign variables to hosts that will be used later in playbooks:

```
[atlanta]
host1 http_port=80 maxRequestsPerChild=808
host2 http_port=303 maxRequestsPerChild=909
```

2.3 Group Variables

Variables can also be applied to an entire group at once:

The INI way:

```
[atlanta]
host1
```

```
host2

[atlanta:vars]
ntp_server=ntp.atlanta.example.com
proxy=proxy.atlanta.example.com
```

The YAML version:

```
atlanta:
  hosts:
    host1:
    host2:
  vars:
    ntp_server: ntp.atlanta.example.com
    proxy: proxy.atlanta.example.com
```

Be aware that this is only a convenient way to apply variables to multiple hosts at once; even hough you can target hosts by group, **variables are always flattened to the host level** befor a play is executed.

2.4 Groups of Groups, and Group Variables

It is also possible to make groups of groups using the :children suffix in INI or the children: en ry in YAML. You can apply variables using :vars or vars::

```
[atlanta]
host1
host2

[raleigh]
host2
host3

[southeast:children]
atlanta
raleigh

[southeast:vars]
some_server=foo.southeast.example.com
halon_system_timeout=30
self_destruct_countdown=60
escape_pods=2

[usa:children]
southeast
northeast
southwest
northwest
```

yml version:

```
all:
  children:
    usa:
      children:
        southeast:
          children:
            atlanta:
              hosts:
                host1:
                host2:
            raleigh:
              hosts:
                host2:
                host3:
          vars:
            some_server: foo.southeast.example.com
            halon_system_timeout: 30
            self_destruct_countdown: 60
            escape_pods: 2
        northeast:
        northwest:
        southwest:
```

If you need to store lists or hash data, or prefer to keep host and group specific variables sepa ate from the inventory file, see the next section. Child groups have a couple of properties to ote:

- Any host that is member of a child group is automatically a member of the parent group.
- A child group’ s variables will have higher precedence (override) a parent group’ s vari bles.
- Groups can have multiple parents and children, but not circular relationships.
- Hosts can also be in multiple groups, but there will only be **one** instance of a host, me ging the data from the multiple groups.

2.5 Default groups

There are two default groups: all and ungrouped. all contains every host. ungrouped contains all hosts that don’ t have another group aside from all. Every host will always belong to at lea t 2 groups. Though all and ungrouped are always present, they can be implicit and not appear in group listings like group_names.

2.6 Splitting Out Host and Group Specific Data

The preferred practice in Ansible is to not store variables in the main inventory file.

In addition to storing variables directly in the inventory file, host and group variables can be s ored in individual files relative to the inventory file (not directory, it is always the file).

These variable files are in YAML format. Valid file extensions include ‘.yml’ , ‘.yaml’ , ‘.js n’ , or no file extension. See YAML Syntax if you are new to YAML.

Assuming the inventory file path is:

/etc/ansible/hosts

If the host is named ‘foosball’, and in groups ‘raleigh’ and ‘webservers’, variables in AML files at the following locations will be made available to the host:

/etc/ansible/group_vars/raleigh # can optionally end in '.yml', '.yaml', or '.json'
/etc/ansible/group_vars/webservers
/etc/ansible/host_vars/foosball

For instance, suppose you have hosts grouped by datacenter, and each datacenter uses some different servers. The data in the groupfile /etc/ansible/group_vars/raleig‘ for the ‘raleigh’ group might look like:

---

ntp_server: acme.example.org
database_server: storage.example.org

It is okay if these files do not exist, as this is an optional feature.

As an advanced use case, you can create directories named after your groups or hosts, and An ible will read all the files in these directories in lexicographical order. An example with the ‘ra eigh’ group:

/etc/ansible/group_vars/raleigh/db_settings
/etc/ansible/group_vars/raleigh/cluster_settings

All hosts that are in the ‘raleigh’ group will have the variables defined in these files availabl to them. This can be very useful to keep your variables organized when a single file starts to e too big, or when you want to use Ansible Vault on a part of a group’s variables.

Tip: The group_vars/ and host_vars/ directories can exist in the playbook directory OR the inv ntory directory. If both paths exist, variables in the playbook directory will override variables s t in the inventory directory.

Tip: Keeping your inventory file and variables in a git repo (or other version control) is an exce lent way to track changes to your inventory and host variables.

2.7 How Variables Are Merged

By default variables are merged/flattened to the specific host before a play is run. This keeps nsible focused on the Host and Task, so groups don’t really survive outside of inventory and host matching. By default, Ansible overwrites variables including the ones defined for a group and/or host (see the hash_merge setting to change this) . The order/precedence is (from lowe t to highest):

- all group (because it is the ‘parent’ of all other groups)
- parent group
- child group
- host

When groups of the same parent/child level are merged, it is done alphabetically, and the last group loaded overwrites the previous groups. For example, an a_group will be merged with b group and b_group vars that match will overwrite the ones in a_group.

New in version 2.4.

Starting in Ansible version 2.4, users can use the group variable ansible_group_priority to cha ge the merge order for groups of the same level (after the parent/child order is resolved). The arger the number, the later it will be merged, giving it higher priority. This variable defaults to 1 if not set. For example:

a_group:

testvar: a

ansible_group_priority: 10

b_group

testvar: b

In this example, if both groups have the same priority, the result would normally have been te tvar == b, but since we are giving the a_group a higher priority the result will be testvar == a.

三、更多配置参数请点我