Patterns: targeting hosts and groups

When you execute Ansible through an ad hoc command or by running a playbook, you must choose which managed nodes or groups you want to execute against. Patterns let you run commands and playbooks against specific hosts and/or groups in your inventory. An Ansible pattern can refer to a single host, an IP address, an inventory group, a set of groups, or all hosts in your inventory. Patterns are highly flexible - you can exclude or require subsets of hosts, use wildcards or regular expressions, and more. Ansible executes on all inventory hosts included in the pattern.

Using patterns

You use a pattern almost any time you execute an ad hoc command or a playbook. The pattern is the only element of an :ref:`ad hoc command<intro_adhoc>` that has no flag. It is usually the second element:

> **System Message: ERROR/3** (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 14); *backlink*
>
> Unknown interpreted text role "ref".

```
ansible <pattern> -m <module_name> -a "<module options>"
```

For example:

```
ansible webservers -m service -a "name=httpd state=restarted"
```

In a playbook the pattern is the content of the `hosts:` line for each play:

```
- name: <play_name>
  hosts: <pattern>
```

For example:

```
- name: restart webservers
  hosts: webservers
```

Since you often want to run a command or playbook against multiple hosts at once, patterns often refer to inventory groups. Both the ad hoc command and the playbook above will execute against all machines in the `webservers` group.

Common patterns

This table lists common patterns for targeting inventory hosts and groups.

Description	Pattern(s)	Targets
All hosts	all (or *)	
One host	host1	
Multiple hosts	host1:host2 (or host1,host2)	
One group	webservers	
Multiple groups	webservers:dbservers	all hosts in webservers plus all hosts in dbservers
Excluding groups	webservers:!atlanta	all hosts in webservers except those in atlanta
Intersection of groups	webservers:&staging	any hosts in webservers that are also in staging

> **Note**
>
> You can use either a comma (`,`) or a colon (`:`) to separate a list of hosts. The comma is preferred when dealing with ranges and IPv6 addresses.

Once you know the basic patterns, you can combine them. This example:

```
webservers:dbservers:&staging:!phoenix
```

targets all machines in the groups 'webservers' and 'dbservers' that are also in the group 'staging', except any machines in the group 'phoenix'.

You can use wildcard patterns with FQDNs or IP addresses, as long as the hosts are named in your inventory by FQDN or IP address:

```
192.0.\*
\*.example.com
\*.com
```

You can mix wildcard patterns and groups at the same time:

```
one*.com:dbservers
```

Limitations of patterns

Patterns depend on inventory. If a host or group is not listed in your inventory, you cannot use a pattern to target it. If your pattern includes an IP address or hostname that does not appear in your inventory, you will see an error like this:

```
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: Could not match supplied host pattern, ignoring: *.not_in_inventory.com
```

Your pattern must match your inventory syntax. If you define a host as an :ref:`alias<inventory_aliases>`:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 105); *backlink***
>
> Unknown interpreted text role "ref".

```
atlanta:
  host1:
    http_port: 80
    maxRequestsPerChild: 808
    host: 127.0.0.2
```

you must use the alias in your pattern. In the example above, you must use `host1` in your pattern. If you use the IP address, you will once again get the error:

```
[WARNING]: Could not match supplied host pattern, ignoring: 127.0.0.2
```

Advanced pattern options

The common patterns described above will meet most of your needs, but Ansible offers several other ways to define the hosts and groups you want to target.

Using variables in patterns

You can use variables to enable passing group specifiers via the `-e` argument to ansible-playbook:

```
webservers:!{{ excluded }}:&{{ required }}
```

Using group position in patterns

You can define a host or subset of hosts by its position in a group. For example, given the following group:

```
[webservers]
cobweb
webbing
weber
```

you can use subscripts to select individual hosts or ranges within the webservers group:

```
webservers[0]       # == cobweb
webservers[-1]      # == weber
webservers[0:2]     # == webservers[0],webservers[1]
                    # == cobweb,webbing
webservers[1:]      # == webbing,weber
webservers[:3]      # == cobweb,webbing,weber
```

Using regexes in patterns

You can specify a pattern as a regular expression by starting the pattern with ~:

```
~(web|db).*\.example\.com
```

Patterns and ad-hoc commands

You can change the behavior of the patterns defined in ad-hoc commands using command-line options. You can also limit the hosts you target on a particular run with the `--limit` flag.

- Limit to one host

```
$ ansible -m [module] -a "[module options]" --limit "host1"
```

- Limit to multiple hosts

```
$ ansible -m [module] -a "[module options]" --limit "host1,host2"
```

- Negated limit. Note that single quotes MUST be used to prevent bash interpolation.

```
$ ansible -m [module] -a "[module options]" --limit 'all:!host1'
```

- Limit to host group

```
$ ansible -m [module] -a "[module options]" --limit 'group1'
```

Patterns and ansible-playbook flags

You can change the behavior of the patterns defined in playbooks using command-line options. For example, you can run a playbook that defines `hosts: all` on a single host by specifying `-i 127.0.0.2,` (note the trailing comma). This works even if the host you target is not defined in your inventory, but this method will NOT read your inventory for variables tied to this host and any variables required by the playbook will need to be specified manually at the command line. You can also limit the hosts you target on a particular run with the `--limit` flag, which will reference your inventory:

```
ansible-playbook site.yml --limit datacenter2
```

Finally, you can use `--limit` to read the list of hosts from a file by prefixing the file name with `@`:

```
ansible-playbook site.yml --limit @retry_hosts.txt
```

If :ref:`RETRY_FILES_ENABLED` is set to `True`, a `.retry` file will be created after the `ansible-playbook` run containing a list of failed hosts from all plays. This file is overwritten each time `ansible-playbook` finishes running.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 212); *backlink*

Unknown interpreted text role "ref".

ansible-playbook site.yml --limit @site.retry

To apply your knowledge of patterns with Ansible commands and playbooks, read :ref:`intro_adhoc` and :ref:`playbooks_intro`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 216); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 216); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]intro_patterns.rst, line 218)

Unknown directive type "seealso".

```
.. seealso::
```

```
:ref:`intro_adhoc`
    Examples of basic commands
:ref:`working_with_playbooks`
    Learning the Ansible configuration management language
`Mailing List <https://groups.google.com/group/ansible-project>`_
    Questions? Help? Ideas?  Stop by the list on Google Groups
:ref:`communication_irc`
    How to join Ansible chat channels
```