**copy - Copies files to remote locations**

**Synopsis**

* The copy module copies a file from the local or remote machine to a location on the remote machine. Use the fetch module to copy files from remote locations to the local box. If you need variable interpolation in copied files, use the template module.
* For Windows targets, use the win\_copy module instead.

**Options**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **parameter** | **required** | **default** | **choices** | **comments** |
| attributes  (added in 2.3) | no | None |  | Attributes the file or directory should have. To get supported flags look at the man page for *chattr* on the target system. This string should contain the attributes in the same order as the one displayed by *lsattr*.  aliases: attr |
| backup | no | no | * yes * no | Create a backup file including the timestamp information so you can get the original file back if you somehow clobbered it incorrectly. |
| content | no |  |  | When used instead of *src*, sets the contents of a file directly to the specified value. For anything advanced or with formatting also look at the template module. |
| decrypt  (added in 2.4) | no | Yes | * yes * no | This option controls the autodecryption of source files using vault. |
| dest | yes |  |  | Remote absolute path where the file should be copied to. If *src* is a directory, this must be a directory too. If *dest* is a nonexistent path and if either *dest* ends with "/" or *src* is a directory, *dest* is created. If *src* and *dest* are files, the parent directory of *dest*isn't created: the task fails if it doesn't already exist. |
| directory\_mode  (added in 1.5) | no |  |  | When doing a recursive copy set the mode for the directories. If this is not set we will use the system defaults. The mode is only set on directories which are newly created, and will not affect those that already existed. |
| follow  (added in 1.8) | no | no | * yes * no | This flag indicates that filesystem links in the destination, if they exist, should be followed. |
| force | no | yes | * yes * no | the default is yes, which will replace the remote file when contents are different than the source. If no, the file will only be transferred if the destination does not exist.  aliases: thirsty |
| group | no |  |  | Name of the group that should own the file/directory, as would be fed to *chown*. |
| local\_follow  (added in 2.4) | no | yes | * yes * no | This flag indicates that filesystem links in the source tree, if they exist, should be followed. |
| mode | no |  |  | Mode the file or directory should be. For those used to */usr/bin/chmod* remember that modes are actually octal numbers (like 0644). Leaving off the leading zero will likely have unexpected results. As of version 1.8, the mode may be specified as a symbolic mode (for example, u+rwx or u=rw,g=r,o=r). |
| owner | no |  |  | Name of the user that should own the file/directory, as would be fed to *chown*. |
| remote\_src  (added in 2.0) | no | no | * yes * no | If no, it will search for *src* at originating/master machine.  If yes it will go to the remote/target machine for the *src*. Default is no.  Currently *remote\_src* does not support recursive copying. |
| selevel | no | s0 |  | Level part of the SELinux file context. This is the MLS/MCS attribute, sometimes known as the range. \_default feature works as for *seuser*. |
| serole | no |  |  | Role part of SELinux file context, \_default feature works as for *seuser*. |
| setype | no |  |  | Type part of SELinux file context, \_default feature works as for *seuser*. |
| seuser | no |  |  | User part of SELinux file context. Will default to system policy, if applicable. If set to \_default, it will use the user portion of the policy if available. |
| src | no |  |  | Local path to a file to copy to the remote server; can be absolute or relative. If path is a directory, it is copied recursively. In this case, if path ends with "/", only inside contents of that directory are copied to destination. Otherwise, if it does not end with "/", the directory itself with all contents is copied. This behavior is similar to Rsync. |
| unsafe\_writes  (added in 2.2) | no |  | * yes * no | Normally this module uses atomic operations to prevent data corruption or inconsistent reads from the target files, sometimes systems are configured or just broken in ways that prevent this. One example are docker mounted files, they cannot be updated atomically and can only be done in an unsafe manner.  This boolean option allows ansible to fall back to unsafe methods of updating files for those cases in which you do not have any other choice. Be aware that this is subject to race conditions and can lead to data corruption. |
| validate | no | None |  | The validation command to run before copying into place. The path to the file to validate is passed in via '%s' which must be present as in the example below. The command is passed securely so shell features like expansion and pipes won't work. |

**Examples**

*# Example from Ansible Playbooks*

- copy:

src: /srv/myfiles/foo.conf

dest: /etc/foo.conf

owner: foo

group: foo

mode: 0644

*# The same example as above, but using a symbolic mode equivalent to 0644*

- copy:

src: /srv/myfiles/foo.conf

dest: /etc/foo.conf

owner: foo

group: foo

mode: u=rw,g=r,o=r

*# Another symbolic mode example, adding some permissions and removing others*

- copy:

src: /srv/myfiles/foo.conf

dest: /etc/foo.conf

owner: foo

group: foo

mode: u+rw,g-wx,o-rwx

*# Copy a new "ntp.conf file into place, backing up the original if it differs from the copied version*

- copy:

src: /mine/ntp.conf

dest: /etc/ntp.conf

owner: root

group: root

mode: 0644

backup: yes

*# Copy a new "sudoers" file into place, after passing validation with visudo*

- copy:

src: /mine/sudoers

dest: /etc/sudoers

validate: /usr/sbin/visudo -cf %s

*# Copy a "sudoers" file on the remote machine for editing*

- copy:

src: /etc/sudoers

dest: /etc/sudoers.edit

remote\_src: yes

validate: /usr/sbin/visudo -cf %s

*# Create a CSV file from your complete inventory using an inline template*

- hosts: all

tasks:

- copy:

content: |

HOSTNAME;IPADDRESS;FQDN;OSNAME;OSVERSION;PROCESSOR;ARCHITECTURE;MEMORY;

**{%** **for** host **in** hostvars **%}**

**{%** **set** vars **=** hostvars**[**host**|string]** **%}**

**{{** vars.ansible\_hostname **}}**;**{{** vars.remote\_host **}}**;**{{** vars.ansible\_fqdn **}}**;**{{** vars.ansible\_distribution **}}**;**{{** vars.ansible\_distribution\_version **}}**;**{{** vars.ansible\_processor**[**1**]** **}}**;**{{** vars.ansible\_architecture **}}**;**{{** **(**vars.ansible\_memtotal\_mb**/**1024**)|round|int** **}}**; # NOQA

**{%** **endfor** **%}**

dest: /some/path/systems.csv

backup: yes

run\_once: yes

delegate\_to: localhost