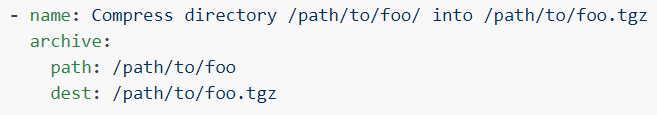
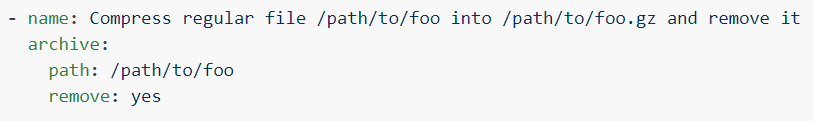
**Archive module:**

* Packs an archive.
* It is the opposite of unarchive.
* By default, it assumes the compression source exists on the target.
* It will not copy the source file from the local system to the target before archiving.
* Source files can be deleted after archival by specifying remove=True



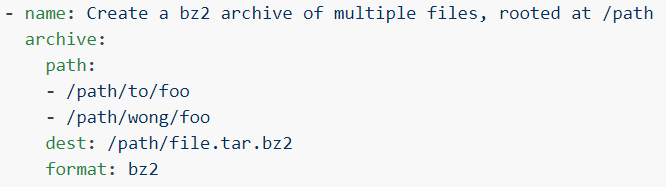
* Create a **“tgz”** file by giving the path



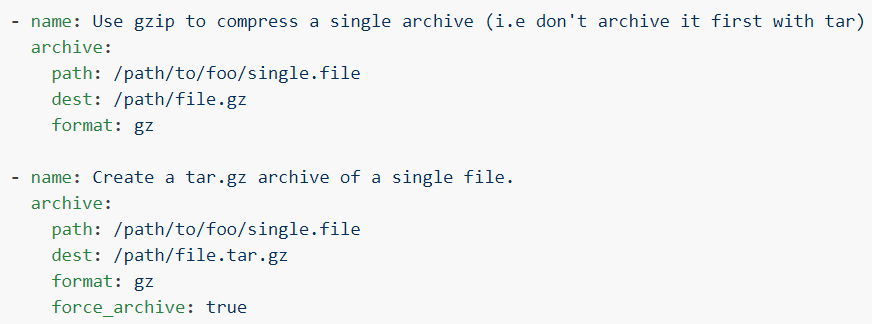
* Create a **“gz”** path and remove it.



* Creating a **“zip”** file.



* Archiving multiple files with **“bz2”**

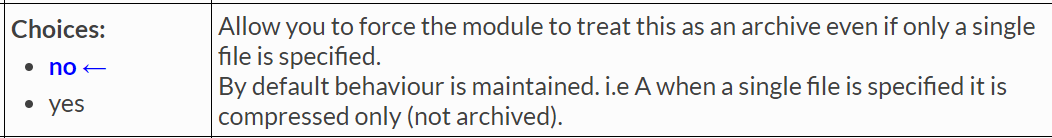


* Archiving with **“gz”** and **“tar.gz”** as above.

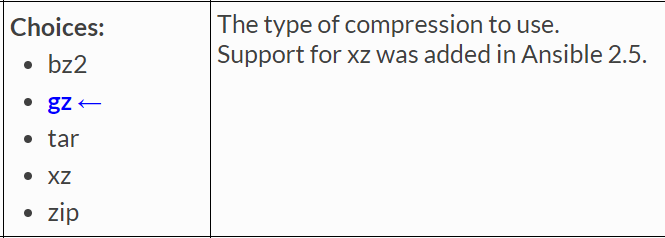
**Options:**

* **“group”**, **“owner”** and **“mode”** options are here which works the same was as we have in **“file”** module.

**force\_archive**



**format**



* **“format”,** we can give the type of compression we want to do. It is by default **“gz”**

**remove**



* With this, we can remove the source file after compressing it.

**Unarchive:**

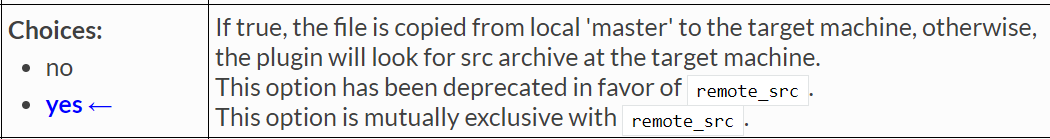
* The unarchive module unpacks an archive. It will not unpack a compressed file that does not contain an archive.
* By default, it will copy the source file from the local system to the target before unpacking.
* So, this module can also be used to copy any archive file from master to node and unarchive there.
* Set **remote\_src=yes** to unpack an archive which already exists on the target.
* If checksum validation is desired, use **get\_url** or uri instead to fetch the file and set **remote\_src=yes**.
* For Windows targets, use the win\_unzip module instead



**Options:**

We have “**owner”, “group” & “mode**” options here as well.

**copy**



* By default, this value is yes. So, it copies the source file to remote servers and unarchive there.
* If we want to unarchive any file which is already present in remote servers. We need to use **“remote\_src=yes”** option.