* Installation
* Getting Started
* Inventory
* Dynamic Inventory
* Patterns
* Introduction To Ad-Hoc Commands
* Configuration file
* BSD Support
* Windows Support
* Networking Support

**Installation**

**Topics**

* Installation
  + Basics / What Will Be Installed
  + What Version To Pick?
  + Control Machine Requirements
  + Managed Node Requirements
  + Installing the Control Machine
    - Latest Release Via Yum
    - Latest Releases Via Apt (Ubuntu)
    - Latest Releases Via Apt (Debian)
    - Latest Releases Via Portage (Gentoo)
    - Latest Releases Via pkg (FreeBSD)
    - Latest Releases on Mac OSX
    - Latest Releases Via OpenCSW (Solaris)
    - Latest Releases Via Pacman (Arch Linux)
    - Latest Releases Via Pip
    - Tarballs of Tagged Releases
    - Running From Source
  + Ansible on GitHub

**Basics / What Will Be Installed**

Ansible by default manages machines over the SSH protocol.

Once Ansible is installed, it will not add a database, and there will be no daemons to start or keep running. You only need to install it on one machine (which could easily be a laptop) and it can manage an entire fleet of remote machines from that central point. When Ansible manages remote machines, it does not leave software installed or running on them, so there’s no real question about how to upgrade Ansible when moving to a new version.

**What Version To Pick?**

Because it runs so easily from source and does not require any installation of software on remote machines, many users will actually track the development version.

Ansible’s release cycles are usually about four months long. Due to this short release cycle, minor bugs will generally be fixed in the next release versus maintaining backports on the stable branch. Major bugs will still have maintenance releases when needed, though these are infrequent.

If you are wishing to run the latest released version of Ansible and you are running Red Hat Enterprise Linux (TM), CentOS, Fedora, Debian, or Ubuntu, we recommend using the OS package manager.

For other installation options, we recommend installing via “pip”, which is the Python package manager, though other options are also available.

If you wish to track the development release to use and test the latest features, we will share information about running from source. It’s not necessary to install the program to run from source.

**Control Machine Requirements**

Currently Ansible can be run from any machine with Python 2 (versions 2.6 or 2.7) or Python 3 (versions 3.5 and higher) installed (Windows isn’t supported for the control machine).

**Note**

Ansible 2.2 introduces a tech preview of support for Python 3 (versions 3.5 and higher). For more information, see Python 3 Support.

This includes Red Hat, Debian, CentOS, OS X, any of the BSDs, and so on.

**Note**

As of version 2.0, Ansible uses a few more file handles to manage its forks. Mac OS X by default is configured for a small amount of file handles, so if you want to use 15 or more forks you’ll need to raise the ulimit with sudo launchctl limit maxfiles unlimited. This command can also fix any “Too many open files” error.

**Warning**

Please note that some modules and plugins have additional requirements. For modules these need to be satisfied on the ‘target’ machine and should be listed in the module specific docs.

**Managed Node Requirements**

On the managed nodes, you need a way to communicate, which is normally ssh. By default this uses sftp. If that’s not available, you can switch to scp inansible.cfg. You also need Python 2.6 or later.

**Note**

Ansible’s “raw” module (for executing commands in a quick and dirty way) and the script module don’t even need that. So technically, you can use Ansible to install python-simplejson using the raw module, which then allows you to use everything else. (That’s jumping ahead though.)

**Note**

If you have SELinux enabled on remote nodes, you will also want to install libselinux-python on them before using any copy/file/template related functions in Ansible. You can of course still use the yum module in Ansible to install this package on remote systems that do not have it.

**Note**

Ansible 2.2 introduces a tech preview of support for Python 3. For more information, see Python 3 Support.

By default, Ansible uses Python 2 in order to maintain compatibility with older distributions such as RHEL 6. However, some Linux distributions (Gentoo, Arch) may not have a Python 2.X interpreter installed by default. On those systems, you should install one, and set the ‘ansible\_python\_interpreter’ variable in inventory (see Inventory) to point at your 2.X Python. Distributions like Red Hat Enterprise Linux, CentOS, Fedora, and Ubuntu all have a 2.X interpreter installed by default and this does not apply to those distributions. This is also true of nearly all Unix systems.

If you need to bootstrap these remote systems by installing Python 2.X, using the ‘raw’ module will be able to do it remotely. For example,ansible myhost --sudo -m raw -a "yum install -y python2 python-simplejson" would install Python 2.X and the simplejson module needed to run ansible and its modules.

**Installing the Control Machine**

**Latest Release Via Yum**

**Note**

We’ve changed how the Ansible community packages are distributed. For users of RHEL/CentOS/Scientific Linux version 7, the Ansible community RPM package will transition from the EPEL repository to the Extras channel. There will be no change for version 6 of RHEL/CentOS/Scientific Linux since Extras is not a part of version 6.

RPMs for RHEL7 are available from the Extras channel.

RPMs for RHEL6 are available from yum for EPEL 6 and currently supported Fedora distributions.

Ansible will also have RPMs/YUM-repo available at `<https://releases.ansible.com/ansible/rpm/`\_.

Ansible version 2.4 can manage earlier operating systems that contain Python 2.6 or higher.

You can also build an RPM yourself. From the root of a checkout or tarball, use the make rpm command to build an RPM you can distribute and install.

$ git clone https://github.com/ansible/ansible.git

$ cd ./ansible

$ make rpm

$ sudo rpm -Uvh ./rpm-build/ansible-\*.noarch.rpm

**Latest Releases Via Apt (Ubuntu)**

Ubuntu builds are available in a PPA here.

To configure the PPA on your machine and install ansible run these commands:

$ sudo apt-get update

$ sudo apt-get install software-properties-common

$ sudo apt-add-repository ppa:ansible/ansible

$ sudo apt-get update

$ sudo apt-get install ansible

**Note**

On older Ubuntu distributions, “software-properties-common” is called “python-software-properties”.

Debian/Ubuntu packages can also be built from the source checkout, run:

$ make deb

You may also wish to run from source to get the latest, which is covered above.

**Latest Releases Via Apt (Debian)**

Debian users may leverage the same source as the Ubuntu PPA.

Add the following line to /etc/apt/sources.list:

deb http://ppa.launchpad.net/ansible/ansible/ubuntu trusty main

Then run these commands:

$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 93C4A3FD7BB9C367

$ sudo apt-get update

$ sudo apt-get install ansible

**Note**

This method has been verified with the Trusty sources in Debian Jessie and Stretch but may not be supported in earlier versions.

**Latest Releases Via Portage (Gentoo)**

$ emerge -av app-admin/ansible

To install the newest version, you may need to unmask the ansible package prior to emerging:

$ echo 'app-admin/ansible' >> /etc/portage/package.accept\_keywords

**Note**

If you have Python 3 as a default Python slot on your Gentoo nodes (default setting), then you must set ansible\_python\_interpreter = /usr/bin/python2 in your group or inventory variables.

**Latest Releases Via pkg (FreeBSD)**

$ sudo pkg install ansible

You may also wish to install from ports, run:

$ sudo make -C /usr/ports/sysutils/ansible install

**Latest Releases on Mac OSX**

The preferred way to install ansible on a Mac is via pip.

The instructions can be found in Latest Releases Via Pip section.

**Latest Releases Via OpenCSW (Solaris)**

Ansible is available for Solaris as SysV package from OpenCSW.

*# pkgadd -d http://get.opencsw.org/now*

*# /opt/csw/bin/pkgutil -i ansible*

**Latest Releases Via Pacman (Arch Linux)**

Ansible is available in the Community repository:

$ pacman -S ansible

The AUR has a PKGBUILD for pulling directly from Github called ansible-git.

Also see the Ansible page on the ArchWiki.

**Note**

If you have Python 3 as a default Python slot on your Arch nodes (default setting), then you must set ansible\_python\_interpreter = /usr/bin/python2in your group or inventory variables.

**Latest Releases Via Pip**

Ansible can be installed via “pip”, the Python package manager. If ‘pip’ isn’t already available in your version of Python, you can get pip by:

$ sudo easy\_install pip

Then install Ansible with [1]:

$ sudo pip install ansible

Or if you are looking for the latest development version:

pip install git+https://github.com/ansible/ansible.git@devel

If you are installing on OS X Mavericks, you may encounter some noise from your compiler. A workaround is to do the following:

$ sudo CFLAGS=-Qunused-arguments CPPFLAGS=-Qunused-arguments pip install ansible

Readers that use virtualenv can also install Ansible under virtualenv, though we’d recommend to not worry about it and just install Ansible globally. Do not use easy\_install to install ansible directly.

**Tarballs of Tagged Releases**

Packaging Ansible or wanting to build a local package yourself, but don’t want to do a git checkout? Tarballs of releases are available on the Ansible downloads page.

These releases are also tagged in the git repository with the release version.

**Running From Source**

Ansible is easy to run from a checkout - root permissions are not required to use it and there is no software to actually install. No daemons or database setup are required. Because of this, many users in our community use the development version of Ansible all of the time so they can take advantage of new features when they are implemented and easily contribute to the project. Because there is nothing to install, following the development version is significantly easier than most open source projects.

**Note**

If you are intending to use Tower as the Control Machine, do not use a source install. Please use OS package manager (like apt/yum) or pip to install a stable version.

To install from source, clone the Ansible git repository:

$ git clone https://github.com/ansible/ansible.git --recursive

$ cd ./ansible

Once git has cloned the Ansible repository, setup the Ansible environment:

Using Bash:

$ source ./hacking/env-setup

Using Fish:

$ . ./hacking/env-setup.fish

If you want to suppress spurious warnings/errors, use:

$ source ./hacking/env-setup -q

If you don’t have pip installed in your version of Python, install pip:

$ sudo easy\_install pip

Ansible also uses the following Python modules that need to be installed [1]:

$ sudo pip install -r ./requirements.txt

To update ansible checkouts, use pull-with-rebase so any local changes are replayed.

$ git pull --rebase

Note: when updating ansible checkouts that are v2.2 and older, be sure to not only update the source tree, but also the “submodules” in git which point at Ansible’s own modules.

$ git pull --rebase *#same as above*

$ git submodule update --init --recursive

Once running the env-setup script you’ll be running from checkout and the default inventory file will be /etc/ansible/hosts. You can optionally specify an inventory file (see Inventory) other than /etc/ansible/hosts:

$ echo "127.0.0.1" > ~/ansible\_hosts

$ export ANSIBLE\_INVENTORY**=**~/ansible\_hosts

**Note**

ANSIBLE\_INVENTORY is available starting at 1.9 and substitutes the deprecated ANSIBLE\_HOSTS

You can read more about the inventory file in later parts of the manual.

Now let’s test things with a ping command:

$ ansible all -m ping --ask-pass

You can also use “sudo make install”.