Beyond the basics

This page introduces some concepts that help you manage your Ansible workflow with directory structure and source control. Like the Basic Concepts at the beginning of this guide, these intermediate concepts are common to all uses of Ansible.

A typical Ansible filetree

Ansible expects to find certain files in certain places. As you expand your inventory and create and run more network playbooks, keep your files organized in your working Ansible project directory like this:

```
.
â”œâ”€â”€ backup
â”‚Â Â  â”œâ”€â”€ vyos.example.net_config.2018-02-08@11:10:15
â”‚Â Â  â”œâ”€â”€ vyos.example.net_config.2018-02-12@08:22:41
â”œâ”€â”€ first_playbook.yml
â”œâ”€â”€ inventory
â”œâ”€â”€ group_vars
â”‚Â Â  â”œâ”€â”€ vyos.yml
â”‚Â Â  â””â”€â”€ eos.yml
â”œâ”€â”€ roles
â”‚Â Â  â”œâ”€â”€ static_route
â”‚Â Â  â””â”€â”€ system
â”œâ”€â”€ second_playbook.yml
â””â”€â”€ third_playbook.yml
```

The `backup` directory and the files in it get created when you run modules like `vyos_config` with the `backup: yes` parameter.

Tracking changes to inventory and playbooks: source control with git

As you expand your inventory, roles and playbooks, you should place your Ansible projects under source control. We recommend `git` for source control. `git` provides an audit trail, letting you track changes, roll back mistakes, view history and share the workload of managing, maintaining and expanding your Ansible ecosystem. There are plenty of tutorials and guides to using `git` available.