Terminal plugins

Terminal plugins contain information on how to prepare a particular network device's SSH shell is properly initialized to be used with Ansible. This typically includes disabling automatic paging, detecting errors in output, and enabling privileged mode if supported and required on the device.

These plugins correspond one-to-one to network device platforms. Ansible loads the appropriate terminal plugin automatically based on the `ansible_network_os` variable.

Adding terminal plugins

You can extend Ansible to support other network devices by dropping a custom plugin into the `terminal_plugins` directory.

Using terminal plugins

Ansible determines which terminal plugin to use automatically from the `ansible_network_os` variable. There should be no reason to override this functionality.

Terminal plugins operate without configuration. All options to control the terminal are exposed in the `network_cli` connection plugin.

Plugins are self-documenting. Each plugin should document its configuration options.

Viewing terminal plugins

These plugins have migrated to collections on Ansible Galaxy. If you installed Ansible version 2.10 or later using `pip`, you have access to several terminal plugins. To list all available terminal plugins on your control node, type `ansible-doc -t terminal -l`. To view plugin-specific documentation and examples, use `ansible-doc -t terminal`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\plugins\[ansible-devel][docs][docsite][rst][plugins]terminal.rst, line 40`)

Unknown directive type "seealso".

```
.. seealso::

   :ref:`Ansible for Network Automation<network_guide>`
       An overview of using Ansible to automate networking devices.
   :ref:`connection_plugins`
       Connection plugins
   `User Mailing List <https://groups.google.com/group/ansible-devel>`_
       Have a question?  Stop by the google group!
   `irc.libera.chat <https://libera.chat/>`_
       #ansible-network IRC chat channel
```