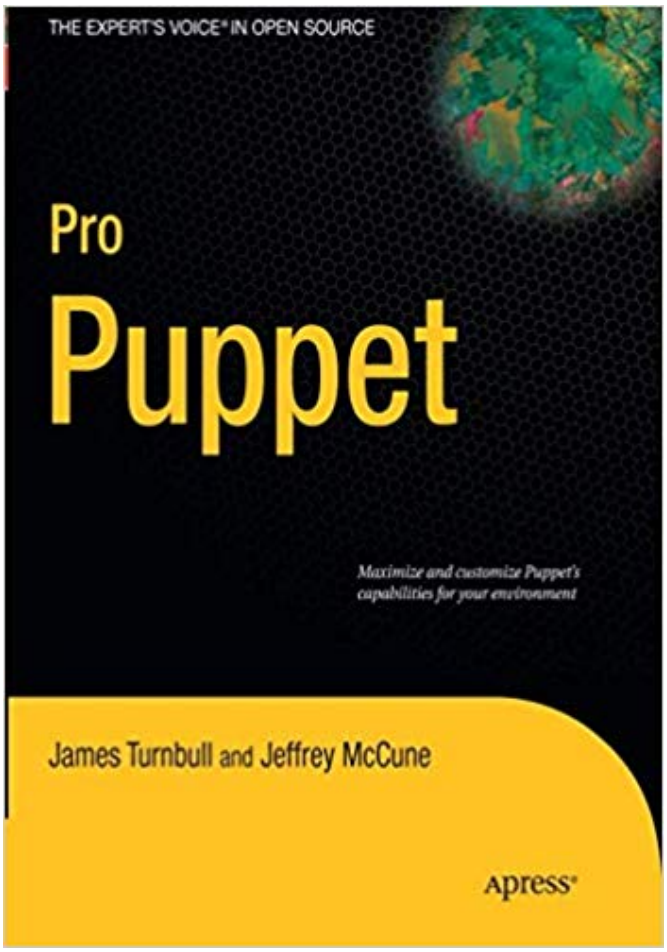
Pro Puppet



Pro Puppet 1st Edition

Back in 2011, James Turnbull partnered with Jeffrey McCune to produce a marvelous and technically complete sequel to his first book – Pulling Strings with Puppet, The called that sequel book Pro Puppet.

I'm moving Unix books reviews from another website, want to keep them here on Unix Tutorial in the Unix Book Reviews section before I start publishing more recent reviews. This review is for the 1st edition of the Pro Puppet, but I know there's been a 2nd edition of Pro Puppet written in 2013.

As it should be obvious from the title, this book is aimed at experienced users of the **Puppet configuration management system**, most likely seasoned systems administrators which have been managing systems with **Puppet** for a while but feel there

is room for improvement.

The **Pro Puppet book** does not disappoint: not only is it an updated introductory material for those of you only discovering Puppet, but it is also a step-by-step, full source code examples kind of a guide to solving more complex issues facing a serious **Puppet deployment** – scalability, **Puppet** modules, stored configurations and **MCollective** are just some of the topics explained in plenty of detail.

Puppet basics revisited

The first few chapters are talking about fundamental features and basic scenarios of deploying **Puppet management system** within your environment. You'll learn about the super-easy way of describing **Puppet nodes** and using nodes inheritance in Puppet server's config.

Naturally, there are full-text examples of creating your own configurations using Puppet classes and modules. There's even a quick intro in case you decide to write a function or two – these are Puppet functionality elements running on the server side.

Class inheritance is shown quite expertly – not just the basics of having separate modules for managing different services with Puppet, but the acctual class-based approach to stopping-starting services – essentially you can have the same class used for installing software (like DNS or NTP server), and then have the flexibility of using different classes for toggling the enabled/disabled state of the freshly installed service for different nodes

Cool stuff you can do with Puppet

Apparently, there is now a new provider specifically for auditing files – very similar to the File one, it only reports the compliance in terms of permissions and ownership for a

given file. There is enough flexibility to get the audit reports exactly the way you need them.

Another really cool thing I've learned is that it's possible and quite convenient to have classes require specific files to be in place before the class functionality is applied. I've been familiar with dependencies before but benefited from extra examples involving custom classes.

I always thought it would be great to use **Puppet system** for deploying **Pupppet infrastructure** itself – server and nodes. Turns out, this is entirely possible – the book includes example of a completely self-referential **Puppet deployment**.

Scaling Puppet environment

There are quite a few challenges you'll be facing when your Puppet environment grows to be large enough. The **Pro Puppet book** gives you advices for most scenarios.

First things first – you have got to use multiple deployment environments, for instance test/dev/prod. From the Puppet server perspective, this will mean getting familiar with how you describe these environments in the puppet.conf file and also creating separate directories for your modules. The approach given in the book will help you cater for both different environments (multiple nodes belonging to production or test environment) and for properly managing stages of Puppet module developments.

The really good thing is that you'll have plenty of examples of how to manage it all with a source control system (git).

When it comes to horizontally scaling the server aspects of Puppet, you'll find a lot of instructions for fronting Puppet instance with Apache webserver via mod_rails (Passenger) module. Naturally, some of the most probable scenarios are described and provided with solutions, so if you're stuck for some immediate help on making your crawling Puppet server run

nice and fast, you'll find some easy to follow steps.

What I enjoyed throughout the book is its attention to detail: it's easy to see how some chapters address not just an isolated issue but the full-scale solution. In case of scaling, you'll certainly appreciate hints on automating the data synchronization between Puppet backends – unless they reside on the same Unix environment, you'll need some behind-the-scenes tricks to make sure all the backeds are in full sync – be it for the Puppet modules/files or SSL certficiates for the Puppet CA element.

Externalizing Puppet configs (storing nodes info in a database

As soon as your **Puppet nodes.pp file** grows past the first few hundred hosts, you'll get this feeling that things could be greatly improved if you managed nodes list in a database of some sorts. Puppet server comes with such an abstraction planned from the very beginning, so it should be easy enough for you to externalize the nodes configuration. You can start off by using external text file or even a shell script, but the same approach and interface can be taken for Ruby or Perl, LDAP or MySQL.

Full text examples make it very easy to get started, you are ready to plug whole scripts into your infrastructure as even LDAP ldif files are provided for your convenience.

Exporting and storing configs of your Puppet managed node

You can configure Puppet master to use MySQL DB for storing all the configs related to managed nodes. In contrast with the nodes list externalization, this functionality will actually store metadata about your nodes – things like Facter facts – which normally reside locally on each node. Once configured,

such a setup may prove to be very useful for syncing configurations between nodes.

A really cool example given in the book is the one for collecting public ssh keys and then distributing the in the updated known_hosts file form.

Puppet modules using Puppet Forge

If you end up using Puppet for managing your environments, it will be only a matter of time before you get curious enough to attempt a development of your own Puppet module. You are in luck: the Pro Puppet book will gve you all the info you need to get started. Apart from learning how to use Puppet Forge for downloading new Puppet modules for use in your environment, the are some steps for configuring multiple source control trunks to take care of all the stages of a typical module development lifecycle. And if you think your newly created module will make a good addition to **Puppet Forge**, there are instructions on how to upload your module.

Extending Puppet and Facter

If you want to get the most out of your Puppet deployment, you'll probably appreciate the sections of the book talking about Puppet improvements like writing your own functions (remember, they are server side!) or custom Facter facts. There are always many different ways to make your changes or deploy custom Facts, and even if they are not shown in every single detail, there is certainly enough information to show you how things are done and help you get moving in the right direction with your Puppet infrastructure.

Using MCollective with Facter and Puppet

One of the reasons many people are buying the **Pro Puppet book** is the chapter talking about **Marionette Collective – MCollective**. It's a message bus solution for rapid scanning of

your Unix servers and for instant command execution. Instead of using SSH or similar mechanism for connecting to each client, **MCollective** relies on a message system like **ActiveMQ** or **RabbitMQ** (both freely available online) so that all the clients are listening to a queue and execute commands as soon as something relevant shows up.

The really powerful way to use **MCollective** is to leverage the power of custom facts of **Facter**. Essentially this means that you abstract from the common list of nodes and instead use specific facts about each node to compile the list you're interested in. Instead of generating a list of hosts, you can have **MCollective** instantly compile a list base on the OS flavor or environment description fact, and target your query at that list.

Summary for the Pro Puppet book

Without a doubt, this is one of the most useful books you can find on Puppet configuration management today. Whether you're after a high level introduction or enjoy all the possible technical details, you will find the **Pro Puppet** to be very relevant, highly educational and amazingly thorough about quite a number of Puppet related topics.

Puppet Configuration Management – links

- Puppet Open Source – configuration management like you've never seen before
- Puppet Forge – a growing collecting of Puppet configuration modules
- Facter – cross-platform library for retrieving operating system facts
- MCollective -rapid query and command orchestration tool
- Apache – the de-facto webserver for Unix
- Phusion Passenger (aka mod_rails) – Apache module for running Ruby and therefore scaling Puppet