MCollective Introduction

The Marionette Collective AKA MCollective is a framework to build server orchestration or parallel job execution systems. Primarily we’ll use it as a means of programmatic execution of Systems Administration actions on clusters of servers. MCollective use modern tools like [Publish Subscribe Middleware](http://en.wikipedia.org/wiki/Publish/subscribe) and modern philosophies like real time discovery of network resources using meta data and not hostnames. Delivering a very scalable and very fast parallel execution environment.

2) What is MCollective and what does it allow you to do?

* Interact with small to very large clusters of servers
* Use a [broadcast paradigm](http://docs.puppetlabs.com/mcollective/reference/basic/messageflow.html) for request distribution]. All servers get all requests at the same time, requests have filters attached and only servers matching the filter will act on requests. There is no central asset database to go out of sync, the network is the only source of truth.
* Break free from ever more complex naming conventions for hostnames as a means of identity. Use a very rich set of meta data provided by each machine to address them. Meta data comes from [Puppet](http://docs.puppetlabs.com/mcollective/reference/integration/puppet.html), [Chef](http://docs.puppetlabs.com/mcollective/reference/integration/chef.html), [Facter](http://code.google.com/p/mcollective-plugins/wiki/FactsRLFacter), [Ohai](http://code.google.com/p/mcollective-plugins/wiki/FactsOpsCodeOhai) or [plugins](http://docs.puppetlabs.com/mcollective/reference/plugins/facts.html) you provide yourself.
* Comes with simple to use command line tools to call remote agents.
* Ability to write [custom reports](http://docs.puppetlabs.com/mcollective/reference/ui/nodereports.html) about your infrastructure.
* A number of agents to manage packages, services and other common components are [available from the community](http://code.google.com/p/mcollective-plugins/).
* Allows you to write [simple RPC style agents, clients](http://docs.puppetlabs.com/mcollective/simplerpc/) and Web UIs in an easy to understand language - Ruby
* Extremely pluggable and adaptable to local needs
* Middleware systems already have rich [authentication and authorization models](http://docs.puppetlabs.com/mcollective/reference/integration/activemq_security.html), leverage these as a first line of control. Include fine grained [Authentication](http://docs.puppetlabs.com/mcollective/reference/plugins/security_ssl.html), [Authorization](http://docs.puppetlabs.com/mcollective/simplerpc/authorization.html) and [Auditing](http://docs.puppetlabs.com/mcollective/simplerpc/auditing.html) of requests.