*Configuration management (CM) comprises the detailed description and updating of information related to an enterprises software and hardware. This article lists the top 10 free and open source tools that can help you carry out configuration management with ease.*

Configuration management (CM) usually includes the updates and the versions that have been applied to software packages, as well as locations and network addresses of hardware devices. When a system needs a software or hardware upgrade, the user can access the CM program and database to see what is currently installed and then make a more informed decision about the upgradation needed.  
The advantages of software configuration management (SCM) are:

* It reduces redundant work
* It effectively manages simultaneous updates
* It avoids configuration related problems
* It simplifies coordination between team members
* It is helpful in tracking defects

**Top ten tools for configuration management**

**Chef**  
Chef is one of the most popular SCM tools. It is basically a framework for infrastructure development. It provides support and packages for framing ones infrastructure as code. It offers libraries for building up an infrastructure, which can be deployed easily. It produces consistent, shareable and reusable components, which are known as recipes and are used to automate infrastructure. It comprises the Chef server, workstation, repository and the Chef client.

***Chef server:***This is the central registry or brain of the entire process and acts as a hub. It carries information about the infrastructure and the cookbooks, which are used to instruct Chef on how each node must be configured.

***Chef client:*** This is installed on the nodes; it contacts the Chef server, fetches cookbooks and executes them on the nodes. The chef-client does the actual configuration.  
Chef provides various tools for designing the infrastructure. These are based on high-level programming languages such as Python and Ruby. Some of Chefs tools are:

* Ohai, a system profiling tool
* Shef, an interactive debugging console for testing and exploring Chefs behaviour
* Chef-Solo, a configuration management tool
* Chef client, an agent that runs on the system managed by Chef to communicate with the Chef server
* Knife, a multi-purpose command line tool

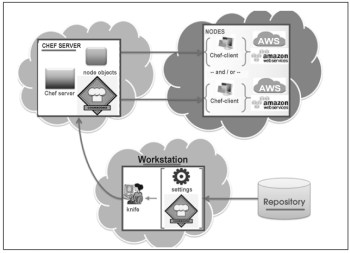
[](http://opensourceforu.com/2015/03/ten-tools-for-configuration-management/figure-1-chefs-components-6/)

Figure 1: Chef’s components

**Puppet**  
Another SCM tool commonly used is Puppet. It was first introduced in 2005 as an open source configuration management tool. It is written in Ruby. This CM system allows defining the state of the IT infrastructure, and then automatically enforces the correct state.  
It is used to manage the configuration of UNIX-like and Microsoft Windows systems. The user describes the systems resources and their state, either by using Puppets declarative language or a Ruby DSL. This information is stored in files known as Puppet manifests. It discovers system information through a utility called Facter and compiles it into a system-specific catalogue containing resources and their dependencies, which are applied against the target systems.  
When an application is deployed or is working with internal software, then Puppet automates every step of the delivery processfrom the provisioning of physical and virtual machines, to reporting code development through testing, updates and product release. It ensures stability, reliability and consistency. It also aids close collaboration between developers and sys admins, enabling more efficient delivery of code.  
Once Puppet is installed, every node (physical server, device or virtual machine) in the infrastructure will have a Puppet agent installed on it. There will also be a server known as Puppet Master. Implementation takes place during Puppet runs as per the steps shown in Figure 2.

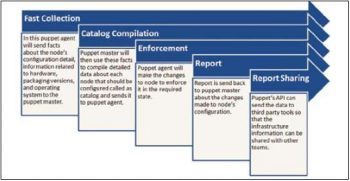
[](http://www.opensourceforu.com/?attachment_id=15234)

Figure 2: Puppet

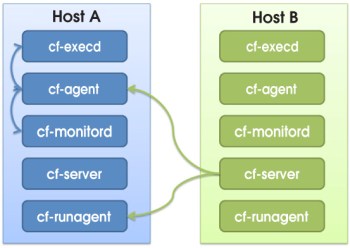
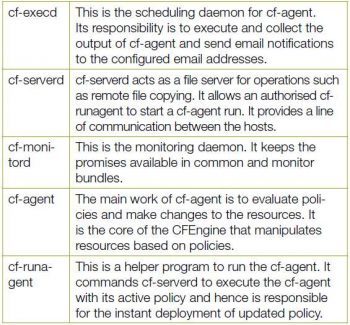
[](http://opensourceforu.com/2015/03/ten-tools-for-configuration-management/figure-3-cfengine-5/)

Figure 3: CFEngine

**CFEngine**  
CFEngine is one of the most popular open source and fully distributed CM systems and provides automated configuration compute resources. In addition, it provides cohesive management and monitoring of physical or virtual machines, tablets, embedded devices, cloud and mobile smartphones, and can be called an IT infrastructure automation framework. It is available in open source and commercial flavours. CFEngine is based on policies, which are used to deploy or patch up systems and are made up of promises. The latter are verified frequently to ensure full compliance with a desired state. Bringing resources into the desired state for compliance with policies is known as convergence. All policies are made available to all resources with the use of the policy server. This server monitors compute resources across the IT network, based on policies. Client hosts need to ensure that they continuously fetch the latest and most up-to-date policies from the policy server and that they remain compliant based on the set of instructions available in the policy language.  
Components in the CFEngine are independent agents and are the basis of automation.

[](http://www.opensourceforu.com/?attachment_id=15236)

cf-execd This is the scheduling daemon for cf-agent. Its responsibility is to execute and collect the output of cf-agent and send email notifications to the configured email addresses.  
cf-serverd cf-serverd acts as a file server for operations such as remote file copying. It allows an authorised cf-runagent to start a cf-agent run. It provides a line of communication between the hosts.  
cf-monitord This is the monitoring daemon. It keeps the promises available in common and monitor bundles.  
cf-agent The main work of cf-agent is to evaluate policies and make changes to the resources. It is the core of the CFEngine that manipulates resources based on policies.  
cf-runagent This is a helper program to run the cf-agent. It commands cf-serverd to execute the cf-agent with its active policy and hence is responsible for the instant deployment of updated policy.

**Juju**  
Juju is an open source configuration management and orchestration management tool. It enables applications to be deployed, integrated and scaled on various types of cloud platforms faster and more efficiently. It allows users to export and import application architectures and reproduce the same environment at different phases on cloud platforms such as Joyent, Amazon Web Services, Windows Azure, HP Cloud and IBM.  
It is used to configure, deploy, scale, manage, monitor, diagnose, notify and efficiently maintain (through the use of best practices) Charms (applications as services) in any deployment model, such as public, private and hybrid clouds either from a powerful GUI or the command-line. It also supports customisation of Charms using Chef, Puppet or any other CM tool.  
The main mechanism behind Juju is known as Charms that can be written in any programming language, whose execution is supported via the command line. They are a collection of YAML configuration files.  
Clients are available for Ubuntu, Windows and Mac operating systems. Once you install the client, environments can be bootstrapped on various cloud platforms such as Windows Azure, HP Cloud, Joyent, Amazon Web Services and IBM.  
Juju supports the GUI as well as the command line. It has a more feature-rich command line in comparison to what can be achieved via the GUI, which supports the drag and drop feature to create a software stack. Juju is limited to Ubuntu on the server side. Its Python version has a community patch to support CentOS.

**Rudder**  
Rudder is an open source CM tool for managing IT infrastructures. It is written in Scala and works on top of the CFEngine. Rudder provides features such as host inventory, a Web interface, reusable policies, a policy editor, generation of per-host policies, API, change requests, etc.  
Rudders asset management function identifies nodes and their characteristics that can be useful to perform configuration management. CM applies rules on nodes. Rudder uses the asset management function to identify nodes for configuration management.

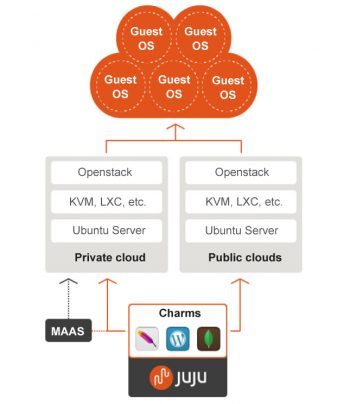
[](http://www.opensourceforu.com/?attachment_id=15237)

Figure 4: Juju deployment

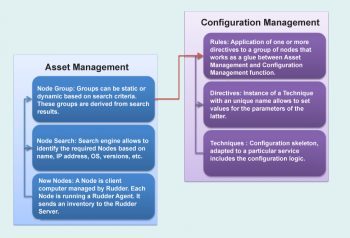
[](http://www.opensourceforu.com/?attachment_id=15238)

Figure 5: Rudder

**Ansible**  
Ansible is an open source platform for CM, orchestration and deployment of compute resources. It manages resources with the use of SSH (Paramiko, a Python SSH2 implementation, or standard SSH). Ansible supports and is available for CentOS and Red Hat Enterprise Linux, and it is also available as a commercial product by Ansible Inc. It is built on the popular Python language. It is possible to install Ansible by using the Git repository clone of a master server. Ansible is agentless. Its design goals are consistent, secure, minimal in nature and highly reliable, and it is easy to learn.  
Its main features are:

* Resources are added to the Ansible configuration.
* SSH authorised keys or sudo credentials (root access is not needed) are needed for each managed compute resource based on the user.
* Ansibles master server communicates with the compute resources using SSH and performs all the necessary tasks.
* Ansible deploys modules to compute resources.
* It monitors quantities such as CPU resources.

Playbooks are configuration files in Ansible, which use YAML syntax. Ansible has a collection of modules to manage resources on various cloud platforms such as Amazon EC2 and OpenStack.  
Ansible supports deployments on various virtualisation platforms as well as public and private cloud environments such as KVM, AWS, VMware, Eucalyptus Cloud, OpenStack and CloudStack. It also supports deployment of Big Data and analytics environments such as Hadoop, Riak and Aerospike.

**Salt**  
Salt is an open source multitasking CM and remote execution tool. It has a Python-based approach to represent infrastructure as a code philosophy. The remote execution engine is the heart of Salt. It creates a high speed and bi-directional communication network for a group of resources. A Salt state is a fast and flexible CM system on top of the communication system provided by the remote execution engine. It is a CLI-based tool. You can use Git to install it. It provides an asynchronous file server to speed up file serving. It also offers a number of modules to address operating systems, tools/applications and cloud platforms. The user can create custom modules in PyDSL or Python. Halite is a Web UI of Salt. Salt has a highly modular and extensible design to support diverse applications. The Salt system has different module types to manage specific actions such as remote execution and state management behaviour.  
The modules can be classified into six groups:

* Execution modules are used for direct execution from the remote execution engine
* State modules are used for the back-end of the Salt CM system
* Grains are used for detecting static information about a system
* Renderers are used to render the information passed to the Salt state system
* Returners allow the return information generated by the remote execution to be returned to an arbitrary location
* Runners are applications executed by the Salt-run command

The dependencies required to run Salt are Python 2.6 >= 2.6 <3.0, msgpack-python, YAML, Jinja2, MarkupSafe, apache-libcloud, Requests – HTTP library, mako and gcc.

**Vagrant**  
Vagrant is an open source CM tool for building and managing easy-to-configure, reproducible and portable virtual development environments. It works on top of virtualisation products such as KVM, VirtualBox, VMware and configuration management tools such as Chef, Puppet, Salt, etc. it is written in Ruby and supports various programming languages such as Java, PHP, C#, Python and JavaScript. It provides support for the libvirt plugin named vagrant-libvirt.  
To install Vagrant, download it from [*https://www.vagrantup.com/downloads*](https://www.vagrantup.com/downloads) based on the platform you are using, and install it following the regular procedure prescribed for your systems OS.  
Vagrant isolates dependencies and configuration within a single disposable, efficient and consistent environment. It also provides consistent workflow for developing and testing infrastructure management scripts.

**Bcfg2**  
Bcfg2 or bee-config is an open source fifth-generation CM tool that assists in con?guration analysis, visualising con?guration information, con?guration auditing, service deployment and reporting tools. It has a central configuration model that enables administrators to manage the configuration of compute resources efficiently. Bcfg2 uses the XML model for configuration specification. Thus, Bcfg2 uses XML schema validators and the custom schemas included in Bcfg2 to validate configuration specifications. It is portable and can be executed on GNU/Linux distributions such as openSUSE, Red Hat/RHEL, Fedora, Gentoo, CentOS, Debian, Ubuntu, FreeBSD, OpenBSD, OpenSolaris, Solaris, AIX, Mac OS X, etc. It enables consistent, repeatable and verifiable descriptions of the target environment. Bcfg2 has client-server architecture.

***Bcfg2 client:***

* It interprets the literal configuration provided by the Bcfg2 server
* It performs all client configuration or reconfiguration
* It does not need client-side processing of the configuration
* It uploads a set of statistics to the Bcfg2 server
* It is aware about how to implement declarative configuration modifications

[](http://www.opensourceforu.com/?attachment_id=15239)

Figure 6: Bcfg2

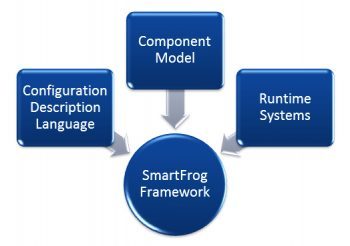
[](http://www.opensourceforu.com/?attachment_id=15240)

Figure7: SmartFrog components

***Bcfg2 server:***

* It accepts network descriptions and converts them into a series of configuration specifications
* It manages data provided by clients
* It keeps track of statistics for clients
* It takes information from a pool of metadata that describes clients, and from the file system repository for mappings from metadata to literal configuration

[](http://www.opensourceforu.com/?attachment_id=15241)

**SmartFrog**

SmartFrog (Smart Framework for Object Groups) is an open source tool written in Java. It is used to manage configuration, deployment and orchestration of various distributed components that comprise a robust system. A prototype-based object-oriented language or domain-specific language is used for component configuration. It encapsulates systems in such a way that it is easy to manage components, to install them automatically and to de-provision.  
SmartFrog consists of:

* **Configuration Description Language:** This is used to define configurations, to provide system modelling capabilities, notions of abstraction and template creation, and notation for describing system configurations.
* ***Component Model:*** This specifies how SmartFrog components can be written and integrated into the framework. It manages the lifecycle of components. It also supports dynamic discovery and the naming of components, distributed workflows, sequencing, script execution, installation and failure detection.
* ***Runtime System:*** This consists of daemons that run on the network. It creates components based on the SmartFrog systems descriptions, which is available in the Configuration Description Language. It enables interactions between components and lifecycle monitoring for executing tasks.  
  Modification in a configuration file can change the deployment architecture swiftly and, hence, it is easier to experiment and innovate. It also enables agility and scalability.

