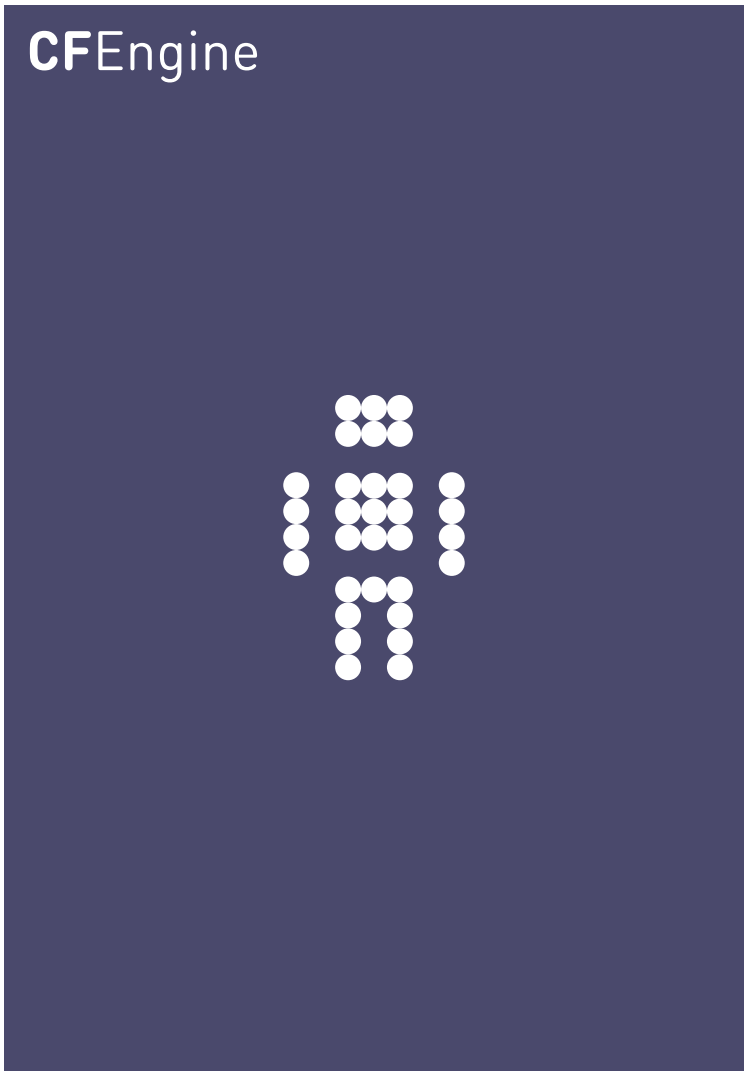


CFEngine and Other Configuration Management Software

A CFEngine Special Topics Handbook

CFEngine AS

This document points to a number of independent comparisons to other software systems in the configuration management space.

This document has the status of internal notes, and is not for redistribution.

Table of Contents

1 Sources

In this document we summarize the findings of a number of third party comparisons between CFEngine and related software.

- The State of Open Source System Automation, by Aleksey Tsalolikhin, in Linux Magazine, August 2010.[1]
- Thomas Delaat, post doctoral research site, 2010 (Belgium)[2]
- Jarle Bjorgeengen, Puppet and CFEngine Compared: Time and Resource Consumption, in USENIX ;login: journal[3]
- N. Timmers, S. Carlier, Automating Configuration CFEngine VS Puppet, University of Amsterdam System and Network Engineering technical report, LISA Project, March 29 2010.
- A. Tsalolikhin, http://verticalsysadmin.com/blog/uncategorized/relative-origins-of-CFEngine-chef-and-puppet
- Wikepedia, Comparison of open source configuration management software, http://en.wikipedia.org/wiki/Comparison_of_open_source_configuration_management_software
- http://www.johnmwillis.com/google/puppet-at-google/ (See appendix)
- http://www.masterzen.fr/2009/01/19/puppet-memory-leaks-or-not/
- http://divisionbyzero.net/blog/2009/01/18/from-CFEngine-to-puppet-a-retrospective/
- http://search.hp.com (Search "CFEngine")

[1] http://www.linux-mag.com/id/7841/1/, August 2010

[2] http://sysconfigtools.cer-wiki.info/tool/CFEngine3
http://sysconfigtools.cer-wiki.info/tool/puppet
http://sysconfigtools.cer-wiki.info/tool/chef
http://sysconfigtools.cer-wiki.info/tool/bcfg2

[3] http://www.usenix.org/publications/login/2010-02/pdfs/bjorgeengen.pdf

2 Comparing CFEngine to Alternatives

2.1 Inception

- CFEngine was conceived in 1993, with major upgrades in 2001 and 2008.
- BCFG2 was released in 2004.
- Puppet was released in 2005, as an offshoot of CFEngine 2.
- Chef was released in 2009 as an offshoot of Puppet.
- Bladelogic was founded in September 2001.
- HP Opsware was founded in September 1999 (as Loudcloud, Inc.).
- Tivoli Systems was acquired by IBM in 1996 for its storage manager. It is unclear when the first provisioning and change management product was released.

2.2 General Remarks

Most of the published comparisons that have been written between CFEngine and Puppet compare to CFEngine version 2.

According to the Wikepedia comparison, CFEngine runs on all documented operating systems. Puppet, Chef and BCFG2 run mainly on Linux, with partial but increasing support for other systems.

Puppet is a GPL, Open Source project, principally the work of Reductive Labs, which changed its name to Puppet Labs in 2010.

Chef is an Open Source project created by a company 'Opsware' and a breakaway community of coders from the Puppet community. Very little is known about Chef from independent sources. Its parent company Opscode was started with connections to the Amazon.com milieu. The project began in January 2009. Like Puppet, it does not seem to be based on documented research, but as an outgrowth of the Puppet project (See Tsalolikhin). Chef seems to be aimed principally at Cloud Computing. Opscode offers a 'boot server' for Chef in the Cloud to simplify the deployment as a managed service.

BCFG2 is an Open Source research project, principally the work or Narayan Desai of the Argonne National Laboratory. It is has been presented as a research project over a number of years, especially at USENIX conferences.

Commercial alternatives to these tools are also available. In practice, CFEngine competes more often head to head with IBM Tivoli, BMC Bladelogic and HP Opsware at larger sites. Each of which has extensive product suites that cover a broad range of issues. These occupy a completely different price category, even compared to the commercial release (CFEngine Nova), and are reputed to be cumbersome to install and use. We shall comment little on these products, as there is no independent verifiable source of information about them. The remarks cited here are essentially cited from a handfull of customers CFEngine has come into contact with.

2.3 Installation requirements

CFEngine is written in native C code, and uses a few common libraries that are bundled for convenience. Puppet and Chef require the Ruby and Perl development environments to cover

their dependency trees. Some of the Puppet library is implemented in Perl, thus two frameworks are needed to support a Puppet infrastructure. BCFG2 requires the Python development environment.

The installation size for Puppet is around 500MB, versus less than 30MB for CFEngine. Puppet uses around 70MB of memory in normal operation, versus about 2MB for CFEngine, according to measurements made by third parties.

2.4 Modus Operandi

CFEngine is a pull-based (service oriented) client-server system, with fully decentralized operation. It establishes the core principle of 'voluntary cooperation', in which no host can be forced to behave in a manner determined by an outside force, without its formally agreeing to do so. Thus every host is responsible for its own state, and performs autonomously, leveraging network services where appropriate. Autonomy is a core security principle in CFEngine. CFEngine has been reported in use on systems of tens of thousands of machines.

Users can write extensions in the declarative language provided by CFEngine, or embed custom scripts through an interface (written in any convenient language), but as a compiled language, it is not a simple matter for users to extend CFEngine without detailed technical competence.

CFEngine conversely tries to change the way system administrators think to observe certain beneficial principles based on research led by its founder. This research has received a lot of critical acclaim and the founder is recognized as one of the authoritative voices in System Administration or IT Operations.

Puppet is a push-pull system that borrows a number of ideas from CFEngine, and implements them in the Ruby framework. It backs away from the core principles of CFEngine and attempts to model 'what system administrators do', wrapped in an object oriented style.

Puppet prepares all configuration customizations centrally and pushes these to its satellite clients. This is recognized as a fundamental limitation on its scalability. See below.

Puppet's capabilities are relatively limited compared to CFEngine, but users may program extensions of their own by writing Ruby code with a little knowledge of scripting. Puppet has a language of its own, but this is not recursively enumerable, so it does not have sufficient power to express new features.

Puppet has obtained a popular following at sites with simple requirements, as it claims to model the way system administrators think 'hands-on' today. The parent company Reductive Labs, renamed Puppet Labs in 2010, has spent much effort on building community around Puppet.

BCFG2 is a centralized, push-based system with only a handful of installations. It remains an essentially private project run by Argonne National Lab.

BCFG2 is XML based and has a number of interesting features from a resource perspective, but it is essentially a centralized template engine for configuration files. BCFG2 is a pull-based system using 'push' semantics. All files are prepared on the central server for deployment out to satellites.

Use of BCFG2 is limited to a few institutions. It was designed mainly to cope with the environment and particular model of the Argonne National Lab. BCFG2 has published a number of research papers on its approach, and presents the impression of a careful and well executed piece of work. The software requires the Python framework to run.

Chef is a decentralized pull-based system. It is essentially a collection of traditional shell scripts, written in the Ruby framework. It has a small user base, and is a splinter project from the Puppet community, though its code seems to share little in common with Puppet.

The quality of workmanship seems to be high, but the style of programming is geared towards 'install once and leave', in contrast to CFEngine's continuous maintenance approach.

BMC Bladelogic/HP Opsware/IBM Tivoli

These software suites are push based deployment tools, with monitoring capabilities. They are considered very complex and take a considerable amount of time to implement. We find that these solutions are often sitting on the shelf and are considered risky due to the investment required to even get started. HP Opsware has a focus on the router and switch environments. IBM is strong on storage management. These solutions still rely solely on patching when systems get out of compliance, using packaged 'fixes'. They have no concept of policy-based real-time repair. HP use CFEngine for configuration management for HPUX, see HP server manuals for more information[1].

2.5 Authentication and encryption

CFEngine uses a simple application specific protocol for encryption and authentication based on a simplified treatment of the Open-SSH protocol. This requires its server component to run on hosts where files or data are copied from. The aim is to provide secure and lightweight data transfer. It has a fully decentralized Public Key System. The CFEngine Protocol has had only one security advisory since 1993. CFEngine has been approved for FIPS 140-2 validated use on government networks in the US.

Puppet uses XML-RPC over SSL/TLS for communication. This requires a basic web-server to be installed on each system. The goal is to use generic open source components rather than design its own protocol.

All other tools piggyback the SSL/TLS web protocol for communication, requiring the registration of a Certificate Authority. The OpenSSL implementation has had a small number (less than 10) of security advisories since 2000.

2.6 Knowledge Management and Reporting

CFEngine pioneered what is variously referred to as 'policy based' or 'model based' configuration. CFEngine's main focus since 2008 has been on integrating Knowledge Management features into its approach to management. Through its declarative 'Promise Theory' model

[1] CFEngine pages at the HP support site have recently been moved out of the reach of public access, but a search still reveals the document stubs.

(which is a model about knowledge and certainty), it enables users to add simple documentation that can then be automatically turned into a browsable model. CFEngine uses ISO 13250 Topic Maps to represent a meta-model of knowledge about the network and relate it to a model of the IT and business organization.

CFEngine's promise model makes a clear distinction between intended state and actual state. Intended state is that which is expressed as policy and is maintained by the automation system. Actual state is then observed by an independent component that is able to see trends and the success and failure of policy decisions. CFEngine brings these data together using a topic map in a browsable 'knowledge map'.

Puppet has started a dashboard 'CMDB' project with some basic reporting about host state. There is no integrated view of policy versus intended and actual state.

Chef has no concept of Knowledge Management, as far as we know.

BCFG2's central knowledge of all templates allows some interesting correlations and analyses to be extracted about a managed network, according to its author. These features are not well documented, however.

2.7 Documentation

- CFEngine has a 500 page reference manual, generated automatically and a few hundred pages more of special topics guides and tutorials. It has hundreds of code examples and an integrated knowledge base. The challenge in CFEngine has been to organize too much information rather than expand the amount. This has further motivated the attention to knowledge management.
- Puppet has around 30 pages of online web-based documentation.
- Chef does not seem to have any documentation, but seems to be designed to not require extensive documentation, in the manner of a black-box solution.
- BCFG2 has rather little documentation in its Wiki site, and is working on an online manual which is currently at about 30 pages.

2.8 Scalability

CFEngine is based on research that shows it is likely to scale to tens of thousands of machines without difficulty. This theory is verified by a handful of large scale users who have installations in this number. Scalability is assured in CFEngine by its autonomous, decentralized pull-based design. CFEngine is designed to run every 5 minutes to perform micro-adjustments to system state, or schedule longer tasks with a high resolution.

The scalability of the Puppet is difficult to assess. The system measures 25-40 times slower than CFEngine in operation, according the Bjorgeengen. It is designed to run at most once per hour. Varying reports indicate that trouble is encountered at 100 machines with default settings. By changing some of the components, some users have reported up to 1000 machines. Puppet has a theoretical limit of scaling due to its push-pull architecture. There will always be a trade-off between managing a large number of machines, and knowing their state accurately or observing them often. CFEngine's internal calculations suggest that Puppet will begin to fail at a few hundred machines, when run on an hourly schedule. Some confirmation about this has been received from blog posting and conversations with users.

Puppet has recently retracted claims of running 35000 servers at a major bank. Some Puppet users have turned to CFEngine recently due to problems in code reliability and heavy processing and network load. Recent measurements show that it is 25-40 times more resource intensive than CFEngine for simple tasks of checking files.

Chef has a decentralized architecture, but allows a centralized server to be included for coordination in the cloud. We have no data on the scalability of Chef installations. Recent advertising suggests installations of up to a hundred machines, but this is unclear.

BCFG2 has been installed on hundreds on machines, in a high availability environment, but it suffers from the same theoretical limitations as other centralized systems. The commercial products BladeLogic (BMC), HP Opsware, etc, are similarly based on a centralized template push approach, with similar consequences.

2.9 User-friendliness and ease of user

CFEngine has acquired reputation for being difficult to learn, mainly because it asks users to think in a new (and we claim better) way about the problems of system administration. Users often resist this initially. CFEngine asks users to think about abstract issues like knowledge, predictability and business alignment. Having tried other options, users often return to CFEngine and are enthusiastic and loyal to the CFEngine approach once they understand it. Often this is helped by a training session. CFEngine makes a point of not 'oversimplifying' problems to sell a solution, as this only causes trouble later. We provide examples and a generic library as a get-started-easily kit to explain a better approach.

Puppet is generally perceived by users as being easier to understand, and more 'the way system administrators think', allowing sysadmins to continue to think about problems in traditional ways. This has led to problems of scalability however. In terms of language, Puppet approaches several things in the same basic way as CFEngine. Feedback from users suggests that Puppet lets administrators keep the habits they are used to, even when such habits are undesirable.

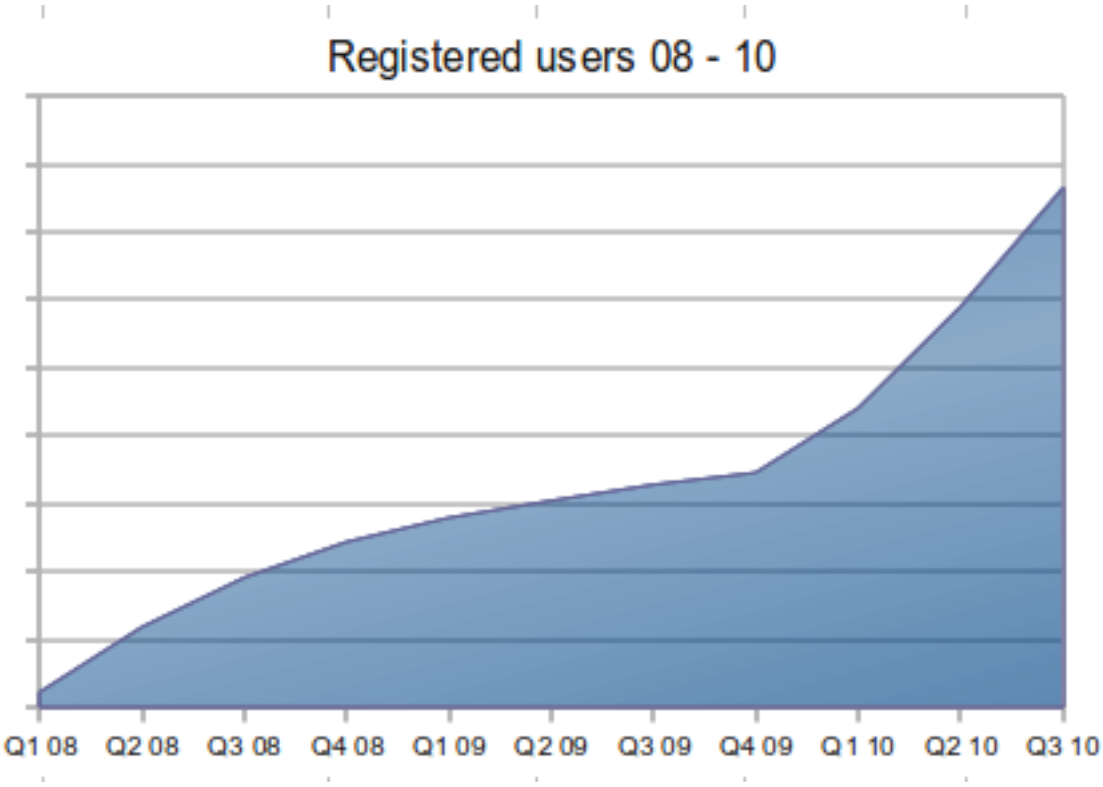
Chef seems to be mostly designed for immediate use, without much user customization. It contains a Domain Specific Language, but this is largely hidden from view.

BCFG is based on an XML document definition. We have no feedback about the perceived user-friendliness.

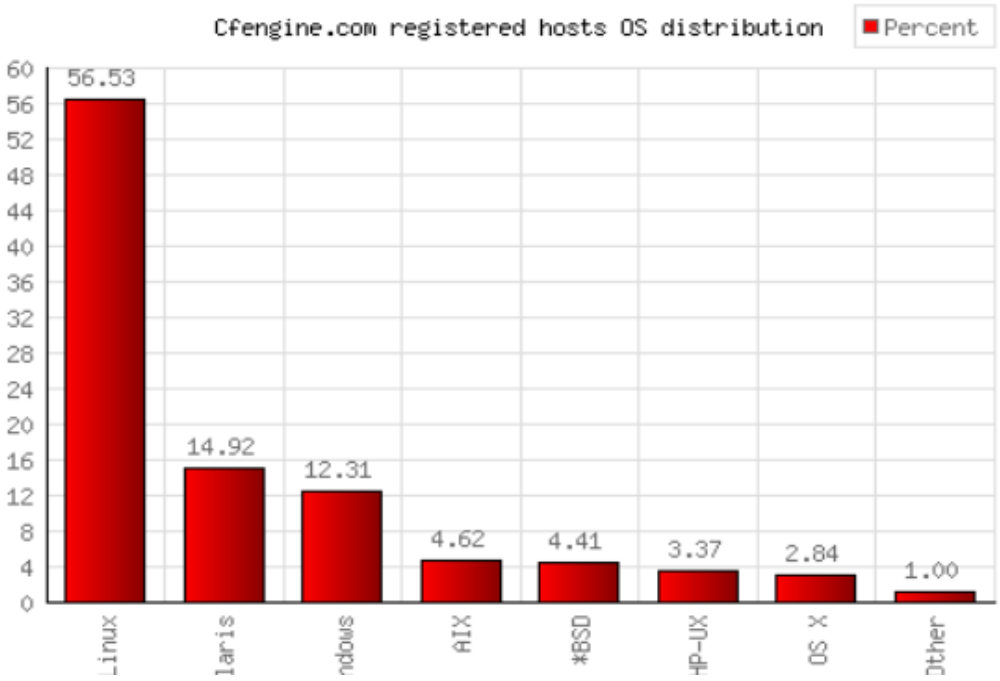
The commercial products from BMC, HP and IBM are reputed to be extremely complex to install and use. The 'get started' time for these systems has been cited by customers to be of the order of months to a year. In a number of cases, the products have never been fully installed. Customers often use one small part of the these product suites, e.g. for disk imaging. This suggests that, in spite of a polished 'look', these products are at least no more user friendly than Open Source sytsems.

2.10 Usage

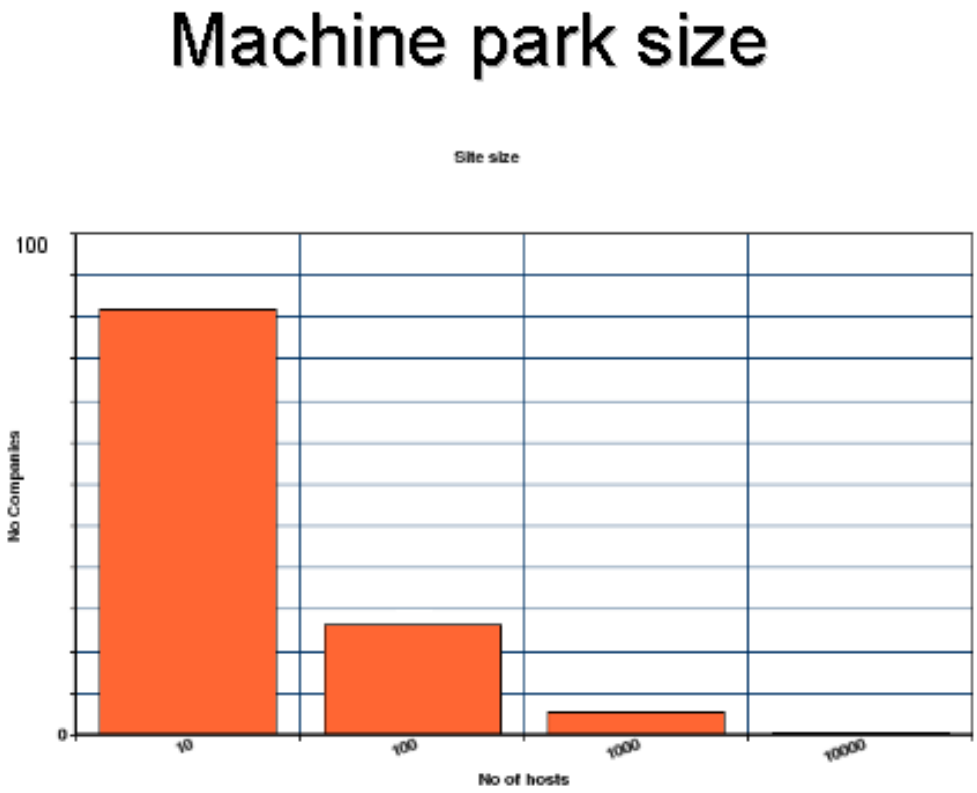
We have no reliable information about the number of installations of the various systems. CFEngine is estimated to be installed in several thousand organizations around the world, on about a million computers.



Past two years growth in user registration at CFEngine.com



Relative frequency of operating systems.



Size of recorded installations in 2008.

Puppet is thought to be installed on tens of thousands of computers. It was recently forced to retract inflated claims about its users, following the USENIX Configuration Management Summit, Boston 2010.

BCFG2 has a rather small following, with perhaps only a few hundred installations.

No information is available about the other projects.

2.11 Features unique to CFEngine

From assessing documentation, we believe that the following features are unique to CFEngine in the configuration management arena.

- A fully decentralized or federated management model.
- Availability of business metrics.
- Supporting scientific literature and research base.
- Extensive use-cases from all areas of industry and academia.
- Extensive programme for Knowledge Management.
- A 5 year road-map for development.

At a technical level:

- Use of user-definable patterns or models to simplify code.
- No single points of failure.
- Decentralized public key model (like SSH).

- Virtualization support.
- File editing language.
- Database management.
- Role based access control
- All known operating systems supported.
- Integrated monitoring and reporting.
- Dynamic runtime adaptation to local environment.
- US Government FIPS compliant mode.
- Multi-node orchestration capabilities.