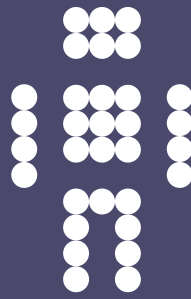


CFEngine



## CFEngine 3 Nova Pilot Handbook Supplement

CFEngine Enterprise Documentation

Updated 5. January 2011

CFEngine

This document supplements the 'CFEngine 3 Nova Pilot Handbook' with examples for all report types available in the CFEngine 3 Nova Mission Portal.

Copyright © 2011 CFEngine AS. The features described herein are in provided for user convenience and imply no warranty whatsoever to the extent of applicable law.

## Table of Contents

1	Standard reports in CFEngine 3 Nova .....	1
1.1	Bundle profile report .....	1
1.2	Business value report .....	2
1.3	Class profile report .....	3
1.4	Compliance by promise report .....	4
1.5	Compliance summary report .....	5
1.6	File change log report .....	6
1.7	File change diffs report .....	7
1.8	Last saw hosts report .....	9
1.9	Patches available report .....	10
1.10	Patch status report .....	11
1.11	Performance report .....	12
1.12	Promises repaired log report .....	13
1.13	Promises repaired summary report .....	14
1.14	Promises not kept log report .....	15
1.15	Promises not kept summary report .....	16
1.16	Setuid/gid root programs report .....	17
1.17	Software installed report .....	18
1.18	Variables report .....	19
2	CDP reports .....	21
2.1	ACLs report: File access controls .....	21
2.2	Commands report: Scheduled commands to execute .....	24
2.3	File changes report: File changes observed on the system .....	25
2.4	File diffs report: Delta/difference comparison showing file changes .....	26
2.5	Registry report: Promised Windows registry setting status .....	27
2.6	Services report: System service status .....	29

## 1 Standard reports in CFEngine 3 Nova

Standard reports in CFEngine 3 Nova can be accessed through the 'Reports finder' in the 'Engineering' room. The finder lists all standard report categories and each category contains information about different aspects of the Mission. When you click one of them, the 'Report finder' will present a query form that is adapted to the chosen report category. CFEngine uses regular expressions in these queries, for maximum flexibility and to minimize system impact. The details of these queries and the content of the resulting reports are outlined in the following sections.

### 1.1 Bundle profile report

Promises are the fundamental statements in CFEngine, they make up the definition of the desired state of a system. A *bundle* is a collection of promises in a 'sub-routine-like' container. The purpose of bundles is to allow greater flexibility to break down the contents of policies and give them names. Bundles also allow to re-use promise code by parameterizing it.

The 'Bundle profile report' is useful for checking when specific bundles were last verified and for seeing statistics about the frequency of verification. Click on **Bundle profile** in the 'Reports finder' to open a query window:

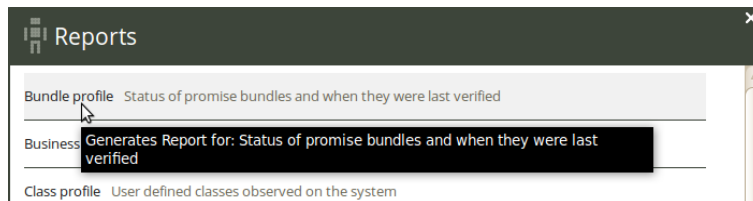


Figure: Go to Bundle profile query

The Bundle profile query can filter by bundle pattern (pattern in bundle name) and host class (i.e. the class/context of a bundle). Leaving the fields blank will result in a report listing all bundles in your policies.

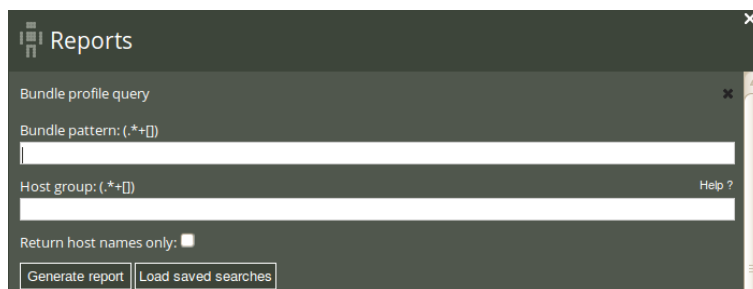



Figure: Bundle profile query

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of bundles that suit the query criteria entered above. It displays the host names on which these bundles can be found ('Host'), the name of the bundles ('Bundle'), the time stamp at the moment of verification ('Last verified'), the time passed since last verification ('Hours ago'), the average time between each verification ('Avg interval'), and the uncertainty of that average ('Uncertainty', measured in one standard deviation of 'Avg interval'). You can add your personal note in the right column,

documenting any thoughts or issues that you might have about the query result. The 'Last verified' value is yellow if more than six hours have passed since last verification.

Bundle profile

PDF  [Select host](#) [Select report](#) [Save this search](#) [New search](#)

Total results found: 89

HOST	BUNDLE	LAST VERIFIED	HOURS AGO	AVG INTERVAL	UNCERTAINTY	NOTE
hub.test.cfengine.com	<a href="#">CfengineEnterpriseFundamentals</a>	September 29, 2011 06:11:01 (GMT+2)	5.83	0.3	3.18	<a href="#">notes</a>
hub.test.cfengine.com	<a href="#">CfengineSiteConfiguration</a>	September 29, 2011 06:11:01 (GMT+2)	5.83	2.05	7.99	<a href="#">notes</a>
hub.test.cfengine.com	<a href="#">CfengineStdLibrary</a>	September 29, 2011 06:11:01 (GMT+2)	5.83	2.05	7.99	<a href="#">notes</a>
hub.test.cfengine.com	<a href="#">access_rules</a>	September 26, 2011 14:28:04 (GMT+2)	69.55	0.1	1.57	<a href="#">notes</a>

Figure: Bundle profile report

## 1.2 Business value report

One of the capabilities of CFEngine 3 Nova is to add business or organizational value to the configuration model. The notion of business value is not necessarily a clear concept, but a simple approach is to attach a monetary value to the outcome of promises.

CFEngine's default values for promises kept, promises repaired and promises not kept are 1, 0.5, and -1, respectively. The values are summed and recorded at a set time interval, and the results are summarized for every host and day.

Click on **Business value** in the 'Reports finder' to open a query window:

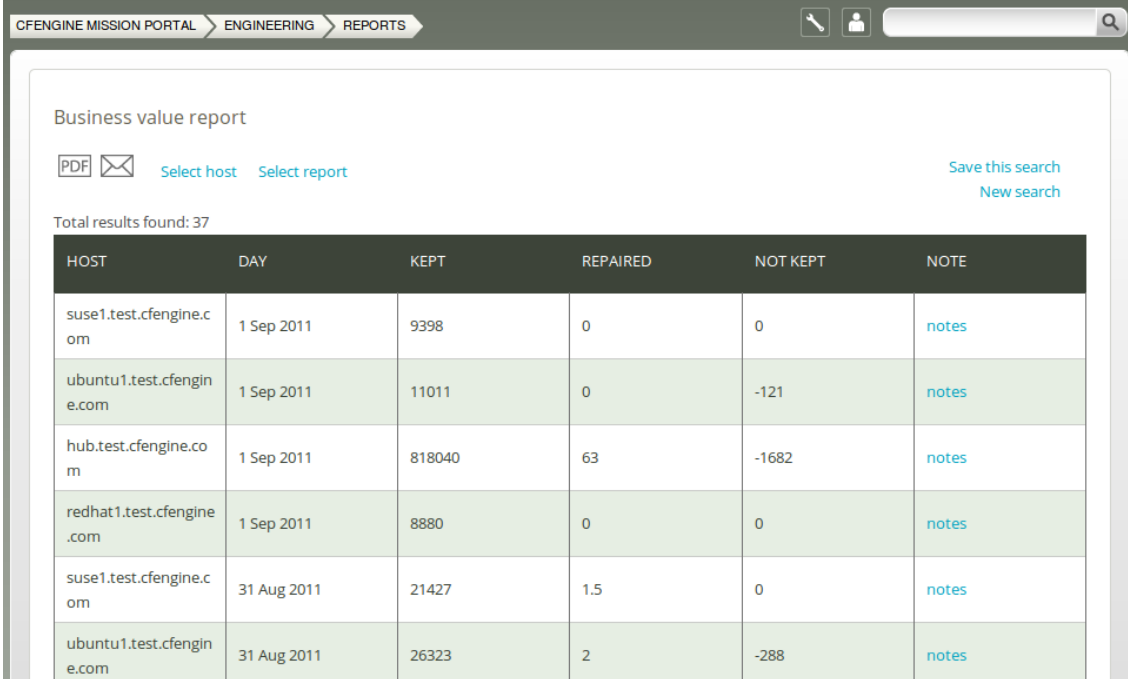


Figure: Business value query

The Business value query can filter by date and host class (i.e. the class/context of a host). Leaving the fields blank will result in a report listing the business value of all promises that have had value attached to them over all hosts and days.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents each host name ('Host'), date ('Day'), and the

sum of the value of the promises kept ('Kept'), repaired ('Repaired'), and not kept ('Not kept'). You can add your personal note in the right column, documenting any thoughts or issues that you might have about the query result.



CFENGINE MISSION PORTAL > ENGINEERING > REPORTS

Business value report

PDF ✉ Select host Select report Save this search New search

Total results found: 37

HOST	DAY	KEPT	REPAIRED	NOT KEPT	NOTE
suse1.test.cfengine.com	1 Sep 2011	9398	0	0	notes
ubuntu1.test.cfengine.com	1 Sep 2011	11011	0	-121	notes
hub.test.cfengine.com	1 Sep 2011	818040	63	-1682	notes
redhat1.test.cfengine.com	1 Sep 2011	8880	0	0	notes
suse1.test.cfengine.com	31 Aug 2011	21427	1.5	0	notes
ubuntu1.test.cfengine.com	31 Aug 2011	26323	2	-288	notes

Figure: Business value report

We will look at an example of how to attach business value of specific promises at a later stage, when it is time to edit a file in the integrated policy editor.

### 1.3 Class profile report

CFEngine classes are certain true/false (Boolean) propositions that determine in what context, or setting, a promise is made. Each time CFEngine runs (by default every five minutes), it discovers and classifies properties of the environment in which it runs. These discovered properties are called 'hard classes' and cannot be changed by users. CFEngine also operates with soft classes, i.e. user-defined types.

The Class profile report is useful for looking at hosts in specific contexts, for instance to find out which machines run on windows. Click on **Class profile** in the 'Reports finder' to open a query window:

The screenshot shows a web interface titled 'Reports'. It contains a 'Class profile query' section with the following fields and controls:

- Class pattern:** A text input field with a placeholder '(.\*+[])'.
- Host group:** A text input field with a placeholder '(.\*+[])' and a 'Help ?' link to its right.
- Return hostnames only:** A checkbox that is currently unchecked.
- Buttons:** 'Generate report' and 'Load saved searches'.

Figure: Class profile query

The class profile query can filter by (pattern in) class name and host class (i.e. the context of a host). Leaving the fields blank will result in a report listing all hosts and classes.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents the host names ('Host'), 'Class context', probability of occurrence ('Occurs with probability'), 'Uncertainty' (standard deviation of 'Occurs with probability'), and the last time the class was observed ('Last seen').

The screenshot shows a 'Class profile' report. At the top, there are icons for PDF and email, and links for 'Select host', 'Select report', 'Save this search', and 'New search'. Below this, it states 'Total results found: 220'. The main content is a table with the following data:

HOST	CLASS CONTEXT	OCCURS WITH PROBABILITY	UNCERTAINTY	LAST SEEN
hub.test.cfengine.com	10_0_0_29	0.0109	0.1613	September 29, 2011 14:26:18 (GMT+2)
hub.test.cfengine.com	64_bit	0.0109	0.1613	September 29, 2011 14:26:18 (GMT+2)
hub.test.cfengine.com	Lcycle_1	0.0109	0.1613	September 29, 2011 14:26:18 (GMT+2)

Figure: Class profile report

## 1.4 Compliance by promise report

Promises are the fundamental statements in CFEngine, the policy atoms. Promises can be made about all kinds of different subjects, from file attributes, to the execution of commands, access control decisions and knowledge relationships. If there is no promise, nothing happens. It is considered compliant if CFEngine is able to keep the promise, and conversely, not compliant, or not kept, in the opposite case.

The 'Compliance by promise' report is useful for checking the current status of your system in high detail. You can find out which parts of a bundle work and which do not. The report also predicts the probability of compliance based on the history of specific promises, allowing you to assess the degree to which the problem is of a more transient or permanent nature. Click on **Compliance by promise** in the 'Reports finder' to open a query window:

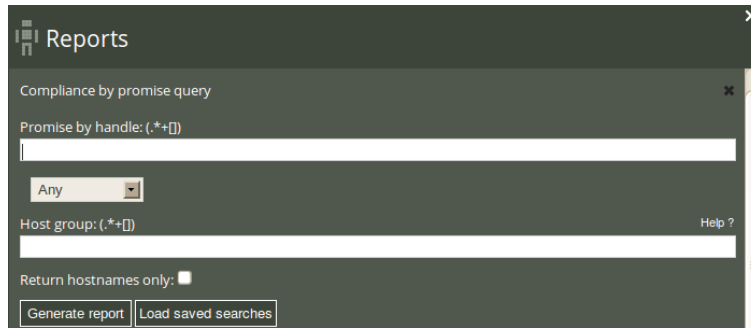


Figure: Compliance by promise query

The compliance by promise query can filter by (patterns in) promise handle, any/compliant/repaired/non-compliant promises (drop-down menu), and host class (i.e. the context of a host). Leaving the fields blank will result in a report listing all hosts and promises.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents the host names ('Host'), the promise identifier 'Promise handle', 'Last known state' (compliant or not compliant), likelihood of a promise being compliant ('Probability kept'), uncertainty of the likelihood ('Uncertainty', measured in one standard deviation of 'Probability kept'), and the time stamp of the last time the promise was run.

#### Compliance by promise

PDF [Select host](#) [Select report](#) [Save this search](#) [New search](#)

Total results found: 173

HOST	PROMISE HANDLE	LAST KNOWN STATE	PROBABILITY KEPT	UNCERTAINTY	LAST SEEN
hub.test.cfengine.com	<a href="#">internal_promise</a>	Not Compliant	0	0	September 29, 2011 15:20:29 (GMT+2)
hub.test.cfengine.com	<a href="#">knowledge_files_doc_root_application_logs</a>	Compliant	100	0	September 29, 2011 15:20:29 (GMT+2)
hub.test.cfengine.com	<a href="#">knowledge_files_doc_root_docs</a>	Compliant	100	0	September 29, 2011 15:20:29 (GMT+2)
hub.test.cfengine.com	<a href="#">knowledge_files_doc_root_hub</a>	Compliant	100	0	September 29, 2011 15:20:29 (GMT+2)

Figure: Compliance by promise report

### 1.5 Compliance summary report

CFEngine policies are collections of promises contained in a text file, they are the CFEngine scripts that define what state you want your system to be in. The compliance summary report gives an overview of policy status. It shows the current status of your system in a coarse manner, allowing you to quickly identify which areas need follow-up. Click on **Compliance summary** in the 'Reports finder' to open a query window:

The screenshot shows a web interface titled 'Reports'. It contains a 'Compliance summary query' section with the following fields and controls:

- 'Promise by handle: (\*+[])' with a text input field.
- A dropdown menu currently set to 'Any'.
- 'Host group: (\*+[])' with a text input field and a 'Help ?' link.
- 'Return hostnames only:' with a checkbox.
- 'Generate report' and 'Load saved searches' buttons.

Figure: Compliance summary query

The compliance summary query can filter by (pattern in) promise handle and host class (i.e. the context of a host). Leaving the fields blank will result in a report listing all hosts and and policies.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents the host names ('Host'), policy file name ('Policy'), percentage of promises kept within the listed policies ('Kept'), percentage of promises repaired within the listed policies ('Repaired'), percentage of promises not kept within the listed policies ('Not kept'), and the time stamp of the last status check ('Last seen').

#### Compliance summary

PDF [Select host](#) [Select report](#) [Save this search](#) [New search](#)

Total results found: 10251

HOST	POLICY	KEPT	REPAIRED	NOT KEPT	LAST SEEN
policy.test.cfengine.com	Failsafe.cf 2.1.0 (agent-0) Promises.cf 2.1.0 (agent-0)	100	0	0	September 29, 2011 15:25:53 (GMT+2)
hub.test.cfengine.com	Failsafe.cf 2.1.0 (agent-0) Promises.cf 2.1.0 (agent-0)	100	0	0	September 29, 2011 15:25:35 (GMT+2)
policy.test.cfengine.com	Failsafe.cf 2.1.0 (agent-0) Promises.cf 2.1.0 (agent-0)	100	0	0	September 29, 2011 15:21:26 (GMT+2)

Figure: Compliance summary report

## 1.6 File change log report

File change monitoring is about detecting when file information on a computer system changes. Awareness of changes is often considered a major part of management, especially if they are unexpected or inadvertent (expected changes are usually not a problem). With CFEngine you can either set a general tripwire to detect all changes, or, in the case of the 'File change log', monitor specific files with changes promises. The report gives you relative detail of change as it presents the name of files that have been changed, when they were changed and on what host they were changed.



The file change log query can filter by (patterns in) file name and host class (i.e. the class/context of a host). Leaving the fields blank will result in a report listing changes detected on all monitored hosts and and policies.

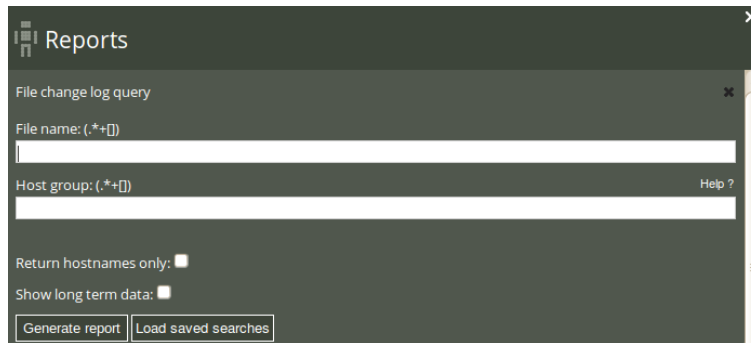


Figure: File change log query

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents the host names ('Host'), name of the file where a change was detected ('File'), and time stamp of change detection ('Change detected at'). You can add your personal note in the right column, documenting any thoughts or issues that you might have about the query result.

#### File change log

[PDF](#)
[✉](#)
[Select host](#)
[Select report](#)
[Save this search](#)
[New search](#)

Total results found: 963

HOST	FILE	CHANGE DETECTED AT	NOTE
ubuntu1.test.cfengine.com	!! File /var/cfengine/inputs /OrionCloud /Orion_Cloud_Pack_Demo.png was not in sha512 database - new file found	September 29, 2011 15:15:31 (GMT+2)	<a href="#">notes</a>
ubuntu1.test.cfengine.com	!! File /var/cfengine/inputs /OrionCloud/.svn/entries was not in sha512 database - new file found	September 29, 2011 15:15:31 (GMT+2)	<a href="#">notes</a>
ubuntu1.test.cfengine.com	!! File /var/cfengine/inputs /OrionCloud/.svn/text-base/c_cpp_env.cf.svn-base was not in sha512 database - new file found	September 29, 2011 15:15:31 (GMT+2)	<a href="#">notes</a>

Figure: File change log report

### 1.7 File change diffs report

A diff is a file comparison utility that outputs the differences between two files. It is typically used to show the changes between one version of a file and a former version of the same file. Diff displays the

changes made per line for text files. Once a file change has been identified, for instance as seen in the file change log, you can browse the details of that change in a file change diff report.

The file change diff query can filter by (pattern in) file name, (pattern in) diff content, and host class (i.e. the class/context of a host). Leaving the fields blank will result in a report listing changes detected on all monitored hosts and and policies.



The screenshot shows a web interface titled 'Reports'. It contains a search form for 'File change diffs query'. The form has three input fields: 'File name: (\*.\*)', 'Match content: (\*.\*)', and 'Host group: (\*.\*)'. Below these fields are two checkboxes: 'Return hostnames only:' and 'Show long term data:'. At the bottom of the form are two buttons: 'Generate report' and 'Load saved searches'.

Figure: File change diffs query

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of hosts that suit the query criteria entered above. The result presents the host names ('Host'), name of the file where a change was detected ('File'), the time stamp of change detection ('Change detected at'), and the actual diff (whether a line was added or deleted, the line number, and the content of the change; 'Change added (+), deleted (-); Line no; Content ').

#### File change diffs

PDF  [Select host](#) [Select report](#) [Save this search](#) [New search](#)

Total results found : 6

HOST	FILE	CHANGE DETECTED AT	CHANGE ADDED(+), DELETED(-), LINE NO, CONTENT												
hub.test.cfengine.com	/tmp/test	September 26, 2011 14:35:28 (GMT+2)	<table border="1"> <tr><td>-</td><td>1</td><td>hello</td></tr> <tr><td>-</td><td>3</td><td>bishwa</td></tr> <tr><td>-</td><td>5</td><td>next line</td></tr> <tr><td>+</td><td>1</td><td>test</td></tr> </table>	-	1	hello	-	3	bishwa	-	5	next line	+	1	test
-	1	hello													
-	3	bishwa													
-	5	next line													
+	1	test													
policy.test.cfengine.com	/etc/group	September 26, 2011 13:20:59 (GMT+2)	<table border="1"> <tr><td>+</td><td>52</td><td>ssl-cert: x:112:</td></tr> </table>	+	52	ssl-cert: x:112:									
+	52	ssl-cert: x:112:													
hub.test.cfengine.com	/tmp/test	September 26, 2011 10:11:04 (GMT+2)	<table border="1"> <tr><td>+</td><td>5</td><td>next line</td></tr> </table>	+	5	next line									
+	5	next line													

Figure: File change diffs report

## 1.8 Last saw hosts report

Sometimes it is not possible to connect to a machine under CFEngine's management, either due to network errors or temporary lack of network entirely (for instance on ships at sea or submarines). CFEngine 3 Nova's Mission Portal monitors all connections, incoming and outgoing, between all managed hosts, and creates a log of when neighbouring hosts were last observed online. This information is used to set the host availability status and, through analysis of the connection history, give an idea of the regularity of connections between hosts.

The Last saw hosts report is useful for checking the communication pattern between managed hosts and when they last were in touch with each other. Click on **Last saw hosts** in the Report finder to open a query window as described previously.

Figure: Last saw hosts query

The Last saw hosts query can filter by (patterns in) remote host name, remote host IP address, remote host key, minimum hours ago (since the last connection was made), and host class (i.e. the class/context of a host). Leaving the fields blank will result in a report listing all connections made to and from the managed machines (including the hub).

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of all communication that suits the query criteria entered above. Every connection is logged on the concerned nodes as incoming (Initiated by them) or outgoing (Initiated by us), the same connection will therefore appear twice in the report (once for each node). The results are presented in the following column format: 'Host' (host name), 'Initiated' (identifies whether the connection is incoming (by them (-)) or outgoing (by us (+))), 'Remote host name', 'Remote host IP address', 'Last seen' (timestamp of the connection), 'Hours ago' (interval between current time and Last seen), 'Avg interval' (average time between each connection), 'Uncertainty' (standard deviation of Average interval), and 'Remote host key' (identifying key of the remote host).

## Last saw hosts


[Select host](#) [Select report](#)
[Save this search](#)  
[New search](#)

Total results found: 19

HOST	INITIATED	REMOTE HOST NAME	REMOTE IP ADDRESS	LAST SEEN	HOURS AGO	AVG INTERVAL	UNCERTAINTY	REMOTE HOST KEY
policy.test.cfengine.com	by them (-)	hub.test.cfengine.com	10.0.0.29	September 29, 2011 15:31:40 (GMT+2)	0	0.09	0.01	SHA=bd6dfcc28b1a7be234a68e3fe77e3c199e68fc28f400de0f94eadf697ca213df
policy.test.cfengine.com	by us (+)	hub.test.cfengine.com	10.0.0.29	September 29, 2011 15:31:22 (GMT+2)	0.01	0.03	0.05	SHA=bd6dfcc28b1a7be234a68e3fe77e3c199e68fc28f400de0f94eadf697ca213df

Figure: Last saw hosts report

## 1.9 Patches available report

Software packaging is a core paradigm in operating system release management today, and CFEngine supports a generic approach to integration with native operating support for packaging. Package promises allow CFEngine to make promises the state of software packages conditionally, given the assumption that a native package manager will perform the actual manipulations. Since no agent can make unconditional promises about another, this is the best that can be achieved.

Some package systems also support the idea of patches. These might be formally different objects to packages; a patch might contain material for several packages and be numbered differently. When you select patching-policy, the package name can be a regular expression that will match possible patch names, otherwise identifying specific patches can be cumbersome.

The patches available report is useful to get an overview of patches claimed to be available by the local package manager. Click on **Patches available** in the 'Reports finder' to open a query window:

Figure: Patches available query

The Patches available query can filter by (patterns in) package name, package version, package architecture, and host class. Leaving the fields blank will result in a report listing all patches that can be installed on the system.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of patches that suit the query criteria entered above. The report presents the following columns: 'Host' (host name), 'Name' (name of the package/patch), 'Version' (patch version), and 'Architecture'.

#### Patches available

PDF  [Select host](#) [Select report](#) [Save this search](#)  
[New search](#)

Total results found: 185

HOST	NAME	VERSION	ARCHITECTURE
opensuse11-1.pilot.cfengine.com	Name	Version	default
centos5-1.pilot.cfengine.com	NetworkManager	1:0.7.0-13.el5	i386
centos5-1.pilot.cfengine.com	NetworkManager	1:0.7.0-13.el5	x86_64
centos5-1.pilot.cfengine.com	NetworkManager-glib	1:0.7.0-13.el5	i386
centos5-1.pilot.cfengine.com	NetworkManager-glib	1:0.7.0-13.el5	x86_64
opensuse11-1.pilot.cfengine.com	SuSEfirewall2	4330	default

Figure: Patches available report

### 1.10 Patch status report

Patch management can be a delicate business: sometimes a patch can cause new problems, or perhaps even more problems than it fixes. IT managers therefore often like to be in control of what patches are applied to a system. The Patch status report gives system administrators a complete overview of applied patches according to the local package manager, and, in conjunction with the patches available report, allows them to consciously decide which patches to apply or not.

Click on **Patch status** in the 'Reports finder' to open a query window:



Reports

Patch status

Package name: (\*+[])

Package version: (\*+[])

Package architecture: (\*+[])

Host group: (\*+[]) [Help ?](#)

Return hostnames only:


[Generate report](#) [Load saved searches](#)

Figure: Patch status query

The Patch status query can filter by (patterns in) package name, package version, package architecture, and host class. Leaving the fields blank will result in a report listing all patches applied to the system.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of patches that suit the query criteria entered above. The report presents results in the same format as the Patches available report: 'Host' (host name), 'Name' (name of the package/patch), 'Version' (patch version), and 'Architecture'.

#### Patch status

PDF  [Select host](#) [Select report](#) [Save this search](#)  
[New search](#)

Total results found: 298

HOST	NAME	VERSION	ARCHITECTURE
centos5-1.pilot.cfengine.com	*	base:	c
centos5-1.pilot.cfengine.com	*	extras:	c
centos5-1.pilot.cfengine.com	*	updates:	c
opensuse11-1.pilot.cfengine.com	ImageMagick	5132	default
opensuse11-1.pilot.cfengine.com	Mesa	4546	default
opensuse11-1.pilot.cfengine.com	ModemManager	4453	default
opensuse11-1.pilot.cfengine.com	MozillaFirefox	4195	default

Figure: Patch status report

## 1.11 Performance report

CFEngine 3 Nova uses several monitoring probes to reflect on general system performance<sup>1</sup>. One probe looks at the time it takes to execute selected promises; results are summarized in the 'Performance report'. The user can thus evaluate which parts of a policy put charge on the system in terms of time spent completing a task. Longer tasks, such as command execution and file copying, are measured by default, but other tasks have to be measured explicitly by stating so in a policy. Note however that most promises made in CFEngine are executed so fast we are not able to measure the time it takes to complete them.

Click on **Performance** in the 'Reports finder' to open a query window:



Figure: Performance query

<sup>1</sup> See also section on Vital signs in the CFEngine 3 Nova Owner's Manual.

The Performance query can filter by (patterns in) job name and host class. Leaving the fields blank will result in a report listing the performance of all monitored jobs.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of events that suit the query criteria entered above. 'Host' (host name), 'Event' (job name), 'Last time' (most recent performance value, i.e. the time it took to complete the job), 'Avg time' (average of all Last time), 'Uncertainty' (expressed as one standard deviation of 'Avg time'), and 'Last performed' (timestamp of last execution). You can add your personal note in the right column, documenting any thoughts or issues that you might have about the query result.

#### Performance

PDF  [Select host](#) [Select report](#) [Save this search](#) [New search](#)

Total results found: 54

HOST	EVENT	LAST TIME	AVG TIME	UNCERTAINTY	LAST PERFORMED	NOTE
ubuntu10-1.pilot.cfengine.com	Copy(localhost/var/cfengine/bin/cf-agent > /var/cfengine/bin/cf-twin)	0	0	0	October 31, 2011 07:21:42 (GMT+1)	<a href="#">notes</a>
ubuntu10-1.pilot.cfengine.com	Copy(localhost/var/cfengine/lib > /var/cfengine/lib-twin)	0	0	0	October 31, 2011 07:21:42 (GMT+1)	<a href="#">notes</a>
ubuntu10-1.pilot.cfengine.com	Copy(localhost/var/cfengine/bin > /usr/local/sbin)	0	0	0	October 31, 2011 07:21:40 (GMT+1)	<a href="#">notes</a>

Figure: Performance report

## 1.12 Promises repaired log report

The Status room in the Nova Mission Portal gives an overview of the general status of your system, including six hour summaries of promises kept, repaired, and not kept from the last week. The Promises repaired log is useful to get a complete overview of the history of promises repaired, including execution order and events that are more than a week old. Click on **Promises repaired log** in the 'Reports finder' to open a query window:

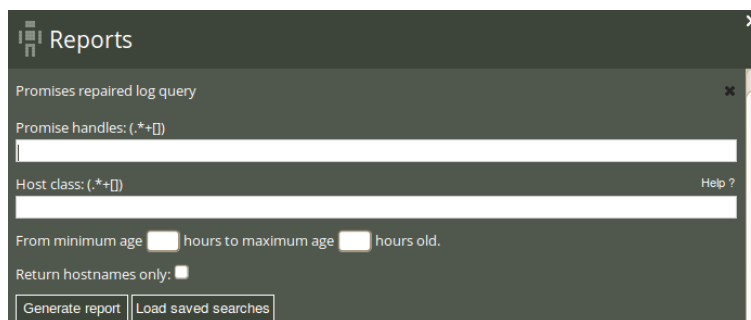



Figure: Promises repaired log query

The Promise repaired log query can filter by (patterns in) promise handles, host class (i.e. the class/context of a host), and a desired time interval. Leaving the fields blank will result in a report listing all promises that were repaired and the time of occurrence.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of promises that suit the query criteria entered above. The results are presented as 'Host' (host name), 'Promise handle' (identifier of the promise), 'Report' (what was repaired), and 'Time' (timestamp of the repair action). You can add your personal note in the right column, documenting any thoughts or issues that you might have about the query result.

#### Promises repaired log

PDF  [Select host](#) [Select report](#) [Save this search](#)  
[New search](#)

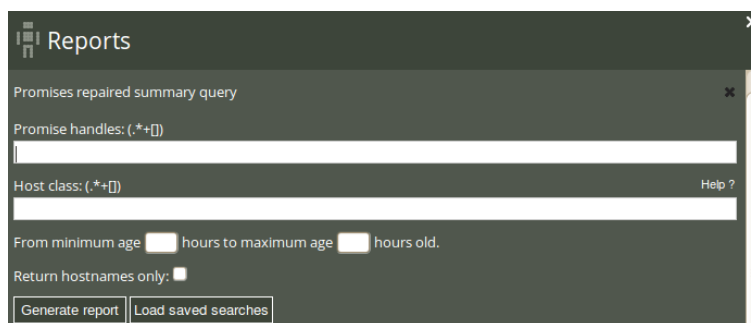
Total results found: 9057

HOST	PROMISE HANDLE	REPORT	TIME	NOTE
hub.test.cfengine.com	<a href="#">garbage_collection_files_tidy_outputs</a>	-> Deleted file /var/cfengine/outputs /cf_cf009lin_cfengine_com _1316789437_Fri_Sep_23_16_50_37_2011_586ef700	September 30, 2011 16:56:05 (GMT+2)	<a href="#">notes</a>
hub.test.cfengine.com	<a href="#">cfengine_policysrv_packages_install_package_debian</a>	-> Finished command related to promiser "php5-json" -- succeeded	September 30, 2011 16:56:00 (GMT+2)	<a href="#">notes</a>
hub.test.cfengine.com	<a href="#">garbage_collection_files_tidy_outputs</a>	-> Deleted file /var/cfengine/outputs /cf_cf009lin_cfengine_com _1316789125_Fri_Sep_23_16_45_25_2011_586ef700	September 30, 2011 16:50:53 (GMT+2)	<a href="#">notes</a>

Figure: Promises repaired log report

### 1.13 Promises repaired summary report

If the Promises repaired log is too detailed for your needs, the Promises repaired summary report eliminates the time stamp of the promises repaired and presents a cumulative summary of promises repaired, i.e. the total number times a promise has been repaired. Click on **Promises repaired summary** in the 'Reports finder' to open a query window:



Reports

Promises repaired summary query

Promise handles: (\*.\*)

Host class: (\*.\*) [Help ?](#)

From minimum age  hours to maximum age  hours old.

Return hostnames only:


Figure: Promises repaired summary query



The Promise repaired summary query can filter by (patterns in) promise handles, host class (i.e. the class/context of a host), and a desired time interval. Leaving the fields blank will result in a report listing all promises that were repaired and their cumulative number of occurrences.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of promises that suit the query criteria entered above. The results are presented as 'Promise handle' (identifier of the promise), 'Report' (what was repaired), and 'Occurrences' (number of occurrences of repair).

#### Promises repaired summary

PDF  [Select host](#) [Select report](#) [Save this search](#)  
[New search](#)

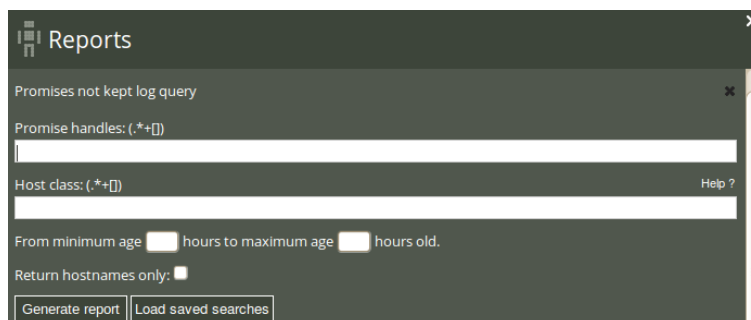
Total results found: 42

PROMISE HANDLE	REPORT	OCCURRENCES
<a href="#">cfengine_test_commands_repair</a>	-> Finished command related to promiser "/bin/touch /tmp/file_991" -- succeeded	6496
<a href="#">cfengine_policysrv_packages_install_package_debian</a>	-> Finished command related to promiser "php5-json" -- succeeded	1375
<a href="#">garbage_collection_files_tidy_outputs</a>	-> Deleted file /var/cfengine/outputs /cf_cf009lin_cfengine_com_1316789437_Fri_Se p_23_16_50_37_2011_586ef700	966
<a href="#">promise_file_change_cf_53</a>	!! File /var/cfengine/Inputs/old/cfengine.cf was not in sha512 database - new file found	26

Figure: Promises repaired summary report

### 1.14 Promises not kept log report

The Status room in the Nova Mission Portal gives an overview of the general status of your system, including six hour summaries of promises kept, repaired, and not kept from the last week. The Promises not kept log is useful to get a complete overview of the history of promises not kept, including execution order and events that are more than a week old. Click on **Promises not kept log** in the 'Reports finder' to open a query window:



**Reports**

Promises not kept log query

Promise handles: (\*.\*)

Host class: (\*.\*) [Help ?](#)

From minimum age  hours to maximum age  hours old.


Return hostnames only:

Figure: Promises not kept log query

The Promises not kept log query can filter by (patterns in) promise handles, host class (i.e. the class/context of a host), and a desired time interval. Leaving the fields blank will result in a report listing all promises that were not kept and the time of occurrence.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of promises that suit the query criteria entered above. The results are presented as 'Host' (host name), 'Promise handle' (identifier of the promise), 'Report' (what was not kept), and 'Time' (time stamp of the event). You can add your personal note in the right column, documenting any thoughts or issues that you might have about the query result.

#### Promises not kept log

PDF  [Select host](#) [Select report](#) [Save this search](#) [New search](#)

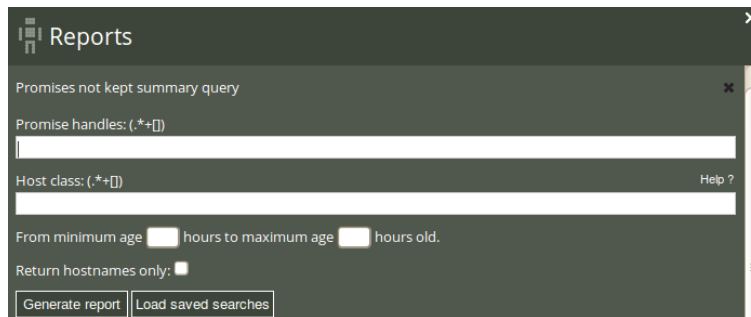
Total results found: 7655

HOST	PROMISE HANDLE	REPORT	TIME	NOTE
windows2003-1.pilot.cfengine.com	<a href="#">cdp_acl_file_c__WINDOWS_system32_drivers_etc_hosts_Windows_Server_2003__Hr09</a>	!! Syntax error in access control list for "c:\WINDOWS\system32\drivers\etc\hosts"	October 30, 2011 12:11:28 (GMT+1)	<a href="#">notes</a>
ubuntu10-1.pilot.cfengine.com	<a href="#">cdp_chg_etc_shadow_linux_solarisx86</a>	!! Hash for file "/etc/shadow" changed	October 30, 2011 12:11:25 (GMT+1)	<a href="#">notes</a>
opensuse11-1.pilot.cfengine.com	<a href="#">cdp_chg_etc_shadow_linux_solarisx86</a>	!! Hash for file "/etc/shadow" changed	October 30, 2011 12:11:20 (GMT+1)	<a href="#">notes</a>

Figure: Promises not kept log report

### 1.15 Promises not kept summary report

If the Promises not kept log is too detailed for your needs, the Promises not kept summary report eliminates the time stamp of the promises repaired and presents a cumulative summary of promises repaired, i.e. the total number times a promise was not kept. Click on **Promises not kept summary** in the 'Reports finder' to open a query window:



Reports

Promises not kept summary query

Promise handles: (\*+[])

Host class: (\*+[]) [Help ?](#)

From minimum age  hours to maximum age  hours old.

Return hostnames only:

[Generate report](#) [Load saved searches](#)

Figure: Promises not kept summary query

The Promise not kept summary query can filter by (patterns in) promise handles, host class (i.e. the class/context of a host), and a desired time interval. Leaving the fields blank will result in a report listing all promises that were not kept and their cumulative number of occurrences.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of promises that suit the query criteria entered above. The results are presented as 'Promise handle', 'Report' (what was not kept), and 'Occurrences' (the number of times the promise was not kept).

## Promises not kept summary


[Select host](#) [Select report](#)
[Save this search](#)  
[New search](#)

Total results found: 9

PROMISE HANDLE	REPORT	OCCURRENCES
<a href="#">internal_promise</a>	Protocol transaction broken off (1)	5398
<a href="#">cfengine_correct_cftwin_files_libtwin</a>	Can't stat /var/cfengine/lib in files.copyfrom promise	3540
<a href="#">cfengine_php_mod_files_mongo_so_ubuntu10_64</a>	Can't stat /var/cfengine/software_updates/mongo/20090626/64/mongo.so in files.copyfrom promise	2035
<a href="#">cfengine_php_mod_files_svn_so_ubuntu10_64</a>	Can't stat /var/cfengine/software_updates/svn/20090626/64/svn.so in files.copyfrom promise	2035

Figure: Promises not kept summary report

## 1.16 Setuid/gid root programs report


setuid and setgid (short for "set user ID upon execution" and "set group ID upon execution", respectively) are Unix access right flags that allow users to run an executable with the permissions of the executable's owner or group. They are often used to allow users on a computer system to run programs with temporarily elevated privileges in order to perform a specific task. The 'Setuid/gid root programs report' is useful to get an overview of what processes have been elevated to root privileges and potentially uncover security issues.

Click on **Setuid/gid root programs** in the 'Reports finder' to open a query window:

Figure: Setuid/gid root programs query

The Setuid/gid root programs query can filter by (patterns in) file name or host class. Leaving the fields blank will result in a report listing all hosts and files that have their permissions elevated to root.

## Setuid/gid root programs

PDF  [Select host](#) [Select report](#) [Save this search](#)  
[New search](#)

Total results found: 107

HOST	FILE
solaris10-1.pilot.cfengine.com	/usr/bin/write
solaris10-1.pilot.cfengine.com	/usr/bin/volrmount
solaris10-1.pilot.cfengine.com	/usr/bin/tsojdslabel
solaris10-1.pilot.cfengine.com	/usr/bin/su
solaris10-1.pilot.cfengine.com	/usr/bin/rsh

Figure: Setuid/gid root programs report

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of promises that suit the query criteria entered above. The results are presented as host name and files that have their permissions elevated to root.

## 1.17 Software installed report

The 'Software installed report' will list the software packages claimed to be installed according to the local package manager. Click on **Software installed** in the 'Reports finder' to open a query window:



Reports

Software installed query

Name:

Version:

Architecture:

Host class: (\*.\*) [Help ?](#)

Return hostnames only:

Figure: Software installed query

The Software installed query can filter by (patterns in) software name, version, architecture, or host class. Leaving the fields blank will result in a report listing all hosts and software installed on the system.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview that suits the query criteria entered above. The results are presented as 'Host' (host name), 'Name' (of software package), 'Version' (of software package), and 'Architecture' (of machine on which software runs).

## Software installed


[Select host](#) [Select report](#)
[Save this search](#)  
[New search](#)

Total results found: 1509

HOST	NAME	VERSION	ARCHITECTURE
opensuse11-1.pilot.cfengine.com	ConsoleKit	0.4.3-6.1	x86_64
centos5-1.pilot.cfengine.com	GConf2	2.14.0-9.el5	x86_64
centos5-1.pilot.cfengine.com	MAKEDEV	3.23-1.2	x86_64
centos5-1.pilot.cfengine.com	NetworkManager	1:0.7.0-10.el5_5.2	i386
centos5-1.pilot.cfengine.com	NetworkManager	1:0.7.0-10.el5_5.2	x86_64
centos5-1.pilot.cfengine.com	NetworkManager-glib	1:0.7.0-10.el5_5.2	i386

Figure: Software installed report

## 1.18 Variables report

The 'Variables report' is useful for tracking your variables and checking their values, for instance to see if they behave in the expected manner. Click on **Variables** in the 'Reports finder' to open a query window:

Figure: Variables query

The Variables query can filter by (patterns in) scope (bundle where the variable is used), Lvalue (name of variable), Rvalue (content of variable), type, or host class. Leaving the fields blank will result in a report listing all variables that were last observed on the system.

Once you have clicked **Generate report**, CFEngine 3 Nova will list an overview of variables that suit the query criteria entered above. The results are presented in table form/blocks of scope (i.e. in which bundle the variables appear) with the following column format: 'Host' (name of host where the variable is defined), 'Type' (type of the variable, 'Name', and 'Value'.

## Variables



[Select host](#)
[Select report](#)

[Save this search](#)  
[New search](#)

**Total : 3222 variables found**

**bundle update\_bins: 6 variables**

HOST	TYPE	NAME	VALUE
solaris10-1.pilot.cfengine.com	string	stop_signal	term
solaris10-1.pilot.cfengine.com	string	pkgarch	i86pc
solaris10-1.pilot.cfengine.com	string	novapkg	CFEcfengine-nova
solaris10-1.pilot.cfengine.com	string	master_software_location	/var/cfengine /master_software_updates
solaris10-1.pilot.cfengine.com	string	local_software_dir	/var/cfengine/software_updates /sunos_5.10_i86pc
solaris10-1.pilot.cfengine.com	string	communitypkg	CFEcfengine-community

Figure: Variables report

## 2 CDP reports

Content-Driven Policies (CDP) were introduced to make policy management easier. In contrast to policies written in the CFEngine language, they are composed of semi-colon separated fields in a text file that the user fills with content, like a spreadsheet or tabular file. Each line in the file is parsed and results in a specific type of promise being made. Reports based on data from CDP policies can be found in 'Engineering room': click the **Engineering** icon in the Mission Portal, then the **CDP reports** finder in the Engineering room:

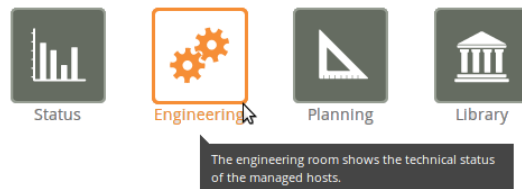


Figure: Go to Engineering

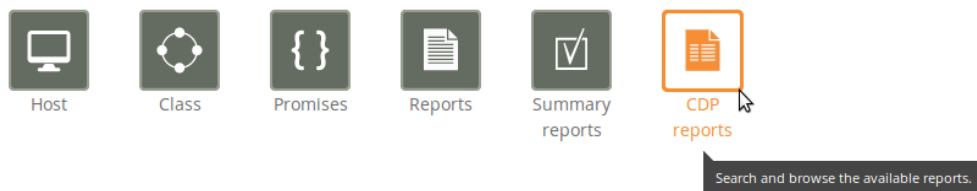


Figure: Go to CDP reports finder

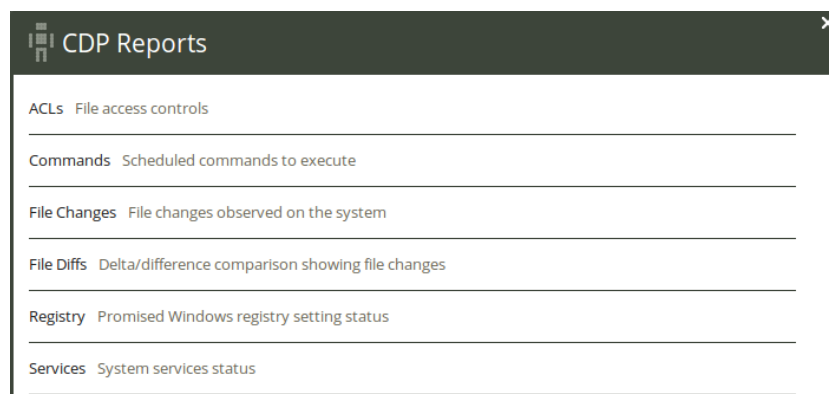


Figure: CDP reports finder

We will now go through the different CDP reports and their corresponding input files.

### 2.1 ACLs report: File access controls

An access control list (ACL) is a list of permissions attached to an object. An ACL specifies which users or system processes are granted access to objects, as well as what operations are allowed on given objects. Each entry in a typical ACL specifies a subject and an operation. For instance, if a file has an ACL that contains (Alice, delete), this would give Alice permission to delete the file.

Click on **ACLs** in the 'CDP Reports finder' to access the ACLs CDP report:

HOST	PATH	PERMISSION (ACL)	OWNER	ACTION	CLASS EXPRESSION	STATE	LAST CHECKED
windows2008-1.pilot.cfengine.com	c:\Windows\System32\drivers\etc\hosts	'user:Administrator:rw','user:SYSTEM:rw','user:Guest:rw'	Administrator	fix	Windows_Server_2008_R2.Hr11	Not Compliant	Thu 15 Dec 2011
windows2008-1.pilot.cfengine.com	c:\Windows\System32\drivers\etc\hosts	'user:Administrator:rw','user:SYSTEM:rw','user:Guest:rw'	Administrator	fix	Windows_Server_2008_R2.Hr11	Compliant	Wed 14 Dec 2011

Figure: ACLs report (bug in presentation)

The report lists an overview of host name ('Host'), path of the affected object ('Path'), the permission setting ('Permission (ACL)'), owner of the affected object ('Owner'), action to execute on the object ('Action'), the context in which the promise was made ('Class expression'), state of compliance ('State'), and the time the promise was last checked ('Last checked').

The default CFEngine 3 Nova ACLs policies allow you to set permissions to directories and files using two different input files ('acl\_directory\_list.txt' and 'acl\_file\_list.txt', respectively). We will limit ourselves to look at one of these in the following example as they are conceptually identical.

The file 'acl\_file\_list.txt' can be found under the 'cdp\_inputs' catalog in the policy editor, click it to open:



Figure: Open the ACLs input file



The content of the file looks like this (lines have been split and indented for presentability):

```
#
# ACLs On Files
#
# FORMAT:  path;entity_type1:entity_name1:perms1,
           entity_type2:entity_name2:perms2,...;owner;action;class_expression
#
# EXAMPLE: C:\tmp;user:Administrator:rxw,user:SYSTEM:r;
           Administrator;fix;windows
#

# Windows 2003
c:\WINDOWS\system32\drivers\etc\hosts;user:Administrator:rw,user:SYSTEM:rw,
  user:Guest:r;SYSTEM;fix;Windows_Server_2003.!Hr09
c:\WINDOWS\system32\drivers\etc\hosts;user:Administrator:rw,user:SYSTEM:rw,
  user:Guest:rw;SYSTEM;fix;Windows_Server_2003.Hr09
# Windows 2008
c:\Windows\System32\drivers\etc\hosts;user:Administrator:rw,user:SYSTEM:rw,
  user:Guest:r;SYSTEM;fix;Windows_Server_2008_R2.!Hr11
c:\Windows\System32\drivers\etc\hosts;user:Administrator:rw,user:SYSTEM:rw,
  user:Guest:rw;SYSTEM;fix;Windows_Server_2008_R2.Hr11
```

We need to look at the header of the file to understand its structure. We saw in the CDP reports introduction that the input consisted of lines containing semi colon separated fields, so anything with a ';' before or after it is a field entry. The structure of these fields are explained in the FORMAT section of the file header, here we have:

```
# FORMAT:  path;entity_type1:entity_name1:perms1,
           entity_type2:entity_name2:perms2,...;owner;action;class_expression
```

Splitting this up into separate fields:

*path*            Path of file to set permissions on.

*entity\_type1:entity\_name1:perms1*

This field defines the permissions ('perms1') that a user ('entity\_type1'), and member of the group ('entity\_name1'), has on the file defined in 'path'.

*entity\_type2:entity\_name2:perms2,...*

Same as entity\_type1:entity\_name1:perms1, but for different user, group, and permission settings.

*owner*            Defines the owner of the file defined in 'path'

*action*          Tells CFEngine what to do if the file permissions differ from what was defined in the ACLs policy. Can take the values 'fix' (set permissions as defined in ACLs policy), 'warn' (log and display a warning that the file permissions differ from what was defined in ACLs policy), and 'nop' (no operation; no log entry, but print a warning in command-line interface).

*class\_expression*

Context in which the permissions are set, i.e. a class expression (boolean) that needs to be fulfilled for the permissions to be set.

## 2.2 Commands report: Scheduled commands to execute

You may use the Commands CDP to schedule script execution on specific hosts. The Commands CDP uses a combination of class expressions to set the context (i.e. time and place) of execution.

Click on **Commands** in the 'CDP Reports finder' to access the Commands CDP report:

### Commands

HOST	COMMAND	FAILCLASS	ACTION	CLASS EXPRESSION	STATE	LAST CHECKED
centos5-1.pilot.cfengine.com	/bin/sleep 5	sleep_failed_solaris	fix	linux	Repaired	Thu Nov 3 13:50:58 2011
opensuse11-1.pilot.cfengine.com	/bin/sleep 5	sleep_failed_solaris	fix	linux	Repaired	Thu Nov 3 13:51:56 2011
policyhub-1.pilot.cfengine.com	/bin/sleep 5	sleep_failed_solaris	fix	linux	Repaired	Thu Nov 3 13:55:55 2011
ubuntu10-1.pilot.cfengine.com	/bin/sleep 5	sleep_failed_solaris	fix	linux	Repaired	Thu Nov 3 13:51:38 2011
windows2003-1.pilot.cfengine.com	c:\windows\system32\cmd.exe /c "echo hello failed > c:\reportfile.txt"	report_failed	fix	hello_failed.windows	Repaired	Thu Nov 3 11:55:52 2011

Figure: Commands report

The report lists an overview of host name ('Host'), the command to execute ('Command'), the class to define if execution fails ('Failclass'), action to execute if there is an error ('Action'; see explanation of the CDP input file below for possible values), the context in which the promise was made ('Class expression'), state of compliance ('State'), and the time the promise was last checked ('Last checked').

The Commands CDP input file, 'command\_list.txt', can be found in the left menu in the 'cdp\_inputs' catalog. The content looks like this (lines have been split and indented for presentability):

```
#
# Command Execution
#
# FORMAT:  command_path;on_error_define_class;action;class_expression
#
# EXAMPLE: c:\windows\system32\cmd.exe /c "echo hello";hello_failed;fix;
#          DomainController
#
# NOTE:    You may use this Content-Driven Policy to schedule script
#          execution on a class of hosts by using a combination of
#          host and time classes in class_expression, e.g. set
#          class_expression to "windows.Tuesday.Hr10.Min30_35".
#
```

```

c:\windows\system32\cmd.exe /c "echo hello";hello_failed;fix;windows.Hr11
c:\windows\system32\cmd.exe /c "echo hello";hello_failed;fix;windows.!Hr11
c:\windows\system32\cmd.exe /c "echo hello failed > c:\reportfile.txt";
  report_failed;fix;hello_failed.windows
c:\windows\system32\cmd.exe /c "echo hello succeeded > c:\reportfile.txt";
  report_failed;fix;!hello_failed.windows
/usr/bin/sleep 5;sleep_failed_solaris;fix;solaris
/bin/sleep 5;sleep_failed_solaris;fix;linux

```

Again, we need to look at the header of the file to understand its structure:

```
# FORMAT:  command_path;on_error_define_class;action;class_expression
```

Separating the fields:

*command\_path*

Path of command to execute.

*on\_error\_define\_class*

Class to define if there is an error in command execution.

*action*

Tells CFEngine what to do if there is an error in command execution. Can take the values 'fix' (attempt to re-execute the command), 'warn' (log and display a warning that the command could not be executed), and 'nop' (no operation; no log entry, but print a warning in command-line interface).

*class\_expression*

Context in which the command is to be executed, i.e. a class expression (boolean) that needs to be fulfilled for the command to take place. In the above example (`windows.Tuesday.Hr10.Min30_35`) the command will only be executed on Windows machines on Tuesdays between 10.30am and 10.35am.

### 2.3 File changes report: File changes observed on the system

We saw that awareness of changes often is considered a major part of infrastructure management in the walk-through of the CFEngine 3 Nova standard reports. The file changes CDP policy differs slightly from the Files changes log report in that it will restore the original file upon detecting a change and report whether the file remains compliant or not.

Click on **File changes** in the 'CDP Reports finder' to access the File changes CDP report:

#### File Changes

HOST	PATH	CLASS EXPRESSION	LAST CHANGE DETECTED	STATE	LAST CHECKED
centos5-1.pilot.cfengine.com	/etc/shadow	linux solarisx86	(never changed)	Compliant	Thu Nov 3 13:56:31 2011
opensuse11-1.pilot.cfengine.com	/etc/shadow	linux solarisx86	(never changed)	Compliant	Thu Nov 3 13:56:45 2011
policyhub-1.pilot.cfengine.com	/etc/shadow	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:00:56 2011
solaris10-1.pilot.cfengine.com	/etc/shadow	linux solarisx86	(never changed)	Compliant	Thu Nov 3 13:56:26 2011
ubuntu10-1.pilot.cfengine.com	/etc/shadow	linux solarisx86	(never changed)	Compliant	Thu Nov 3 13:56:31 2011

Figure: File changes report

The report lists an overview of host name ('Host'), the path of the concerned file ('Path'), the context in which the promise was made ('Class expression'), time stamp of when a change was detected ('Last Change Detected'), state of compliance ('State'), and the time the promise was last checked ('Last checked').

The File changes CDP input file, 'file\_change\_list.txt', can be found in the left menu in the 'cdp\_inputs' catalog. The content looks like this:

```
#
# File Changes Detection and Revert
#
# FORMAT:   file_path;class_expression
#
# EXAMPLE:  C:\pwd.txt;windows
#
# NOTE:     The file is always restored on change, and change
#           reports will be generated.
#           Use file_diff Content-Driven Policy to allow changes.
#
/etc/shadow;linux|solarisx86
```

Looking at the header of the file:

```
# FORMAT:   file_path;class_expression
```

Separating the fields:

*file\_path* Path of file to monitor and repair changes on.

*class\_expression*

Context in which the change detection/repair is to be executed, i.e. a class expression (boolean) that needs to be fulfilled for the event to take place.

## 2.4 File diffs report: Delta/difference comparison showing file changes

The file diff CDP policy does the same as the File change CDP policy, except that it does not repair the file to original state if a change is detected. Click on **File diffs** in the 'CDP Reports finder' to access the File diffs report:

## File Diffs

HOST	PATH	CLASS EXPRESSION	LAST CHANGE DETECTED	STATE	LAST CHECKED
centos5-1.pilot.cfengine.com	/etc/group	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:05:47 2011
opensuse11-1.pilot.cfengine.com	/etc/group	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:06:33 2011
policyhub-1.pilot.cfengine.com	/etc/group	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:06:03 2011
solaris10-1.pilot.cfengine.com	/etc/group	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:06:13 2011
ubuntu10-1.pilot.cfengine.com	/etc/group	linux solarisx86	(never changed)	Compliant	Thu Nov 3 14:06:29 2011

Figure: File diffs report

The report lists an overview of host name ('Host'), the path of the concerned file ('Path'), the context in which the promise was made ('Class expression'), time stamp of when a change was detected ('Last Change Detected'), state of compliance ('State'), and the time the promise was last checked ('Last checked').

The File diff CDP input file, 'file\_diff\_list.txt', can be found in the left menu in the 'cdp\_inputs' catalog. The content looks like this:

```
#
# File Difference Reporting
#
# FORMAT:   file_path;class_expression
#
# EXAMPLE:  C:\users.txt;windows
#
# NOTE:     The file is always allowed to change.
#           Use file_change Content-Driven Policies to revert a
#           changed file. Detailed change reports will
#           be generated.
#
/etc/group;linux|solarisx86
```

Looking at the header of the file:

```
# FORMAT:   file_path;class_expression
```

Separating the fields:

*file\_path* Path of file to monitor and warn about changes on.

*class\_expression*

Context in which the change detection/warning is to be executed, i.e. a class expression (boolean) that needs to be fulfilled for the event to take place.

## 2.5 Registry report: Promised Windows registry setting status

The Windows Registry is a hierarchical database that stores configuration settings and options on Microsoft Windows operating systems. It contains settings for low-level operating system components as well as the applications running on the platform. Registry keys are similar to folders: in addition to

values, each key can contain subkeys, which may contain further subkeys, and so on. Keys are referenced with a syntax similar to Windows' path names, using backslashes to indicate levels of hierarchy. Each subkey has a mandatory name, which is a non-empty string that cannot contain any backslash, and whose letter case is insignificant.

Click on **Registry** in the 'CDP Reports finder' to access the Registry report:

Registry

HOST	KEY	VALUE	ACTION	CLASS EXPRESSION	STATE	LAST CHECKED
windows2003-1.pilot.cfengine.com	HKEY_CURRENT_USER\Control Panel\Desktop	ScreenSaveTimeOut,REG_SZ,1200	fix	windows.Hr11	Compliant	Fri Jan 6 11:59:22 2012
windows2003-1.pilot.cfengine.com	HKEY_CURRENT_USER\Control Panel\Desktop	ScreenSaverIsSecure,REG_SZ,0	fix	windows.Hr11	Compliant	Fri Jan 6 11:59:22 2012
windows2003-1.pilot.cfengine.com	HKEY_CURRENT_USER\Control Panel\Desktop	ScreenSaveTimeOut,REG_SZ,600	fix	windows.Hr11	Compliant	Fri Jan 6 14:29:22 2012
windows2003-1.pilot.cfengine.com	HKEY_CURRENT_USER\Control Panel\Desktop	ScreenSaverIsSecure,REG_SZ,1	fix	windows.Hr11	Compliant	Fri Jan 6 14:29:22 2012

Figure: Registry report

The report lists an overview of host name ('Host'), the key identifier ('Key'), the key value ('Value'), action to take if there is an error in the key ('Action'; see explanation of the CDP input file below for possible values), the context in which the promise was made ('Class expression'), the state of compliance ('State'), and the time the promise was last checked ('Last checked').

The Registry CDP input file, 'registry\_list.txt', can be found in the left menu in the 'cdp\_inputs' catalog. The content looks like this (lines have been split and indented for presentability):

```
#
# Windows Registry Management
#
# FORMAT:   key;name,type,data;action;class_expression
#
# EXAMPLE:  HKEY_CURRENT_USER\Control Panel\Desktop;ScreenSaverIsSecure,
            REG_SZ,1;fix;windows
#
# NOTE:     Currently, type must be REG_SZ (string).
#
HKEY_CURRENT_USER\Control Panel\Desktop;ScreenSaverIsSecure,REG_SZ,1;
    fix;windows.!Hr11
HKEY_CURRENT_USER\Control Panel\Desktop;ScreenSaverIsSecure,REG_SZ,0;
    fix;windows.Hr11
HKEY_CURRENT_USER\Control Panel\Desktop;ScreenSaveTimeOut,REG_SZ,600;
```

```
fix;windows.!Hr11
HKEY_CURRENT_USER\Control Panel\Desktop;ScreenSaveTimeOut,REG_SZ,1200;
fix;windows.Hr11
```

Looking at the header of the file:

```
# FORMAT: key;name,type,data;action;class_expression
```

Separating the fields:

*key* The path to the key in question.

*name,type,data*  
Name, type, and value of the key.

*action* Tells CFEngine what to do if there is a difference between the registry entry and the definition in the Registry CDP. Can take the values 'fix' (set the registry entry as defined in the Registry CDP), 'warn' (log and display a warning that there is a discrepancy between the registry entry and the Registry CDP), and 'nop' (no operation; no log entry, but print a warning in command-line interface).

*class\_expression*  
Context in which the promise is to be executed, i.e. a class expression (boolean) that needs to be fulfilled for the command to take place.

## 2.6 Services report: System service status

Services are programs that once started run continuously in the background. They perform specific functions which are designed not to require user intervention and are ready for input or monitor changes in your system and respond to them. For example, the Apache server has a daemon called httpd that listens on port 80 on your machine. When it receives a request for a page it sends the appropriate data back to the client machine.

With the three lines of semicolon separated fields, we ensure the correct status of three services on all our Windows machines and are given specialized reports on the outcome. The Content-Driven Policy services report is shown below. Click on **Services** in the 'CDP Reports finder' to access the Services report:

### Services

HOST	SERVICE NAME	RUNSTATUS	ACTION	CLASS EXPRESSION	STATE	LAST CHECKED
windows2003-1.pilot.cfengine.com	wuauerv	start	fix	windows.IHr10	Compliant	Thu Nov 3 14:00:52 2011
windows2003-1.pilot.cfengine.com	Dnscache	stop	fix	windows.Hr10	Compliant	Thu Nov 3 10:56:40 2011
windows2003-1.pilot.cfengine.com	wuauerv	stop	fix	windows.Hr10	Compliant	Thu Nov 3 10:56:40 2011
windows2003-1.pilot.cfengine.com	Dnscache	start	fix	windows.IHr10	Compliant	Thu Nov 3 14:00:52 2011
windows2008-1.pilot.cfengine.com	wuauerv	start	fix	windows.IHr10	Compliant	Thu Nov 3 14:00:53 2011
windows2008-1.pilot.cfengine.com	Dnscache	stop	fix	windows.Hr10	Compliant	Thu Nov 3 10:56:12 2011
windows2008-1.pilot.cfengine.com	wuauerv	stop	fix	windows.Hr10	Compliant	Thu Nov 3 10:56:12 2011
windows2008-1.pilot.cfengine.com	Dnscache	start	fix	windows.IHr10	Compliant	Thu Nov 3 14:00:53 2011

Figure: Services report

The report lists an overview of host name ('Host'), 'Service Name', the runstatus of the service ('Runstatus'), action to take if there is a difference from policy ('Action'; see explanation of the CDP input file below for possible values), the context in which the promise was made ('Class expression'), the state of compliance ('State'), and the time the promise was last checked ('Last checked').

The Registry CDP input file, 'service\_list.txt', can be found in the left menu in the 'cdp\_inputs' catalog. The content looks like this (lines have been split and indented for presentability):

```
#
# Windows Service Management
#
# FORMAT:  service_name;run_status;action;class_expression
#
# EXAMPLE: Dnscache;start;fix;windows
#
# NOTE:    Service name is not the "Display name"
#          -- see the properties of the service.
#          run_status can be start/stop/disable. If start then
#          the service is started, if disable then service is
#          stopped and "Startup type" is set to disable,
#          if stop, then service is stopped "Startup type" is left
#          unchanged.
#
Dnscache;stop;fix;windows.Hr10
wuauserv;stop;fix;windows.Hr10
Dnscache;start;fix;windows.!Hr10
wuauserv;start;fix;windows.!Hr10
```

Looking at the header of the file:

```
# FORMAT:  service_name;run_status;action;class_expression
```

Separating the fields:

*service\_name*

Name of the service (not necessarily the same as the display name, see properties of the service).

*run\_status*

The run status of the service defines whether it should be running or not. Can take the values 'start', 'stop', or 'disable' (will stop the service and change its startup mode to disable, i.e. will not be restarted upon reboot of the machine, for example).

*action*

Tells CFEngine what to do if there is a difference between the run status what has been defined in the Services CDP. Can take the values 'fix' (set the run status as defined in the Services CDP), 'warn' (log and display a warning that there is a discrepancy



between the run status and the Services CDP), and 'nop' (no operation; no log entry, but print a warning in command-line interface).

*class\_expression*

Context in which the promise is to be executed, i.e. a class expression (boolean) that needs to be fulfilled for the command to take place.