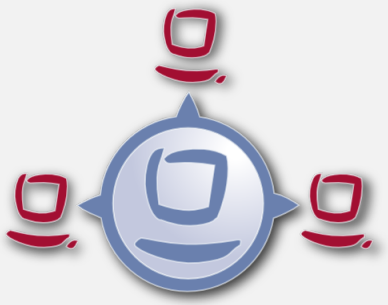

opsi Version 4.0.3 Release Notes

uib gmbh
Bonifaziusplatz 1b
55118 Mainz
Tel.:+49 6131 275610
www.uib.de
info@uib.de

Stand: 26.02.2013

Contents

Chapter 1

Copyright

Chapter 2

Overview of the new features

The opsi Service Release 4.0.3 comes with a whole bunch of new features and improvements. Here comes a survey:

- opsi extension: Install-on-shutdown
- opsi feature: Silent-Install
- Italian and Danish Localisation
- Support for the following distributions:
  - UCS 3.1
  - Ubuntu 12.10

The installation guide is to be found in the *opsi-getting-started* manual.

- Discontinuation at end of March 2013 for the support for:
  - CentOs 5
  - RedHat 5
  - OpenSuse 11.3 / 11.4
- opsi-client-agent
  - Extensively revised version
  - Open Source release of the opsi-Feature: Dynamic Depot Selection
  - opsi extension: Install-on-shutdown
  - opsi feature: Silent-Install
  - Improved customizing
  - Adaptions for Windows 8 Support
  - Revised session handling: better identification of user accounts.
  - Improvements for the WAN-Support
  - Several fixes
- opsi server components
  - Improvements of the DHCP-Backend
  - New HostControlSafe-Backend
  - Generating the webservice SSL certificate via opsi-setup

- Windows Netboot products
  - Windows 8 / Server 2012 support
  - Improvement of the byAudit procedure
  - Configurable WinPE partition size
  - New property data_partition_create: shall a data partition be created, if there is disk space available (true/false; default: true).
  - UEFI support
  - Support for Softwareraid 1: new property *useRaid1* handles 2 existing disks the same way.
- opsi-winst:
  A lot of new features, which are described in detail further down.
- Announcing the change of the license type GPLv3 to AGPLv3:
  With the opsi 4.0.3 Release we will switch to AGPLv3.
  This change of the license type affects only those companies, that distribute opsi as part of a Closed Source product.
  Anybody in need of a license to sell opsi as part of a Closed Source solution may contact us.

For further details please refer to the opsi-v403-releasenotes and the revised opsi manuals.

Chapter 3

Discontinuation

In this chapter the discontinuations are listed. These distribution versions by different reasons are not supported by opsi anymore.

3.1 Discontinuation of Python 2.4 support

With this feature pack release the official support for Python 2.4 is discontinued. This regards the Python 2.4 distributions: Redhat 5 and Centos 5. The repositories concerned will be closed at the end of March 2013. The final packet versions will be available from the archive of download.uib.de. For opsi installations concerned we recommend to upgrade to the new distribution versions, which still are supported.

3.2 Discontinuation of Linux distribution versions

The following Linux distribution versions are not supported anymore because of insufficient maintenance by the distributor:

- openSuse 11.3
- Ubuntu 10.10
- Ubuntu 11.04
- Ubuntu 11.10

These repositories will be closed at the end of March 2013. For installations based on these distribution versions we strongly recommend an upgrade of the operating system. But before doing so, please check whether your chosen version to upgrade to is supported by opsi.

Chapter 4

Installation notes

The products contained within this Release in several parts are dependent from each other. So install the Release as a packet and don't try to install just some parts of it.

At first you should update the server and then the opsi products.

The installation does not require any special procedures and can be done as part of the standard update of the server and the opsi products.

As the first step you should create a server update and then install the opsi products. This is to be done with the opsi-product-updater:

```
opsi-product-updater -i -vv
```

In case you have a multi depot installation, first upgrade the config server, then the depot servers..

Caution
Since opsi 4.0.3 we removed the dependencies of the opsi server packages to the packages *dhcp server* and *java runtime environment*.
We made these changes, because most of our customers run an other dhcp server and don't need to run the opsi-configed directly on the opsi server.
This leads to an automatic uninstall of these packages if they were installed before by the opsi dependency.
If you need one of these packages on your opsi server, you should check and (if needed) reinstall these packages after upgrade to opsi 4.0.3.

Chapter 5

opsi Installation on Shutdown

5.1 Introduction

Usually the client installation of opsi software packets is started when booting the client. So the user has to wait for the installation to finish, before the user logon is granted. So it might be useful to do most of the software installation when the client is going to shut down.

The opsi module for installation on shutdown is providing that feature. Selected clients can be configured to do installation on shutdown.

5.2 Preconditions for Installation on Shutdown

The opsi module Installation on Shutdown can be used for clients with **Windows XP** or above. The required components are part of the opsi packet *opsi-client-agent*. Currently this module is a cofunding project. So it requires a special modules file to unlock this feature. For further details on cofunding projects see Section 12.1.1 and the weblink opsi cofunding projects.

The packet *opsi-client-agent* must be version 4.0.2.3-2 or above with an `opsiclientd` version 4.0.75 or above.

Some technical restrictions and constraints are the following cases:

- WAN extension: the Installation on Shutdown module currently is not applicable for clients run with WAN extension. Using the WAN extension for a client under some conditions requires to use the local configuration files, which interferes with the state handling of the installation on shutdown mechanism.
- Group Policies: part of the mechanism for Installation on Shutdown is based on shutdown scripts per Local Group Policy. Other Group Policies might override these local configurations. In this case, the required configuration for running shutdown scripts should be included to your Group Policies in charge. See Section 5.4.2 for the required configurations.
- Windows Home Edition: Windows Home does not provide the required Local Group Policy shutdown script mechanim. Therefore Installation on Shutdown cannot be used for Windows Home Edition.
- Windows 2000: For Windows 2000 clients the maximum timeout for shutdown scripts is 10 minutes. After 10 minutes the installation is aborted by the Windows system. Therefore Installation on Shutdown cannot be used for Windows 2000 clients.

5.3 Activating Installation on Shutdown

The required components for Installation on Shutdown are part of the current `opsi-client-agent` packet. For unlocking it, a valid `modules` file must be purchased and copied to the server.

Then Installation on Shutdown can be configured for suitable clients (see Section 5.2) by setting the `opsi-client-agent` product property `on_shutdown_install=on` and setting `opsi-client-agent=setup`.

This is it. All the rest of the configuration is done by the `opsi-client-agent` packet during setup.

The Installation on Shutdown is done in addition to the Installation at Startup. Usually this is the best way to ensure that the clients always have the current security updates installed, even if the client was off for a long time (when the user was on holiday for instance). If required, the standard Installation on Startup can be disabled, see Section 5.4.5. Installations in progress are continued anyway, which is triggered by the precondition `installation_pending`.

The Windows system during system shutdown does not distinguish shutdown from reboot. The Installation on Shutdown is started in both cases and it is not possible, to distinguish them during the installation process. Also it is not possible to transform a system shutdown into a reboot (or vice versa). So in case of a system shutdown, any further installations are continued at the next system startup.

5.4 Technical Concept

The following explanations are for understanding the technical design and the interactions of the components in case of special configurations or error states. Usually all of the required configurations are done by the `opsi-client-agent` packet.

5.4.1 Overview

The Installation on Shutdown module is based on several system components. A major part is the Windows shutdown script mechanism per Local Group Policy. Shutdown scripts allow to run tasks at shutdown when the user is logged off already, but all the system services are still available.

The shutdown script performs an opsi task, which triggers the opsi system service *opsiclientd* to start the installation process and waits for the task to end. So the system waits for the installation process to finish and then shuts down. During the shutdown handling, the Windows system does not distinguish shutdown from reboot, so the installation is triggered in either case.

The opsi client service *opsiclientd* is configured for processing the action `on_shutdown`, which is starting the installation process. In case of required reboots, the precondition `installation_pending` allows the process control. If during an installation a reboot is required before continuing the installation, the precondition `installation_pending` leads to continuing the installation at the next system startup (even if installation at `gui_startup` is disabled). During the state of `installation_pending`, no further Installations on Shutdown are triggered, otherwise there would be no reboot between Installation on Startup and Installation on Shutdown. So until the current installation has completed, the Installation on Shutdown is suppressed by the setting `event_on_shutdown{installation_pending}`.

The following chapter gives some further details on the Installation on Shutdown.

5.4.2 Installing by shutdown script

By special registry entries, the Local Group Policy is configured to perform an opsi shutdown skript, which triggers the installation process. The registry entries used by `opsi-client-agent` are the same as can be done by using the Group Policy-Editor `gpedit.msc`.

This is how to configure the shutdown script by using the Group Policy-Editor `gpedit.msc`:

- Local Computer Policy
- Computer Configuration
- Windows Settings
- Scripts (Startup/Shutdown)
- Shutdown

- Scripts - Add - Browse
- C:\Programme\opsi.org\opsi-client-agent\on_shutdown\doinstall32.cmd (or doinstall64.cmd for 64Bit systems)

To configure the system to wait for the completion of the installation process, the maximum waittime is set to infinite (0 seconds):

- Administrative Templates
- System - Skripts
- Maximum Waittime for Group Policy scripts
- Setting - Enabled - Seconds: 0

The opsi shutdown script `doinstall32.cmd` / `doinstall64.cmd` changes the working directory and triggers the *on_shutdown* event:

```
echo Start opsi product installation ...
cd "%ProgramFiles%\opsi.org\opsi-client-agent"
opsiclientd_shutdown_starter.exe on_shutdown
```

For 64Bit systems:

```
echo Start opsi product installation ...
cd "%ProgramFiles(x86)%\opsi.org\opsi-client-agent"
opsiclientd_shutdown_starter.exe on_shutdown
```

The `opsiclientd_shutdown_starter` triggers the installation and waits for completion, so the system shutdown is delayed while the `opsiclientd_shutdown_starter` task is running.

5.4.3 Registry Entries for executing the shutdown script

These registry entries are set by the `opsi-client-agent` packet and start the execution of the shutdown script *doinstall32.cmd* on WinXP / 32Bit clients. For 64Bit-systems the script name is `doinstall64.cmd` (instead of `doinstall32.cmd`).

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex(b):00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex:00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system]
"MaxGPOScriptWait"=dword:00000000
```

These are the registry entries for Win6 64Bit (Vista / Win7 / Win8):

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex(b):00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\Windows\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"
"PSScriptOrder"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0\0]
"Script"="C:\\Program Files (x86)\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall64.cmd"
"Parameters"=""
"IsPowershell"=dword:00000000
"ExecTime"=hex:00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system]
"MaxGPOScriptWait"=dword:00000000
```

5.4.4 Configuration of the opsiclientd

For managing the Installation on Shutdown, the opsi client service *opsiclientd* has got entries for the special Event *on_shutdown* in the configuration file *opsiclientd.conf*. These are the relevant entries:

```
[event_gui_startup]
active = True

[event_gui_startup{installation_pending}]
active = True

[event_on_shutdown]
active = False

[event_on_shutdown{installation_pending}]
active = False

[precondition_installation_pending]
installation_pending = true
```

The precondition `installation_pending` is set to *true* by the *opsiclientd* while an installation is in progress and set to *false* when the installation is finished. So an unfinished installation, which needs to proceed the installation after a reboot, can be detected by the precondition `installation_pending` still being *true* at the end of the script processing. When the `event_on_shutdown` section is set to False (which is the default), the Installation on Shutdown is disabled.

This is the configuration for a client with activated Installation on Shutdown:

```
[event_gui_startup]
active = True

[event_gui_startup{installation_pending}]
active = True

[event_on_shutdown]
```

```
active = True

[event_on_shutdown{installation_pending}]
active = False

[precondition_installation_pending]
installation_pending = true
```

So the only difference is the *event_on_shutdown* section being *True*:

```
[event_on_shutdown]
active = True.
```

The setting of the *event_on_shutdown* section is managed by the product switch `on_shutdown_install` of the `opsi-client-agent` packet.

The precondition `precondition_installation_pending = true` indicates, that the current installation process has not completed yet. This precondition stays *true*, even after several Reboots, until the current installation has completed. In case of a required Reboot during the installation process, the configuration `[event_gui_startup{installation_pending}] active = True` triggers the installation to continue at the next system startup. This configuration may not be changed, because the current installation has to be completed, before the user is allowed to logon.

Also the configuration `[event_on_shutdown{installation_pending}] active = False` must keep its value *False* at all times, because the installation has to continue AFTER the next reboot. Otherwise there wouldn't be a reboot between Installation at Startup and Installation on Shutdown.

As soon as the installation process has completed, the precondition is set to `installation_pending = false`, so the Installation on Shutdown is active again.

5.4.5 Special Configuration of Installation on Shutdown

For activating the Installation on Shutdown, just a valid modules file is required, as described in Section 5.3. Then the Shutdown on Install can be activated by setting the product switch `on_shutdown_install` of the `opsi-client-agent` packet.

As the default, the Installation at startup is also active. So it is ensured, that the client always has the current software. Even if it had been offline for a while (when the user returns from holiday, for instance).

The Installation at startup can be turned off, if desired. The configuration of the *opsi-client-agents* can be done by the web service (see: Section 12.1.2), therefore an *host parameter* should be created:

- `opsiclientd.event_gui_startup.active` (boolean, default: `true`)

By this *host parameter* the *gui_startup* event can be configured for a client. The *host parameter* can be created by *opsi-configed* or *opsi-admin*.

For creating the *host parameter* with *opsi-admin*, execute the following command on the *opsi-config-server*:

```
opsi-admin -d method config_createBool opsiclientd.event_gui_startup.active "gui_startup active" true
```

The default value *true* is the default value as it comes with the default `opsiclientd.conf`.

To disable the Installation at Startup for a client, the *host parameter* can be configured like this:

- `opsiclientd.event_gui_startup.active`: `false`

Be careful with disabling the Installation on startup, for there is a high risk of outdated software. Never change the settings of the sections with the precondition *installation_pending*. The default settings are required to manage the installation process.

For setting the *host parameter* with *opsi-admin*, execute the following command on the *opsi-config-server* (in this example for a client with the opsi-host-Id `myclient.domain.de`):

```
opsi-admin -d method configState_create opsiclientd.event_gui_startup.active myclient.domain.de false
```

With the configuration *event_gui_startup=false* there is no connect to the *opsi-config-server* and no software installation at startup. With the exception of installations in progress, which continue at system startup. This is managed by the configuration `event_on_shutdown{installation_pending}`. So do not change the settings for sections with the precondition *installation_pending*, for they are important for products, that request a reboot during installation.

5.4.6 Local Logfile

During Install on Shutdown a logfile is written:

- `C:\opsi.org\tmp\doinstall.log`

with, in case of success, has the following content:

```
doinstall32.cmd started
The current date is: .Tue 02/012/2013
Enter the new date: (mm-dd-yy) Backend connected.
modules: passed first checkpoint.
Modules file signature verified (customer: my company)
Check completed.
Event fired
Task completed.
```

This Logfile is rewritten each time and can be checked in case of a malfunction.

Chapter 6

opsi Feature *SilentInstall*

The opsi feature SilentInstall-provides for administrators the ability to install software on a client without the logged on user being disturbed. This chapter describes the characteristics of this feature and offers a guideline for configuring this new installation method.

6.1 Preconditions for the Silent Installation

For using this feature, opsi version 4.0.3. or above is required. Basically the *opsi-client-agent* version 4.0.3.1 or above is required.

6.2 Overview of the SilentInstall feature

The SilentInstall method offers the ability to install a pre-defined list of products on a client, without the user having to interrupt his work. Unlike the installation by the onDemand-Event (push Installation), the SilentInstall method does not display anything on the user desktop.

All displays are suppressed and are not to be seen on any desktop. Of course this feature bears some risk. In case of a problem, e.g. a syntax error of the *opsi-winst* script, there is no way to interact with the installation process, for no dialog windows are shown. So this would result in the *opsi-winst* and so the Event processing not coming to an end, and so no more events will be executed. To avoid this "Worst case scenario", the maximum installation time is limited by a timeout configuration. This timeout value might have to be adapted in case of an extended installation. For further information, see the configuration chapter.

Another very important ability of this feature is the predefined list of products to be installed silently. Contact to the service is established, but different from the usual procedure, the ActionRequests given by the service are ignored. The list of software to be installed is defined by an *opsi-client-agent* configuration. The "setup" action will be executed for all the products on this list and they do not have to be set to setup. As usual after processing the setup script, also the update script will be executed, if there is one. `No product dependencies will be resolved.` So either no products containing product dependencies should be installed by the SilentInstall feature, or all the products from the dependency list must be added to the SilentInstall product list. As usual the installation process is completed by sending the installation status and installation logfiles to the service. In summary it is recommended to use the SilentInstall only for products, that meet the following requirements:

- small packets or installations only
- little system load: some software installations, so for instance most of the MSI installations, request during installation most of the clients ressources. This could result in a poor system performance remaining for the user.
- installable within a fixed amount of time: the default timeout for this event is set to 300 seconds. If the installation process has not completed within the timeout, the 'opsi-winst'process will be terminated and so the event can be completed.

- no reboots required: software requesting a reboot should not be installed from the SilentInstall. With the default configuration the event is configured not to process any reboot requests. Without this safety configuration the *opsi-client-agent* could reboot the client without any warning to the user. This could result in loss of data if there is a logged on user. This could result in in an inoperable software installed by SilentInstall without reboot.

Within the default configuration swaudit and hwaudit are installed by this method. The inventory products of opsi meet all the requirements above and so are applicable for this method. With the default configuration the opsi hard- and software inventory are generated on demand, without the need to set the setup action request with the *opsi-configed*. With this method the inventory information can be generated in real-time operation. Also applicable would be any configuration products, that perform automatic repairs or restore client patches.

6.3 Executing the Silent Installation

This event will not be triggered automatically like other events. So there are two ways to perform this event.

The first way is to trigger the event from the opsi webservice, like e.g.:

```
opsi-admin -d method hostControl_fireEvent silent_install client.domain.local
```

So this command is scriptable and can be used within scripts that can be combined with an *at-job* to plan the execution of the event.

As an alternative the event can be triggered by a timer event after a certain amount of time. The default configuration for this event is 6 hours. This value presumes, that a work station usually is in use for 8 hours. So the event would be executed once a day after six hours of uptime. For mor information on configuring and activating this event see the following configuration chapter.

6.4 Configuring the opsi-feature: *SilentInstall*

This chapter is about the default configuration of this feature. The default *opsiclientd.conf* has got a SilentInstall event. This listing just shows the important options:

Standard Event SilentInstall:

```
[event_silent_install]
process_shutdown_requests = false
action_processor_productIds = swaudit,hwaudit
action_processor_command = %action_processor.command% /productlist %action_processor_productIds% /silent
action_processor_desktop = winlogon
action_processor_timeout = 300
```

- action_processor_productIds
  - This option is an important new property for the event control. For all events that perform product actions, this option can define a list of products. The product list must be given as a comma separated list.
- process_shutdown_request = false
  - this configuration suppresses reboot requests from the *opsi-winst*.
- action_processor_command
  - this prepares the call of *opsi-winst*.
- action_processor_desktop
  - This option defines the desktop to display the *opsi-winst* GUI.

- action_processor_timeout
  - This option sets the timeout for terminating the *opsi-winst*-process.

The second event is the Timer Event, which triggers the event after a certain amount of time:

Default Timer Event for SilentInstall

```
[event_timer_silentinstall]
super = silent_install
type = timer
active = false
interval = 21600
```

- super
  - This option defines the event to inherit properties from. As the default configuration the Timer-Event inherits from the event silent_install.
- type
  - This option defines this event configuration to be a Timer-Event.
- active
  - as default this event is not active. To acivate it, set this option to *true*.
- interval
  - This option defines the intertval to fire the event. The default value is set to 6 hours, so after six hours of uptime the event is triggered the first time and then every other six hours. So this interval should (like any timer intervall) not be too short, otherwise the event would be peformed most of the time and thereby block the execution of other actions. On the other hand the interval also should not be too long, for the *opsi-client-agent* must be running all that time until the event is triggered. If the client or the *opsi-client-agent* always is restarted before the interval elapsed, this event never will be triggered.

Also the SilentInstall event could be triggered by another system event detected by an WMI request. Therefore a *wql* option can be specified. How to specify a *wql* option is to be seen in the event_net_connection section. If the *wql* option is used, the event should be set to *active = false* as default, so it can be activated later on when requested.

To trigger the event by a timer, usually it only needs to set a host parameter. Therefore at first a default configuration has to be created. In this case it is sufficient to activate the Timer Event.

To create the standard option the following *host parameter* are to be created by the *opsi-admin*. Also this configuration could be created by the *opsi-configed*:

```
opsi-admin -d method config_createBool opsiclientd.event_timer_silentinstall.active "event_timer_silentinstall active" \
    false
```

So at first this event is disabled for all the clients. Then the event can be enabled for single clients:

```
opsi-admin -d method configState_create opsiclientd.event_timer_silentinstall.active silentclient.domain.de true
```

To define the products to be installed, the following entry must be set. If for instance instead of *swaudit* and *hwaudit* the product firefox shall be installed, the entries should be created as described above:

```
opsi-admin -d method config_createUnicode opsiclientd.event_silent_install.action_processor_productIds "\
    event_silent_install productIds" "swaudit,hwaudit"
```

With this option as the default for all clients the product list for the Silent Install Event is set to swaudit and hwaudit. To change the product list for a single client into *firefox* execute the following command:

```
opsi-admin -d method configState_create opsiclientd.event_silent_install.action_processor_productIds client.domain.de "\
    firefox"
```

As you can see, the product list can be different for each client.

Chapter 7

Protect your CI changes from updates: the custom directory

(available since opsi-client-agent version 4.0.2.3)

The custom directory can be used to protect your configuration changes during opsi-client-agent updates: (`/opt/pcbin/install/opsi-client-agent/files/opsi/custom`). During server updates of opsi-client-agent the whole custom directory will be saved and restored after the update, so that your custom changes will persist.

- `custom/cfg/config.ini`
  Values from this config file override values from the default `cfg/config.ini`. Except of the values for `pckey` and `bootmode`, which never are picked from that file. Add to your custom config file **only** those values, that are different from the default settings.
- `custom/winstskin/*.*`
  All the files from this directory will be copied to the clients `C:\Program Files (x86)\opsi.org\opsi-client-agent\custom\winstskin` directory during installation of the opsi-client-agent on the client. This `winstskin` directory, if it exists, since opsi-winst Version 4.11.3.4. is the preferred one. It must contain all required winstskin files and configurations, for the content of the default directory is ignored.
- `custom/notifier/*.*`
  All the files from this directory will be copied to the clients `C:\Program Files (x86)\opsi.org\opsi-client-agent\notifier` directory during installation of the opsi-client-agent and overwrite the files from the server side `files/opsi/dist/notifier/` directory.
- `custom/opsiclientd.conf`
  If it exists, the `custom/opsiclientd.conf` will be copied to the clients `C:\Program Files (x86)\opsi.org\opsi-client-agent\opsiclientd` directory during installation of the opsi-client-agent and overwrites the default opsiclientd.conf from the server side `files/opsi/dist/opsiclientd/` directory. So the custom opsiclientd.conf must contain **all** the required configuration entries.
  Attention:
  Using a custom `opsiclientd.conf` is not recommended. To customize your client configuration, use the host parameter configuration for single features as described in the opsi-client-agent chapter. Using a custom `opsiclientd.conf` is applicable for very complex configurations only. By using a custom `opsiclientd.conf`, after each update of opsi-client-agent it is required to check the server default file `files/opsi/dist/opsiclientd/opsiclientd.conf` for changes to be patched to your custom `opsiclientd.conf`.
  So: hands off this feature, unless you really know what you are doing!

Chapter 8

Adaptions of the driver integration method byAudit

Some manufacturers use model identifier codes including special characters, which cannot be used in file and directory names. For instance a model identification : "5000/6000/7000". The slash cannot be used for valid directory names. Therefore since the third service release opsi 4.0.3 the following special characters: < > ? " : | \ / * internally will be replaced by an underscore _ . So the example above will be changed to : "5000_6000_7000" and the directory will be assigned automatically, although the information in the hardware inventory is different.

Chapter 9

HostControlSafe-Backend

The default behaviour of opsi4.0 methods when called without any parameter is, that it matches all existing objects. For instance the command "host_getObjects" without any parameters results in returning all existing host objects. This could be dangerous when using the HostControl-Backend. Especially with commands like: "hostControl_shutdown" and "hostControl_reboot". In these cases calling the method without any parameter would shutdown or reboot all the clients.

Therefore service release opsi 4.0.3 comes with two changes:

- The methods: "hostControl_shutdown" and "hostControl_reboot" don't have the standard behaviour anymore and result in an error message when they are called without any parameter.
- A new backend is introduced: HostControlSafe Backend, that throws an error message for all of the methods, if they are called without any client parameter. To explicitely address all of the clients by a HostControlSafe-Backend method, the wildcard * can be used:

```
opsi-admin -d method hostControlSafe_shutdown *
```

So for the reasons mentioned above, we recommend to use the hostControlSafe methods at the console. Especially if you have little experience in using the service methods.

Chapter 10

opsi-winst 4.11.3.5

The following descriptions are excerpts from the *opsi-winst* manual.

"(...)" marks the omitted parts.

10.1 Skinnable opsi-winst

Since version 3.6 the *opsi-winst* GUI can be customized. The elements for customizing are to be found in the winstskin subdirectory of the *opsi-winst* directory. The configuration file for customization is `skin.ini`.

Since version 4.11.3.5 the *opsi-winst* searches the skin directory in the following order (first directory with a `skin.ini` to be found wins):

1. `%WinstDir%\..\custom\winstskin`
2. `%WinstDir%\winstskin`

With the Command `SetSkinDirectory` the SkinDirectory to be used can be defined in the script. If the path specified is empty or not valid, the default path will be used.

Example:

```
SetSkinDirectory "%ScriptPath%\testskin"
sleepseconds 1
SetSkinDirectory ""
```

10.1.1 Window Mode / Skin

- `SetSkinDirectory` <skindir> sets the SkinDirectory to be used and loads the skin. If the path given is empty or not valid, the default path will be used.

Example:

```
SetSkinDirectory "%ScriptPath%\testskin"
sleepseconds 1
SetSkinDirectory ""
```

10.2 Commands to control the logging

(. . .)

- `SetConfidential` <secret string>
  This prevents confidential information (like passwords) from being logged. In the logfile the confidential information will be replaced by *"(confidential)"*.
  When the loglevel is set to *9*, the confidential information will be logged.
  (since version 4.11.3.5)

Example:

```
message "SetConfidential"
SetConfidential "forbidden"
comment "This is a forbidden string"
comment "shown in the should be in the log file: This is a ***(confidential)*** string"
```

Log:

```
message SetConfidential
comment: This is a ***(secret)*** string
comment: should be in the log file: This is a ***(confidential)*** string
```

10.3 Calculating with numbers

opsi-winst scripts do not have a special type of variables for numbers. But there are some functions to help calculating with numbers.

- `calculate(`<str>`)`
  this string function calculates the arithemtic expression of the string <str> and returns the rounded result as a string.
  Internally the calculations are done with real numbers. This function accepts the operators `+`, `-`, `*`, `/` and round brackets `(,)`.
  In case of an error, an empty string is returned and the error counter is incremented. If the passed string contains any characters other than numbers, valid operators and brackets, this results in an error.
  If the second operand is missing, the first operand is also taken as the second operand: 5+ = 10 ; 5* = 25. So the strings that are used to assemble the argument should be validated by the funktion `isNumber`.
  (since version 4.11.3.5)

Example:

```
set $ConstTest$ = "0"
set $CompValue$ = calculate("-1+1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $ConstTest$ = "1"
set $CompValue$ = calculate("0+1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
```

```
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $ConstTest$ = "-1"
set $CompValue$ = calculate("0-1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "25"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "1"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "0"
set $ConstTest$ = ""
comment " expecting devision by zero error and empty string result"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "9"
set $string2$ = "10"
set $ConstTest$ = "1"
comment "result 0.9 is rounded to 1 "
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "10"
set $string2$ = "9"
set $ConstTest$ = "1"
comment "result 1.1111 is rounded to 1 "
```

```
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "55"
comment " rule * before +"
set $CompValue$ = calculate($string1$+"+"+$string2$+"*10")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "100"
comment "brackets before  rule * before + "
set $CompValue$ = calculate("("+$string1$+"+"+$string2$+")*10")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "ten"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = ""
set $ConstTest$ = "25"
comment "5* is interpreted as 5*5"
set $CompValue$ = calculate($string1$+"*")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = ""
set $ConstTest$ = "10"
comment "5+ is interpreted as 5+5"
set $CompValue$ = calculate($string1$+"+")
if ($ConstTest$ = $CompValue$)
```

```
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "nothing"
set $string2$ = "foo"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "foo"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
```

For more examples refer to the product *opsi-winst-test* at the section *$Flag_calculate$ = "on"*

10.4 PatchTextFile sections

10.4.1 Parameters

The name of the file to be patched is passed as parameter.

Optional modifiers are:

- `/AllNTUserProfiles`
  If a *PatchTextFile* section is called with this modifier and the path of the file to be patched contains the constant `%UserProfileDir%`, the patch section will be executed for all the profiles.
  For a *PatchTextFile* section, which is called from a `[ProfileActions]` section in the *Machine* mode, the modifier `/AllNTUserProfiles` is implied. In the *Loginscript* Mode the `%UserProfileDir%` is interpreted as `%CurrentProfileDir%`.
  (since version 4.11.3.5)

10.5 LinkFolder sections

(. . .)

In a *LinkFolder* section at first must be defined, in which virtual system folder the subsequent instructions are meant to operate. This expression defines the base folder:
`set_basefolder` *<virtual system folder>*

Virtual system folders to be used are:

desktop, sendto, startmenu, startup, programs, desktopdirectory, common_startmenu, common_programs, common_startup, common_desktopdirectory

These folders are virtual, for it depends on the operating system (and version), what the resulting physical directory name is.

In the context of standard *machine* installations, only the virtual system folders starting with `common_` are relevant.

The system folders *desktop, sendto, startmenu, startup, programs, desktopdirectory* can only be used in the context of a logged on user respectively in a *userLoginScript* in the context of the opsi extension *user Profile Management*.

10.6 ExecWith sections

10.6.1 Calling parameters (Modifier)

(. . .)

The following *opsi-winst*-options are available:

- `/32Bit`
  This is the default. The interpreter path is assumed to be a 32 bit path.
  Example: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` calls (also when running on a 64 bit system) the 32 bit *powershell.exe*.
- `/64Bit`
  The interpreter path is assumed to be a 64 bit path.
  Example: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` calls (on a 64 bit system) the 64 bit *powershell.exe*.
- `/SysNative`
  The given interpreter path is assigned accoring to the OS architecture.
  Example: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` calls on a 64 bit system the 64 bit *powershell.exe* and running on a 32bit system the 32 bit *powershell.exe*.

Since Version 4.11.3.5, if the interpreter path contains *powershell.exe*, the temporary file is saved with the extension `.ps1`.

10.7 WinBatch sections

10.7.1 Calling Parameters (Modifier)

- `/32Bit`
  This is the default. The paths within the section are assumed to be 32 bit paths.
  Example: `c:\windows\system32\regedit.exe` executes (even when running on a 64 bit system) the 32 bit *regedit.exe*.
- `/64Bit`
  The paths within the section are assumed to be 64 bit paths.
  Example: `c:\windows\system32\regedit.exe` executes (running on a 64 bit system) the 64 bit *regedit.exe*.
- `/SysNative`
  The paths within the section are assigned according to the OS architecture.
  Example: `c:\windows\system32\regedit.exe` running on a 64bit system calls the 64 bit *regedit.exe* and running on a 32 bit system the 32 bit *regedit.exe*.

Example:

```
Winbatch_add_reg winst /64Bit
[Winbatch_add_reg]
"c:\windows\system32\regedit.exe" /s "%scriptpath%\my64.reg"
```

- `/RunAsLoggedonUser` //since 4.11.3.5
  This is available only in the context of *userLoginScripts*. The program is executed as the user, who has just logged on. This modifier has the following limitation:

  – hardly tested on NT6 and might be of limited effect.

10.8 LDAPsearch section

10.8.1 LDAPsearch section syntax

A LDAPsearch section has the following specifications:

(. . .)

- `user:`
  user name to be applied. Since 4.11.3.5
- `password:`
  user password to be applied. Since 4.11.3.5

(. . .)

10.8.2 Examples

(. . .)

Example with user / password

```
comment ""
comment "-------------------------------------"
comment "Testing: "
comment "user / password"
Set $LdapHost$ = "vmix7.uib.local"
Set $LdapPort$ = "389"
Set $LdapUser$ = "cn=Administrator,cn=Users,dc=uib,dc=local"
Set $LdapPassword$ = "Linux123"
Set $LdapResultType$ = "objects"
Set $LdapSearchDn$ = "cn=Users,dc=uib,dc=local"
Set $LdapSearchAttributes$ = "name,objectClass"
Set $LdapFilter$ = "(&(objectclass=*))"

markErrorNumber
set $list1$ = getReturnListFromSection("ldapsearch_users /" + $LdapResultType$)
if errorsOccuredSinceMark > 0
        comment "failed while ldapsearch"
        set $TestResult$ = "not o.k."
else
        comment "passed"
endif
```

```
[ldapsearch_users]
targethost: $LdapHost$
targetport: $LdapPort$
user: $LdapUser$
password: $LdapPassword$
dn: $LdapSearchDn$
attributes: $LdapSearchAttributes$
filter: $LdapFilter$
```

Chapter 11

opsi-configed 4.0.3.1

11.1 Host parameters in the client and server configuration

For a better overview the host parameters now are devided into functional groups. The groups are displayed on the left side as a tree. The parameters and values of the selected group are displayed on the right side.

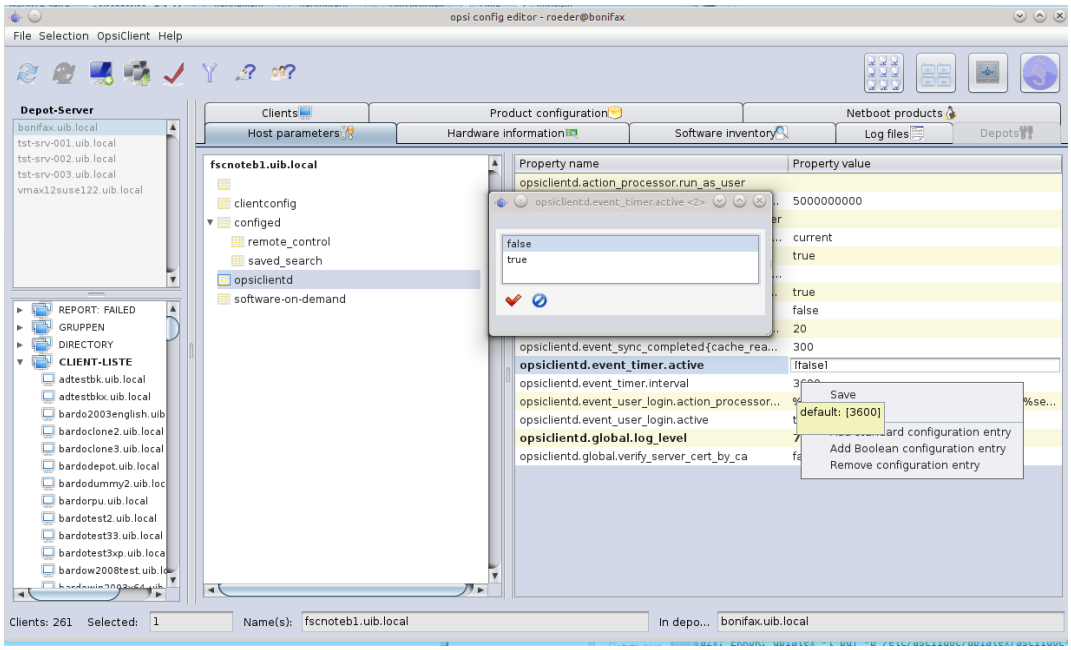

Figure 11.1: *opsi-configed*: Tab host parameter (server and client configuration)

11.2 Logfiles

The level of log entries to be shown can be selected by a slider, so it is easier to find errors. The slider can also be moved using the mouse wheel.

Logs referring to an event can be selected.

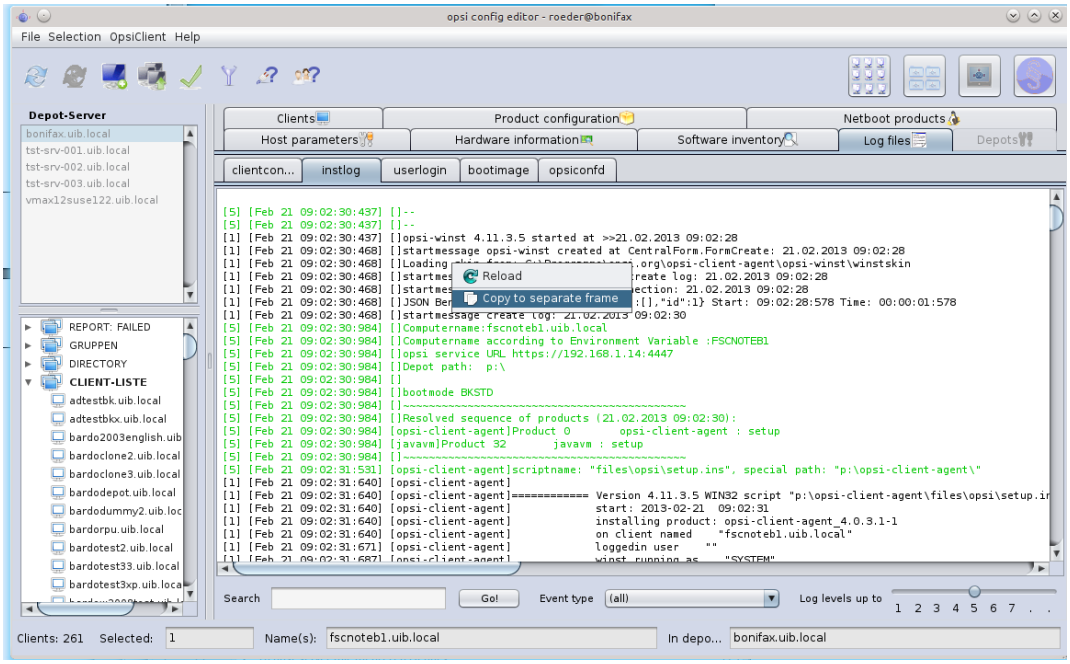

Figure 11.2: *opsi-configed*: Tab Logfiles

Chapter 12

Miscellaneous

12.1 References to the opsi manual

Some chapters of the Release Notes are also part of the *opsi-manual* and refer to other chapters of the *opsi-manual*. The chapters referred to are:

12.1.1 Activating licensed extension modules

More information on activating licensed extension modules are to be found in the *opsi-manual* in the repective chapter on extension modules.

12.1.2 Configuration by the webservice (host parameter)

More information on configuring the web service are to be found in the *opsi-manual* in the repective chapter on the client agent configuration.

12.2 Changelogs:

12.2.1 Changelog opsi-winst

opsi-winst (4.11.3.6) stable; urgency=low

- fix: no [productId] while listing of products

— Detlef Oertel <d.oertel@uib.de> Thu, 11 Mar 2013:15:00:00 +0200

opsi-winst (4.11.3.5) stable; urgency=low

- del / delete will not delete if the given Filename starts with \ (eg. *Dummy*)
- danish localization added
- italien localization added
- extended LDAPSearch (username/password)
- fix: do not store isfatalerror in registry if nor running as admin
- execwith: winst (/32bit|/64bit|/sysnative) no allowed

- execwith: if *powershell.exe* is part of the interpreter string the script extension is *.ps1*
- winbatch: /32bit|/64bit|/sysnative no allowed
- wilog: replace ' -→' by ' ==⇒' because the opsi web service get in trouble because it is interpreted as xml comment, fixes #350
- switch to Lazarus 1.0 / fpc 2.6.0
- wisynt: doAktionen, fileActionsMain: do ApplyTextConstants for filesections in fileActionsMain. fixes wrong resolution of %CurrentProfileDir% in Files /AllNTuserProfiles
- wisynt: doLinkFolderActions: fix: now comments allowed between set_link and end_link
- wisynt: doLinkFolderActions: linkfolder section now usable in userLoginScripts: the predefined basfolders: *desktop*, *sendto*, *startmenu*, *programs*, *startup*, *desktopdirectory* are pointing to the loggedin user directories if called from userLoginScript
- wimain: ProcessNonZeroScript: fix: do not run update after setup if setup is failed, fixes #432
- fix: widatamodul: opsidata4.initproduct now boolean function: false = error
- new: new winst cli parameter '/productlist <productid>[,<productid>]* as sub parameter to /opsiservice try to run the setup-scripts of the given products, change installation states, send log files
- new: string function calculate(<string expr>) ; calculates the mathematical expression and returns the result as string. on error the result is a empty string and the errorcounter is increased
- new command *setConfidential <confidential string>* ; adds the given string to a list of strings which are replaced with ***(confidential)*** while logging

— Detlef Oertel <d.oertel@uib.de> Thu, 05 Dec 2012:15:00:00 +0200

opsi-winst (4.11.3.4) stable; urgency=low

- new winbatch parameter /RunAsLoggedOnUser
- PatchTextFile sections may now called with %userprofiledir% as part of the file name and /AllNTuserProfiles so it will run on all profiles. If running in Machine mode, the */AllNTuserProfiles* is implizit in the [ProfileActions]. If running in LoginScript mode, the *%userprofiledir%* is the %CurrentProfileDir%.
- will now look for skins (more exactly: skin.ini) in the following sequence: 1) by parameter 2) new custom directory of opsi-client-agent ../custom/winstskin 3) own winstskin: winstskin

— Detlef Oertel <d.oertel@uib.de> Thu, 26 Oct 2012:15:00:00 +0200

12.2.2 Changelog opsi-winst-test

opsi-winst-test (4.11.3.5-1) stable; urgency=low

- Test of Linkfolder in login.opsiscript
- Test of use of temporary loop constant in subsub call
- Test of calculate
- Test of *no update script if setup is failed*: new property: setfailed
- Test of SetConfidential
- Test of SetSkinDirectory
- Test of LDAPSearch (username/password)

- Test of execwith: winst (/32bit|/64bit|/sysnative)
- Test of winbatch: /32bit|/64bit|/sysnative
- Test of PatchTextFile /AllNTuserProfiles

— detlef oertel <d.oertel@uib.de> Wed, 16 Jan 2013 17:00:13 +0200

opsi-winst-test (4.11.3.4-1) stable; urgency=low

- Test of Winbatch /RunAsLoggeedOnUser in login.ins

— detlef oertel <d.oertel@uib.de> Thu, 26 Oct 2012 17:00:13 +0200

12.2.3 Changelog jedit

jedit_5.0.0-3 stable; urgency=low

- opsi-winst.xml for version 4.11.3.5
- code cleanup

— detlef oertel <d.oertel@uib.de> Fri, 15 Feb 2013 16:01:53 +0200

jedit_5.0.0-2 stable; urgency=low

- jedit version with localization support

— rupert röder <r.roeder@uib.de> Tue, 27 Nov 2012 16:01:53 +0200

12.2.4 Changelog opsi-adminutils

opsi-adminutils (4.0.3.-1) stable; urgency=low

- change to 4.0.3

— detlef oertel <d.oertel@uib.de> 15 Feb 2013

opsi-adminutils (4.0.2.1-5) stable; urgency=low

- correct parameters for configed.jnlp URL

— rupert roeder <r.roeder@uib.de> 10 Jan 2013

opsi-adminutils (4.0.2.1-4) stable; urgency=low

- removed parameter from configed.jnlp URL
- removed opsi-configed version from control file

— detlef oertel <d.oertel@uib.de> 04 Jan 2013

opsi-adminutils (4.0.2.1-3) stable; urgency=low

- regshot 1.8.3 (include 64-bit)
- added kitty

— detlef oertel <d.oertel@uib.de> 11 Oct 2012

opsi-adminutils (4.0.2.1-2) stable; urgency=low * swingx.jar * configed.jar 4.0.2.5.4 — bardo wolf <b.wolf@uib.de> 04 Sep 2012

opsi-adminutils (4.0.2.1-1) stable; urgency=low

12.2.5 Changelog opsi-template

opsi-template (4.0.3-1) stable; urgency=low

- default is now without property and 32 Bit

— detlef oertel <d.oertel@uib.de> Thu, 27 Sep 2012 16:01:53 +0200

12.2.6 Changelog windows netboot products

windows (4.0.3-2) stable; urgency=low

- fix in logging in real uefi mode

```
-- uib GmbH <info@uib.de>  Thu, 11 Mar 2013 15:19:15 +0000
```

windows (4.0.3-1) stable; urgency=low

- do not copy winpe uefi boot files if not in uefi mode
- code cleanup
- new property winpe_partition_size
- showdrivers.py:
  - Option -V for version output added.
  - byAudit: Translating model and vendor from hwinvent: characters <>?":|\/* will be translated to _

— uib GmbH <info@uib.de> Thu, 14 Feb 2013 15:19:15 +0000

windows (4.0.2-2) experimental; urgency=low

- new property data_partition_create
- detect if we are on uefi
- if we are on uefi use gpt partitions
- win8 (NT 6.2) support
- new property use_raid1

— uib GmbH <info@uib.de> Mon, 27 Aug 2012 15:19:15 +0000

12.2.7 Changelog opsi-client-agent

opsi-client-agent (4.0.3.1-2) stable; urgency=low

- opsi-winst 4.11.3.6

— Detlef Oertel <d.oertel@uib.de> Tue, 12 Mar 2013 15:00:00 +0100

opsi-client-agent (4.0.2.3-1) testing; urgency=low

- opsiclientd.conf:
  - added new silent_install event

  - added new time event for silent_install
- fix search algorithm for finding usersessions
- fixed setting userrights for local opsi-winst
- added new service-method for cleaning cache (WAN-Support)

  -- Erol Ueluekmen <e.ueluekmen@uib.de> Wed, 14 Nov 2012 09:00:00 +0100

opsi-client-agent (4.0.2.3-2) testing; urgency=low

- opsiclientd.conf:
  -
- opsiclientd:
  - v4.0.1.75
  - new state installation_pending for future use (event_on_shutdown)
- new product property: on_shutdown_active (true/false)
- new updatescript, which installs shutdown files if on_shutdown_active=true

  -- Miriam Michaelis <m.michaelis@uib.de> Wed, 05 Dec 2012 13:00:00 +0100

opsi-client-agent (4.0.2.3-1) testing; urgency=low

- opsiclientd:
  - fix for user_cancelable = 0
  - fix for getSessioninformation
  - updated sessionhelper in utilities
  - added /usercontext parameter for user-login events. This option gives now opsi-winst the username.
  - fix for User-Login-Events on Windows 8: Do not fire event, if the Window Management Service is logged on.
- new custom directory
  - is saved in preinst and restored in postinst
  - cfg/config.ini values will be overwritten by values from custom/config.ini (execption: pckey , bootmode) fixes #333
  - it is a good idea to delete all not used key from a custom/config.ini
  - files at the depot in custom\winstskin*.* will be copied to custom\winstskin\ at the client and will be used by opsi-winst since version
  - files at the depot in custom\notifier*.* will be overwrite the files at the clients notifier directory
  - a existing file at the depot: custom\opsiclientd.conf will overwrite the opsiconfd.conf files at the clients opsiclientd directory, which comes from the *dist* directory

    -- Erol Ueluekmen <e.ueluekmen@uib.de> Wed, 14 Nov 2012 09:00:00 +0100

12.2.8 Changelog python-opsi

python-opsi (4.0.3.1-1) testing; urgency=low

- dhcp-backend: ddns-rev-domainname added to list where the values are written in double quotas
- System: opsi-setup --init-current-config gives an warning instead of error, when vendor not found for network device
- Posix:
  - saveImage returns the result from partclone if run was successfull.
  - readPartitionTable: Try to find out the right filesystem with blkid tool.
  - createPartition: allows linux as filesystem-type and produces partition with id 83
- WindowsDriver: byAudit: Translating model and vendor from hwinvent: characters <>?":|\/* will be translated to _
- python-opsi locale: danish added
- compareVersion: fixed handling with versions from custom packages.
- global.conf: fixed hostname entries
- fixed resource directory listing for custom packages /repository
- fix for ubuntu 12.10

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 05 Feb 2013 17:40:23 +0100
```

python-opsi (4.0.2.6-1) testing; urgency=low

- Posix: getBlockDeviceControllerInfo():
  - if no devices attached on a AHCI-Controller (maybe a lshw or a kernel bug) try to find AHCI-Controller, if found try return the first found AHCI Controller for textmode-driverintegration (only for nt5)
- Posix: modifications for newer ms-sys version
- rpm-spec-file: noreplace option for dispatch.conf.default in files-section

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Mon, 07 Nov 2012 17:34:13 +0100
```

python-opsi (4.0.2.5-1) testing; urgency=low

- fix in hwinvent procedure, don't crash if lshw don't work properly
- fix for resizeNTFSPartition if blockAlignmnet is used (ntfs-restore-image)

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 02 Nov 2012 15:00:34 +0200
```

12.2.9 Changelog opsipxeconfd

opsipxeconfd (4.0.3.1-1) stable; urgency=low

- featurepack-release 4.0.3
- this version is stable, because no changes were made, only version modified for new featurepack-release

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Thu, 14 Feb 2013 17:57:02 +0100
```

12.2.10 Changelog opsiconfd

opsiconfd (4.0.3.1-1) testing; urgency=low

- ressource: webstart jnlp build fixed for arguments.
- added monitoring debug switch in opsiconfd.conf
- fixed init-Script for using global.conf

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 05 Jan 2013 13:15:56 +0100
```

12.2.11 Changelog opsi-utils

opsi-utils (4.0.3.1-1) testing; urgency=low

- opsi-product-updater: Added Blank line between header and message-body to fix empty body mails on exchange environment.
- opsi-product-updater: fix for handling with custom products.
- opsi-admin: added new HostControlSafeBackend in BackendManager initialization.

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 12 Feb 2013 12:42:44 +0200
```

12.2.12 Changelog opsi-linux-bootimage

opsi-linux-bootimage (20130207-1) testing; urgency=low

- added bc

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 01 Feb 2013 12:34:59 +0100
```

opsi-linux-bootimage (20130117-1) experimental; urgency=low

- downgrade: ms-sys: 2.1.2 kernel 3.6.11

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 17 Jan 2013 16:25:11 +0100
```

opsi-linux-bootimage (20130111-1) experimental; urgency=low

- python-opsi 4.0.2.7-1
- locale: danish and italian localisation added
- kernel 3.7.1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 11 Jan 2013 11:48:23 +0100
```

opsi-linux-bootimage (20121107-1) experimental; urgency=low

- new ms-sys-version: 2.2.1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 07 Nov 2012 17:51:31 +0100
```

opsi-linux-bootimage (20121105-1) experimental; urgency=low

- python-opsi 4.0.2.6-1

-- Erol Ueluekmen <e.ueluekmen@uib.de> Mon, 05 Nov 2012 17:36:27 +0100

opsi-linux-bootimage (20121102-1) experimental; urgency=low

- kernel 3.6.5
- python-opsi 4.0.2.5-1

-- Erol Ueluekmen <e.ueluekmen@uib.de> Fri, 02 Nov 2012 15:11:43 +0100

opsi-linux-bootimage (20121024-1) unstable; urgency=low

- kernel 3.6.3
- python-opsi 4.0.2.5-1
- added packages: cloop, cloop-utils, dmraid, rsync, partclone-utils, nbd-client, genisoimage

-- Erol Ueluekmen <e.ueluekmen@uib.de> Wed, 24 Oct 2012 12:57:00 +0200

opsi-linux-bootimage (20121011-1) experimental; urgency=low

- kernel 3.6.1
- python-opsi 4.0.2.4-1
- Missing firmware from Broadcom-Chipsets added

-- Erol Ueluekmen <e.ueluekmen@uib.de> Thu, 11 Oct 2012 00:45:26 +0200

12.2.13 Changelog opsi-depotserver

opsi-depotserver (4.0.3.1-1) testing; urgency=low

- opsi-setup: added renew-opsiconfd-cert task
- opsi-setup: added special files in set rights to add execute bit for these files in depot

-- Erol Ueluekmen <e.ueluekmen@uib.de> Mon, 28 Jan 2013 17:34:42 +0100

12.2.14 Changelog opsi4ucs

opsi4ucs (4.0.3.1-1) testing; urgency=low

- join script: execute /usr/lib/univention-pam/ldap-group-to-file.py to resync the groups (for ucs 3.1 support)
- opsi-setup: set executebit for special opsi files in depot

-- Erol Ueluekmen <e.ueluekmen@uib.de> Tue, 05 Feb 2013 11:07:47 +0100

12.2.15 Changelog opsi-configed

opsi-configed (4.0.3.2-1) testing; urgency=low

- checking names for remote controls
- improvement of restart after language change
- improvement of swaudit page

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 13 Mar 2013 12:00:00 +0200
```

opsi-configed (4.0.3.1-1) testing; urgency=low

- host config editing grouped by a tree
- last change column for products now possible
- the configed log files are now placed in the user home directory in a folder .configed
- log panes now allow selection of displayed levels
- log panes allow selection of event type if a log file has lines with event type specification
- improved search handling in log panes
- log panes, software and hardware audit panes can be copied into separated frames

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 25 Jan 2013 12:00:00 +0200
```

- search function for windows software panel
- reset client selection when returning from server or depot editing
- bugfix in mac address editing
- configed checks if read only version is requested and, if wanted, produces it

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 03 Dec 2012 12:00:00 +0200
```

opsi-configed (4.0.2.10-1) testing; urgency=low

- new logging format, included time check for service calls

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 02 Nov 2012 12:00:00 +0200

opsi-configed (4.0.2.8-1) testing;  urgency=low
```

- check of transgression of activated client numbers
- read only option

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 22 Oct 2012 12:00:00 +0200
```

opsi-configed (4.0.2.7-1) testing; urgency=low

- reopening a saved search

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 15 Oct 2012 12:00:00 +0200
```

opsi-configed (4.0.2.7-1) testing; urgency=low

- more caching and less reloading actions

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 15 Oct 2012 12:00:00 +0200
```