HP OpenView Operations for UNIX Configuration Value Pack

Installation and Administration Guide

Version: 3.0

(Document version 3.0.11 - 2006/12/20)

Manufacturing Part Number: None
December 2006
U.S.

© Copyright 2006 Hewlett-Packard Development Company, L.P.

Legal Notices

Warranty.

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend.

Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph

(c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company

United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices.

©Copyright 2006 Hewlett-Packard Development Company, L.P.

© Copyright 2006 Blue Elephant Systems GmbH

No part of this document may be copied, reproduced, or translated to another language without the prior written consent of Hewlett-Packard Company. The information contained in this material is subject to change without notice.

This software includes software developed by various open-source projects and organizations as listed below. The corresponding files and components are copyright to the corresponding organization or vendor and all rights reserved. The software files and components distributed under the open-source licenses are distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the license of the corresponding project for specific rights and limitations under the license. Depending on the license, any product derived from the products may not be called with the name of the project nor may the name of the project appear in their name, without prior written permission. For written permission, please contact the corresponding project owner by visiting the corresponding project home page as listed below.

We greatly appreciate the work of those projects. blue elephant systems GmbH tries to contribute as much as possible to some of those projects in order to compensate them for their contributions.

All license files can be found in the installation directory (./docs/licenses). Furthermore, a full overview of all external dependencies and license conditions is also accessible via the web interface: HELP button -> About menu -> External Dependencies. This includes links to all projects.

This product includes software developed by the Acegi System for Spring Project (http://acegisecurity.org/)

This product includes software developed by the ActiveMQ project (http://activemq.org/)

This product includes software developed by the Ant-Contrib project (http://sourceforge.net/projects/ant-contrib)

This product includes software developed by the Apache Software Foundation (http://www.apache.org/). These are “Ant”, “BCEL”, “Cocoon”, “Commons”, “Excalibur”, “FOP”, “Forrest”, “ FTPServer”, “Jasper”, “Log4j”, “Lucene”, “ORO”, “POI”, “Xalan”, “Xerces” and “XML

RPC”.

This product also includes software developed by the dnsjava project (http://www.dnsjava.org/)

This product also includes software developed by the Docbook project (http://www.docbook.org/)

This product also includes software developed by the dom4j project (http://dom4j.org/)

This product also includes software developed by the Drools project (http://drools.codehaus.org/)

This product also includes software developed by the eXist project (http://www.exist-db.org/)

This product also includes software developed by IBM. Copyright (c) 1995-2003 International Business Machines Corporation and others. All rights reserved. (http://www.ibm.com/software/globalization/icu/)

This product also includes software developed by the j2ssh project (http://sourceforge.net/projects/sshtools/)

This product also includes software developed by the Janino project (http://www.janino.net/)

This product also includes software developed by the Jaxen project (http://jaxen.org/)

This product also includes software developed by the Jaxup project (http://klomp.org/jaxup/)

This product also includes software developed by the JDOM project (http://www.jdom.org/)

This product also includes software developed by the Jencks project (http://jencks.org/)

This product also includes software developed by the Jetty project (http://jetty.mortbay.org/)

This product also includes software developed by the jRegistryKey project (http://sourceforge.net/projects/jregistrykey/)

This product also includes software developed by the JPam project (http://jpam.sourceforge.net/)

This product also includes software developed by the Jsch project (http://www.jcraft.com/jsch/)

This product also includes software developed by the Jython project (http://www.jython.org/)

This product also includes software developed by the Marser project (https://marser.dev.java.net/)

This product also includes software developed by the MX4J project (http://mx4j.sourceforge.net/)

This product also includes software developed by the Netbeans CVS project (http://javacvs.netbeans.org/library/)

This product also includes software developed by the openadaptor project (https://www.openadaptor.org/)

This product also includes Oracle JDBC Driver software developed by the Oracle Corporation which is used to connect to Oracle databases from java programs. You may not use this file except in compliance with the License. Software under this license is provided "as is" without warranty of any kind. All warranties, express and implied, including without limitation, any implied warranties of merchantability, fitness for a particular purpose or noninfringement are disclaimed. (http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html)

This product also includes software developed by the Quartz project (http://www.opensymphony.com/quartz/)

This product also includes software developed by the Rhino project (http://www.mozilla.org/rhino/)

This product also includes software developed by the ServiceMix project

(http://www.servicemix.org/)

This product also includes software developed by the Spring project (http://www.springframework.org/)

This product also includes software developed by the StaX project (https://sjsxp.dev.java.net/)

This product also includes software developed by the TM4J project (http://www.tm4j.org/)

This product also includes software developed by the VMTools project (http://www.vmsystems.net/vmtools/)

This product also includes software developed by the Java Service Wrapper project (http://wrapper.tanukisoftware.org/)

This product also includes software developed by the XBean project (http://geronimo.apache.org/xbean/)

This product also includes software developed by the XIA project (http://www.jeckle.de/freeStuff/xia/)

Trademark Notices.

Adobe ® is a trademark of Adobe Systems Incorporated.

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft ® is a U.S. registered trademark of Microsoft Corporation.

Netscape™ and Netscape Navigator™ are U.S. trademarks of Netscape Communications Corporation.

Oracle ® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

OSF, OSF/1, OSF/Motif, Motif, and Open Software Foundation are trademarks of the Open Software Foundation in the U.S. and other countries.

SQL*Plus ® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

UNIX ® is a registered trademark of the Open Group.

Windows NT ® is a U.S. registered trademark of Microsoft Corporation.

Windows® and MS Windows® are U.S. registered trademarks of Microsoft Corporation.

Zip and UnZip are U.S. registered trademarks of Info-ZIP.

Export and Cryptography Notice.

This software may not be exported, re-exported, transferred or downloaded to or within (or to a national resident of) countries under U.S. economic embargo including the following countries:
Cuba, Iran, Iraq, Libya, North Korea, Sudan and Syria
This list is subject to change.
This software may not be exported, re-exported, transferred or downloaded to persons or entities listed on the U.S. Department of Commerce Denied Persons List, Entity List of proliferation concern or on any U.S. Treasury Department Designated Nationals exclusion list, or to parties directly or indirectly involved in the development or production of nuclear, chemical, biological weapons or in missile technology programs as specified in the U.S. Export Administration Regulations (15 CFR 744).
In addition, please be advised that this software contains cryptography and is subject to U.S. Cryptography export regulations.

Conventions

Boldface	Words in boldface type represent programs and commands.
Capitalization	Capitalized first letters represent company or product names.
`Computer font`	Words in computer font represent file or path names, command syntax statements, prompts or messages that appear on your screen or text you should type on your workstation or terminal.
Italics	Words in italics represent variables in syntax statements or words that are emphasized in the text.
{ }	Represents required elements in a syntax statement. When several elements are separated by the | symbol you must select one of the elements.
[ ]	Represents optional elements in a syntax statement.

Document History

FONTS used:

Arial, A.C.M.E._Secret_Agent_Italic.ttf, Bitstream Vera Serif, Courier New

New editions are complete revisions of the manual. The printing dates for each edition are listed below.

Edition 1	Dec. 2006

Table of Content

1 Introduction

Your company's business success relies on high-quality IT services and IT infrastructure agility. To keep your IT services available and well performing, you need a proven operations management solution that gives you control over your ever-changing IT infrastructure. That solution is HP OpenView Operations for UNIX.

Operations for UNIX discovers, monitors, controls and reports on the availability and performance of your heterogeneous, large-scale IT environment. It consolidates information for all IT components that control your business: network, systems, storage, databases, and applications. With its service-driven approach, it shows what IT problems affect your business processes, helping you to focus on what's most important for your company's business success.

For a general overview about OVO/UNIX's feature set, refer to the *HP OpenView Operations Concepts Guide (UNIX),* which is available in PDF format on the HP product manual website.

This manual covers installation and administration aspects of the **HP OpenView Operations for UNIX Configuration Value Pack** (CVP). It is an add-on product for OVO/UNIX to help you run and operate OVO with numerous improvements and added features. It is based on the MIDAS 2.x Documentor product (see http://www.blue-elephant-systems.com for details). MIDAS stands for Management Instrumentation Documentation and Automation System.

This document provides information about installation, configuration and troubleshooting of CVP. See also the CVP on-line help, particularly for all aspects of actually using the product.

NOTE: Check the following web site periodically for the latest versions of this and other OVO/UNIX manuals:

http://ovweb.external.hp.com/lpe/doc_serv/

Select "Operations for UNIX" and version 8.x.

---

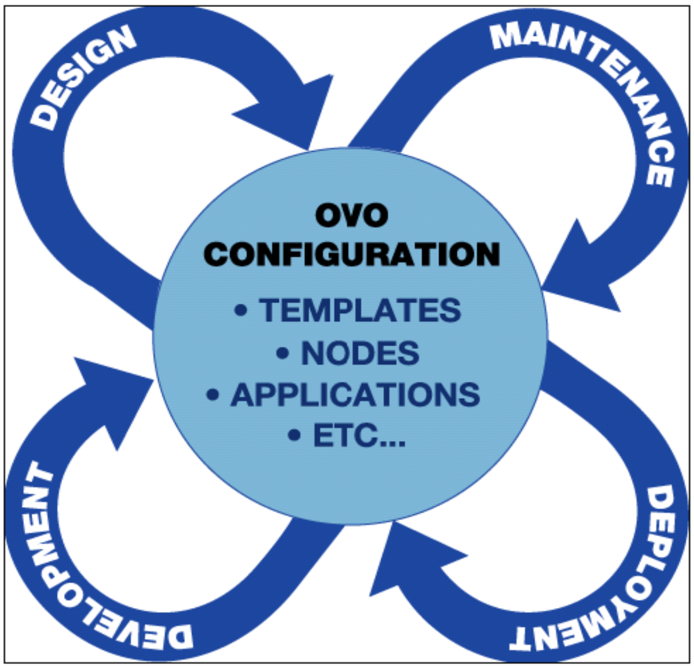
3 About CVP

CVP 3.0 is based on the MIDAS Documentor 2.x. The initial focus was documentation (on-line browsing and generation of documentation, e.g. in PDF format). The CVP 3.0 extends these documentation functions by real administrative capabilities like creating, modifying and assigning configuration objects. The current version supports OVO/UNIX only.

CVP 3.0 will be marketed by both HP and blue elephant systems under their respective brands. All statements in this manual apply to both products identically unless stated otherwise.

IT management products like HP OVO are used in large companies world-wide to monitor and control the behavior of IT resources such as hardware, software and applications. This task involves standard products like HP OVO, but also requires a set of custom configuration data, scripts, programs, etc. (collectively referred to as *instrumentation*).

Corporations have made large investments in recent years to develop and deploy this instrumentation, according to the specific needs to manage their IT environments. Both, the actual IT infrastructure and the instrumentation to manage it, have become very complex, and keep changing over time. Good documentation and a powerful interface are the keys to efficient operations, change and analysis of the IT infrastructure.



CVP defines, organizes and automates configuration tasks. The product streamlines the process of designing, developing, deploying and maintaining HP OVO configuration items such as templates, applications, nodes, and users.

CVP improves the efficiency of the entire instrumentation life cycle.

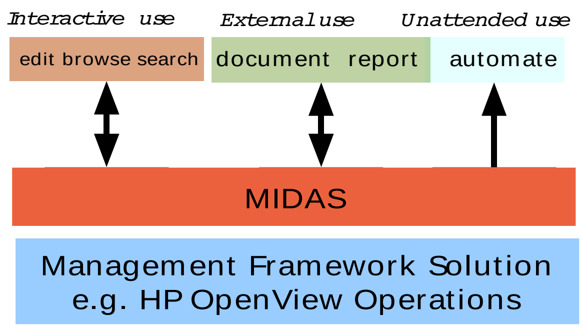
CVP enforces and supports conventions and standard procedures, and significantly increases the consistency of the management configuration. CVP helps to automate the implementation and deployment of IT management services.

How did the last upgrade to a new SPI work out for you? Trying to migrate two years of SPI modifications to a new release can cause a lot of headaches. With CVP you can easily compare policy groups and policies and directly see what has changed, is new or has been moved. You can then easily copy your modifications to the new policy.

3.1 Benefits

CVP increases management control over a distributed and complex IT environment in the following ways:

- **More efficient workflow**
  automation of development and deployment tasks.
- **Overview and Insight**
  efficient and flexible access to accurate HP OVO instrumentation
- **Shared Access**
  Share instrumentation documentation across organizations or with external suppliers. No need to wait for the opc_adm user to be available. Concurrent access by multiple users with individually defined Read, Write, Execute privileges down to template or node level is available.



3.2 Features

CVP provides efficient and flexible means to access & edit configuration information and instrumentation.

CVP

- provides a single view of all configuration information.
- generates dynamic documentation (HTML, PDF, XLS, RTF)
- consolidates data on all configuration items (live view on data)
- provides generic search capabilities
- automatic document generation (e.g. nightly)
- enables automation of IT management processes
- seamlessly integrates with HP OpenView Operations.
- provides seamless integration to CVS (Concurrent Versions System) version control
- multiple, parallel user access into HP OpenView Operations/UNIX

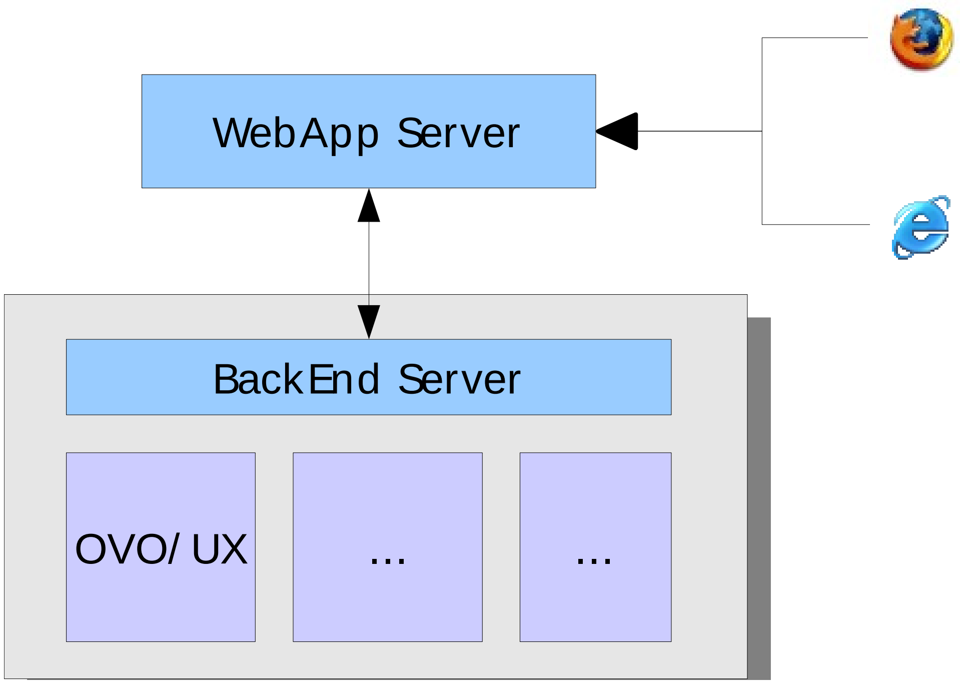
- compare two policies or policy groups with two clicks
- negative queries: find Message Groups, Node Groups not in a responsibility matrix, Policies, Policy Groups without assignments or unused
- easier analysis: see all parent policy groups and all assignments to nodes or node groups
- user model with detailed Read, Write, Execute privileges down to template or node level

4 Architecture

4.1 Architecture Overview

CVP is implemented based on a three-tier architecture. Main user interface is a regular web browser, so there is no need to install any additional software on the end-user system. The **CVP Web Application** (**WebApp** – see below) is responsible for generating the dynamic web pages, central data storage and processing. It accesses one or multiple **BackEnd** servers which provide the actual management information.

One BackEnd server forms a logical *BackEnd* that might encapsule multiple IT management applications installed on that same system such as a OVO and a NNM server. All such sub-systems together are referred to by one unique BackEnd name by **CVP** typically named “hostname_server” (where hostname is the name of the system this BackEnd is installed on).



The main data flow between WebApp Server and the individual BackEnd Servers requires one HTTP(S) connection. See below for details.
The current version of the CVP supports only OVO/Unix.

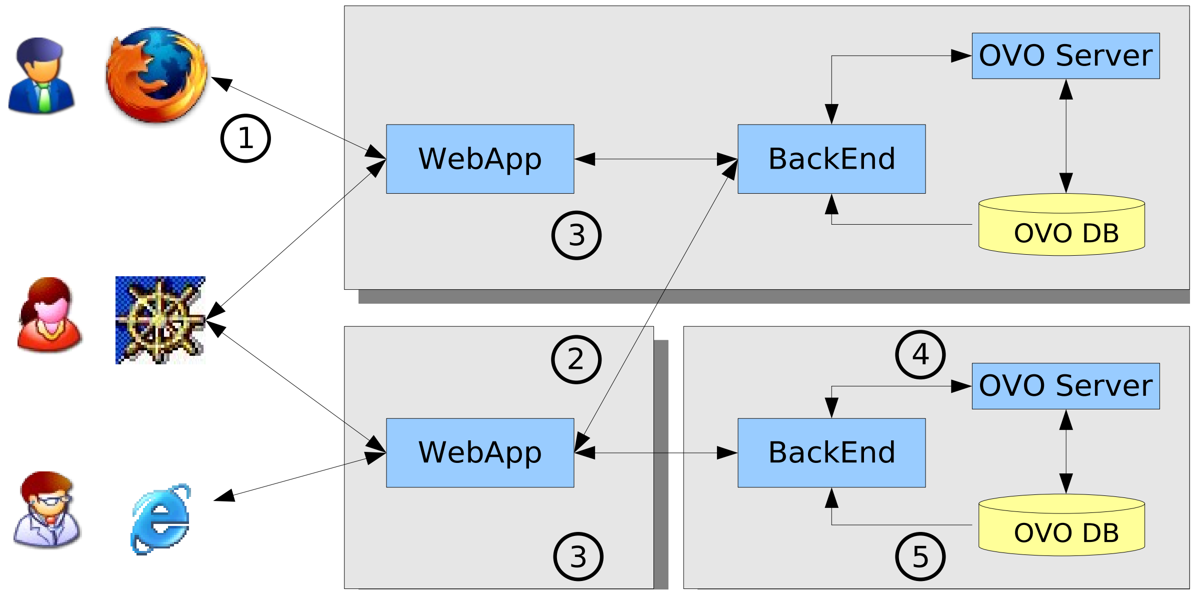
4.2 CVP Program Components

Generally, **CVP** consists of

- **CVP** BackEnd server (BackEnd)
  This component on one hand interacts with the management framework server (e.g. HP OVO server). It uses the interfaces of the management framework server to obtain data about the management instrumentation and configuration (4 and 5 below), according to the requests it gets from the web application.
  On the other hand it provides this data as XML through a URL schema to the **CVP** web application.
  Currently, the **CVP** BackEnd Server runs always on the same machine as the management framework server.

- **CVP** Web Application ("WebApp")
  This component links and formats the management instrumentation and configuration data to enable comfortable data display through a web browser. The **CVP** web application may run on the same system as a BackEnd but can also run on a separate server (3). The latter is recommended to share the system load. One **CVP** web application can handle multiple **CVP** BackEnd servers (2).
- The user front-end, a regular web browser like Mozilla Firefox, Internet Exploder ... The user connects to the WebApp server (or even multiple (1)) and may now work concurrently with all registered BackEnds.

This can be illustrated with the following picture:



The following guidelines help determining the most appropriate structure for your CVP environment:

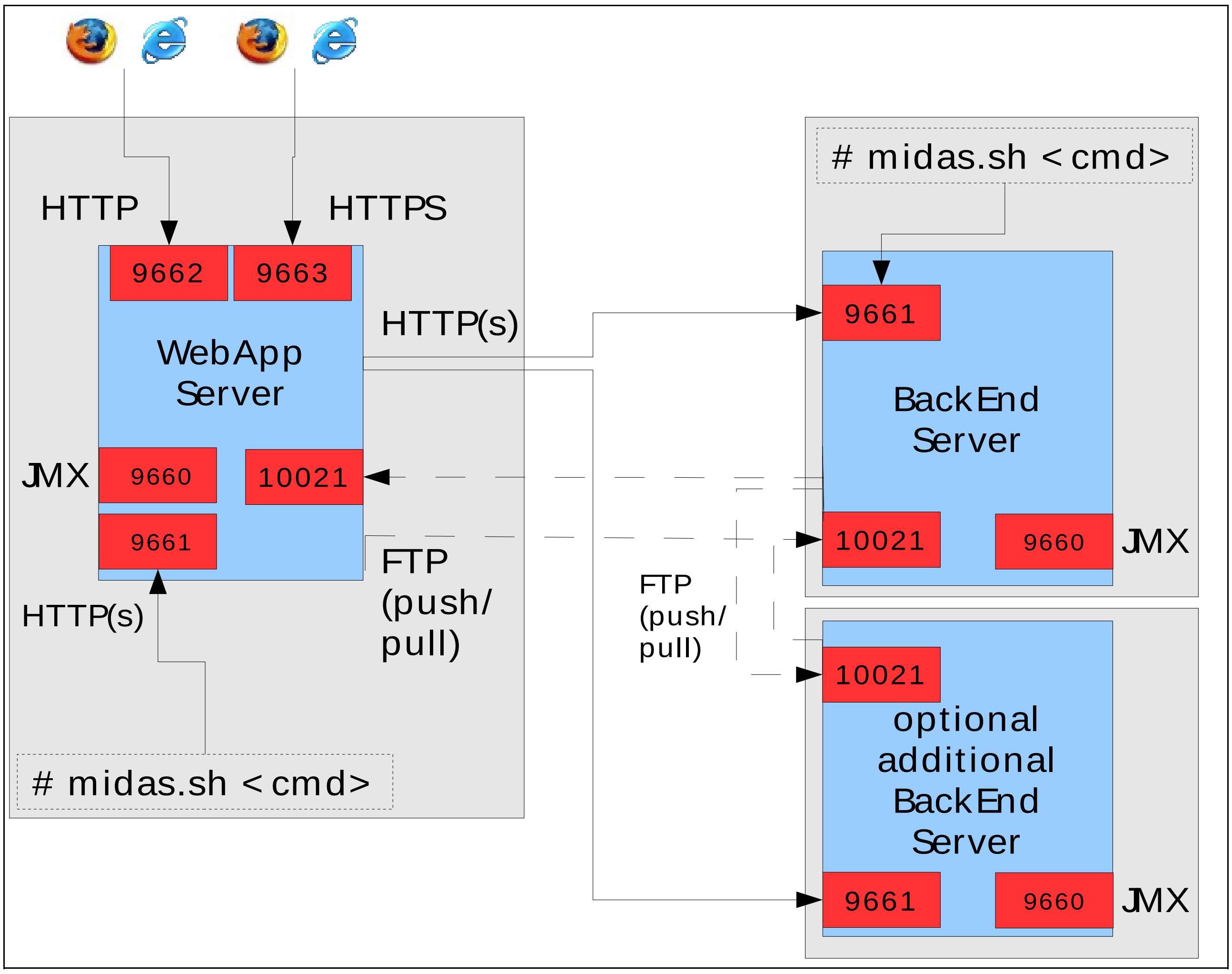
- Avoid multiple WebApp systems except in completely separate environments. The WebApp system hosts the CVP user configuration and the CVS repository. With multiple WebApps this information has to be maintained multiple times.
  Ideally there is one WebApp (possibly clustered) handling all (one or many) BackEnd systems.
- For testing and evaluation purposes a Full Installation with both BackEnd and WebApp on the OVO Management Server is suitable. However, in a productive environment it is recommended to install the WebApp on a dedicated system (can be a Linux or Windows PC) to move system load off the OVO Management Server.

4.3 Communication and Ports

Generally there are two types of communication within CVP:

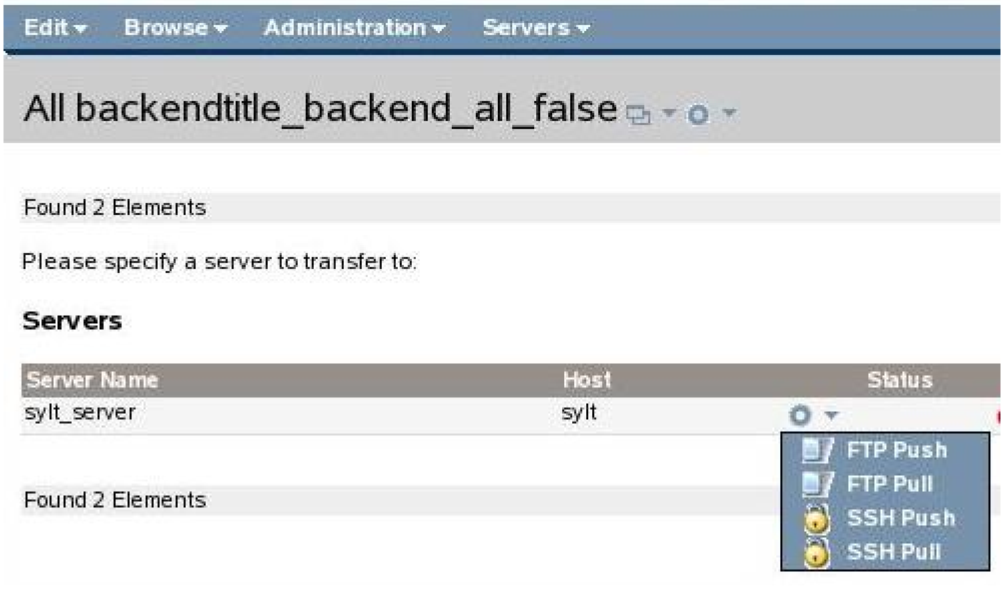
- Main HTTP(s) traffic for all regular operations
- Volume data traffic for exchanging mass data (either FTP or SSH)

For the general HTTP communication, only the ports have to be configured correctly. HTTPS requires in addition the installation of SSL keys and certificates. This data flow can be illustrated like this:



4.3.1 FTP vs. SSH communication

Mass data (for example downloaded configuration) can be transferred between CVP servers (both WebApp and/or BackEnds) either via FTP or SSH. Assuming that both methods have been enabled during installation and configured correctly, the user can choose between FTP and SSH during runtime in combination with a push or pull model as shown in the following picture:



All backendtitle_backend_all_false

Found 2 Elements

Please specify a server to transfer to:

Servers

Server Name	Host	Status
sylt_server	sylt	

Found 2 Elements

In this picture, the user already has selected the elements to transfer (not visible) and the target server (sylt_server).

In CVP, FTP is the default method and does not require any extra configuration. Both the FTP client and server components are built into every CVP server (WebApp and BackEnd).

For secure data transfer use SSH instead. If SSH is desired, the SSH software has to be installed on all involved systems **before** installing CVP. The SSH client is built into all CVP servers. In addition, the OS user `midas` must exist on systems acting as SSH server and SSH keys must be exchanged for non-interactive access. See chapter 7.3 Secure Shell (SSH) for details.

Files exchanged between CVP always reside in the so-called clipboard in `<MIDAS_HOME>/data/clipboard`. Downloaded objects will be created here as well as data received from other CVP servers. This applies to both FTP and SSH transfer.

For further details about configuring FTP and SSH, see chapter 9.9 File transfer configuration.

4.3.2 Push vs. Pull mode

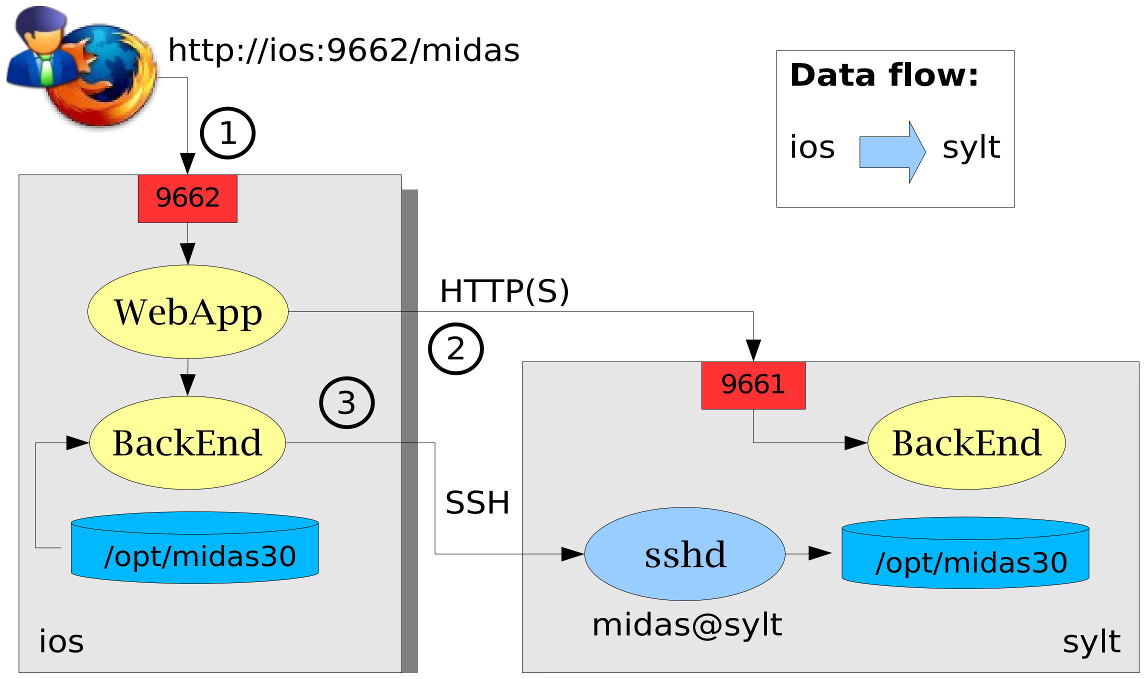
No matter whether using FTP or SSH, there is one party acting as server and the other as client. The server just waits for requests, whereas the client initiates the communication and calls the server. In the CVP scenario, the client wants to push or pull data files to/from the server.

The only difference between pushing and pulling data is where the network connection gets initiated – on the logical source or target system. This is not relevant for the final result (the files end up on the target server), but it may be for environments with firewalls between both. In such an environment, traffic may only be permitted in one direction.

Both modes (push and pull) are available for both FTP and SSH. The following examples use SSH, but FTP generally works the same way.

Also, the following scenarios assume that the WebApp resides on the same system as the source BackEnd – this is not required though. Also, transfers can be performed from/to WebApp systems as well.

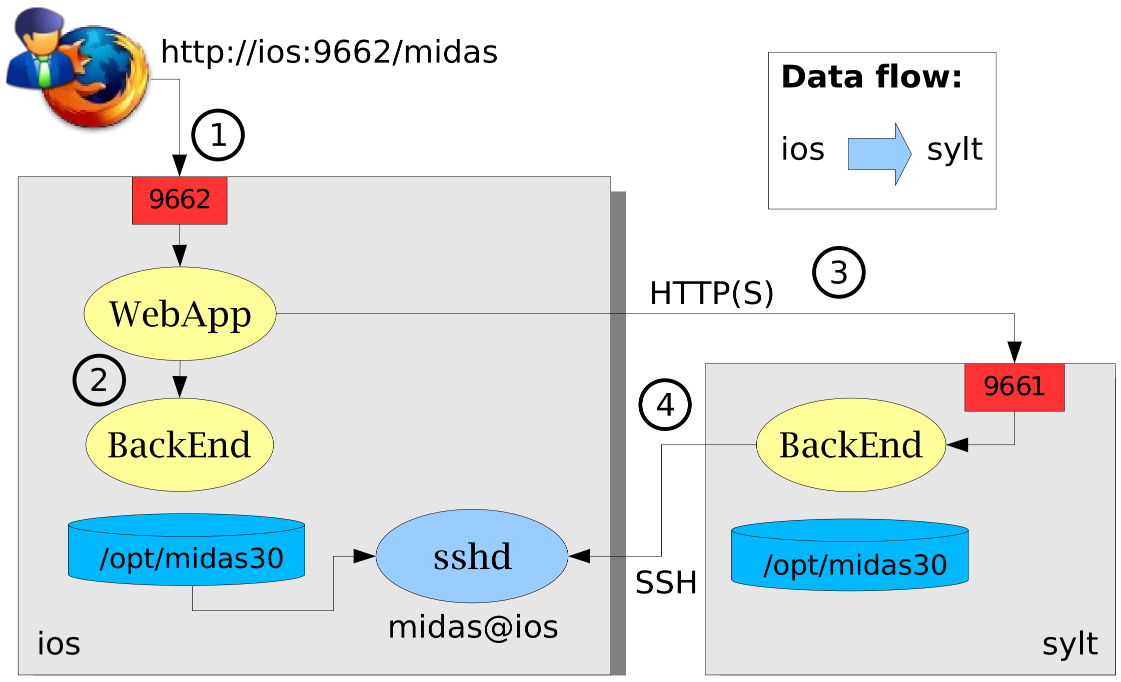
In push mode, the data flows as shown in the following picture:



The individual steps performed are:

1. The GUI is connected to the WebApp on ios. The user initiates a SSH transfer from ios to sylt in Push mode.
2. The WebApp on ios contacts the BackEnd on sylt to query BackEnd parameters (mainly the installation directory).
3. The WebApp sends a transfer request to BackEnd ios including the target host name and installation directory. The source BackEnd on ios performs the actual SSH transfer by invoking the built-in SSH client (as the same OS user as as who the CVP server is running) which connects the `sshd` on sylt. The SSH user account on sylt is `midas`, who must be able to write the received data to `<MIDAS_HOME>/data/clipboard`.

In pull mode, the procedure is slightly different as shown in the following picture:



Again, the logical data flow is from ios to sylt, this time however, initiated on sylt.

1. The GUI contacts the WebApp on ios.
2. The WebApp resolves the BackEnd ios (particularly the installation directory).
3. The WebApp contacts the BackEnd on sylt and requests the SSH transfer. The source directory on ios as determined in step 2 will be passed to the BackEnd on sylt.
4. The built-in SSH client on the BackEnd on sylt connects to the `sshd` on ios as user `midas` who must be able to read the data to be transferred to sylt.

In both Push and Pull mode the following requirements must be fulfilled to be able to perform a transfer between two CVP servers:

- All involved CVP servers (WebApp, source and target server) must be running.
- Both the source and target server must be registered and reachable from the WebApp.
- FTP or SSH connectivity must exist between source and target CVP server. Depending on the used mode (Push or Pull), traffic will be initiated from either side.
- If using SSH, on the system acting as SSH server, the OS user `midas` must exist and SSH keys must be exchanged for non-interactive log-on.
  The user `midas` must be able to read and write files in the source and target directories (in both cases the clipboard under `<MIDAS_HOME>/data/clipboard`).

4.4 Directory Layout

The **CVP** installer contains all necessary files & applications (e.g. Java). All CVP components are installed in one directory on a node, if the CVP installation on the local system acts in multiple roles (i.e. both BackEnd and WebApp as in a Full Installation) all related components are in the same location. Each component has its own sub-directory below the **CVP** installation directory. The installation directory will be referenced later as <MIDAS_HOME>.

The default location on Unix is /opt/MIDAS or on Windows C:\MIDAS.

For details about the most important directories and contained files see chapter Directory Structure.

4.5 Processes

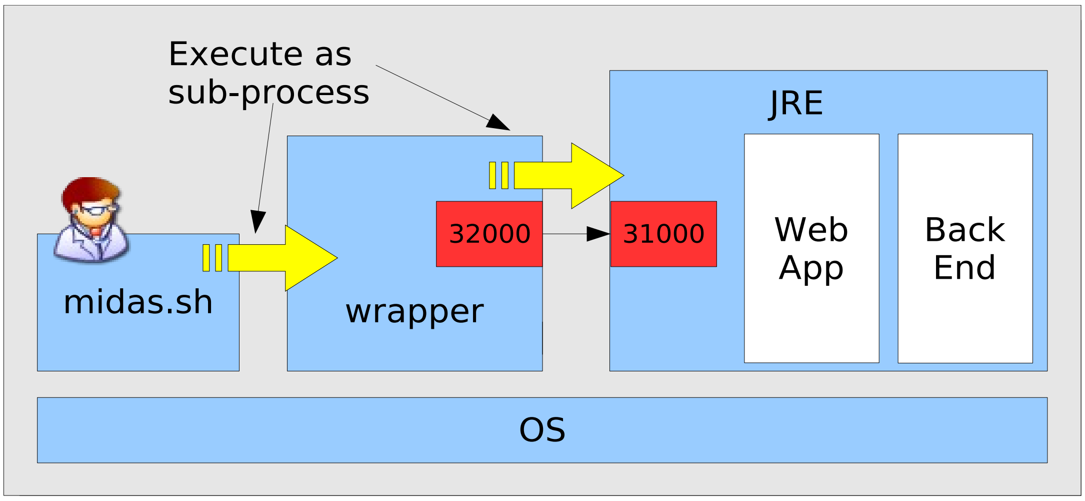
CVP consists of two processes on OS level:

```
# ps -ef | grep midas
[...]
 root   1974  1973  0 Nov 20 ? 66:36 /opt/midas30/jre/bin/IA64N/java
-DbesId=midas_server -Dclassworlds.conf=/opt/conf/servicemix.conf -Ds
  root  1973    1  0 Nov 20 ?  1:13 /opt/midas30/wrapper
/opt/midas30/conf/wrapper.conf wrapper.syslog.ident=midas_server
wrapper.pidfile=/opt/mi
[...]
```

The wrapper process is the OS interface and controls the java process, which runs the JRE performing all real work. The wrapper process starts, stops and monitors the java process.

The midas.sh command launches the wrapper process, if the CVP server gets started from the command line (during system boot the OS will do this). The wrapper process runs as OS daemon, so the parent PID of the wrapper process is 1 as shown above. To stop the CVP server, the midas.sh script just kills the wrapper process on OS level (on Windows systems, the service control functions will be used).

This looks like the following:



The ports shown are used locally for internal communication. The actual port numbers are the default numbers and are configurable (see chapter 9.7.1.4 Internal communication).

More details about the wrapper open-source software can be found under http://wrapper.tanukisoftware.org.

4.6 Temporary storage

CVP allows collecting and storing data in to dedicated areas:

- The shopping cart is a volatile structure existing in memory of the GUI session, i.e. the web browser. It is specific per GUI session and connected BackEnd. The shopping cart may contain a set of references of arbitrary type. It is not organized hierarchically. Upon terminating the GUI session, the shopping cart is lost.
- The clipboard is a persistent area in the file system. It exists on all CVP servers (WebApp and BackEnd) and is shared by all users connected to that CVP server.
  The contents depends on the data source:
  - Data checked out from CVS is stored as directory structure containing XML files as leaf elements.
  - Data downloaded using CVP exists as OVO config download structure. See the OVO product documentation for details.

  In both cases, the data is stored in individual sub-directory structures, which are named uniquely, for example `opccfg_1166559172367_Release1.3`. This name contains a unique ID (like 1166558172367) and a user-definable label (for example Release1.3) to avoid conflicts.
  The clipboard is always located in `<MIDAS_HOME>/data/clipboard`. To review the contents, use the CVP GUI or OS commands.

The shopping cart is typically used to collect items belonging together logically, for example all elements needed for managing a certain application (OVO policies and groups, node groups, user profiles, ...). These objects can be collectively checked in to CVS or downloaded for archiving or transfer to another CVP BackEnd.

The clipboard is the location where data is stored at the beginning and end of download, upload and transfer operations. For example, a download writes into the clipboard, an upload reads from the clipboard.

4.7 The CVP Server component architecture

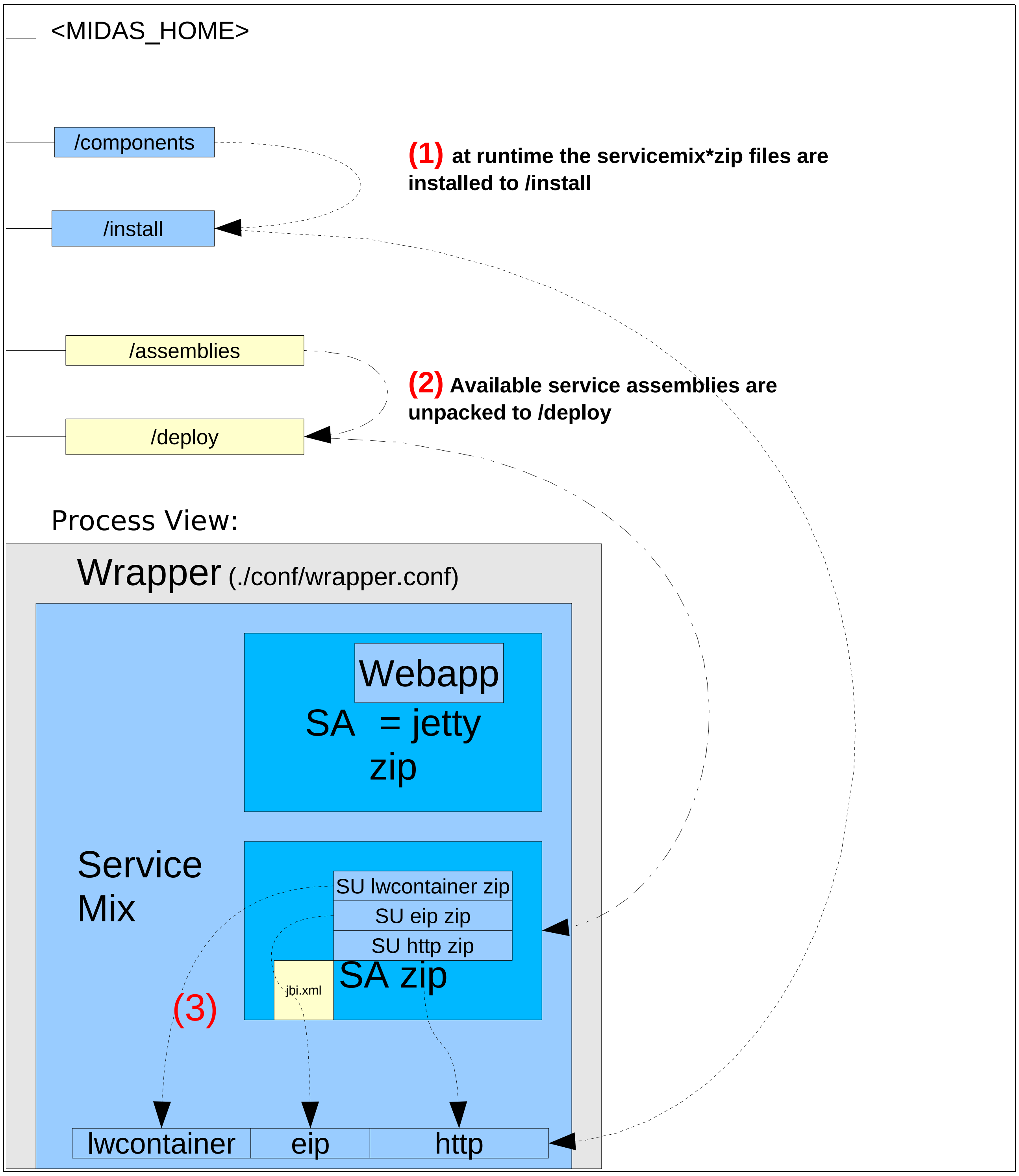
4.7.1 Layer mechanism

An important core technology used in CVP is the component container ServiceMix (http://www.servicemix.org). It provides an infrastructure for deploying so-called ***service assemblies***. A service assembly (SA) is a collection of ***service units*** (SU), which in turn provide a certain service. In CVP, there are service units like an OVO Adapter, a CVS server, an FTP server, ...

Both a service assembly and a service unit must conform to a well-defined interface. This allows a simple deployment of additional custom service assemblies and units.

Technically, a service assembly is a ZIP archive containing the SUs (other ZIP files) and resources belonging to a component and a file `jbi.xml` that tells the ServiceMix JBI (Java Business Integration) container the names of the components to deploy each of the Service Units to.

A SU may have specific configuration. For example, the OVO Adapter needs the opc_adm password. All SU configuration is stored in so-called ***layer configuration*** files, which are located in <MIDAS_HOME>/conf. See chapter for details.



Normally end users will not need to do anything related to this, however it may help understanding the internal architecture needed for troubleshooting.

4.7.2 CVP components

The CVP server is composed of multiple adapters and components that run in a enterprise service bus framework. Each adapter has a central configuration file that contains parameters to control its operations. Depending on the role of a CVP server (BackEnd or WebApplication) different adapters are installed.

Technically all adapters and core components are delivered as service assemblies (*-sa.zip files).

A OVO backend server has the following adapters

- ovoconfig - access and manipulate OVO configuration via database and API
- opccfg - access and manipulate OVO configuration via opccfg upload and download tools
- ovotask - execution of automation tasks
- file - access to files, execution of transfers

A web application server has the following adapters

- backend - CVP server registration
- usermgmt - CVP user model
- file - access to files, execution of transfers
- vcs - central version control system access

4.7.3 Web container

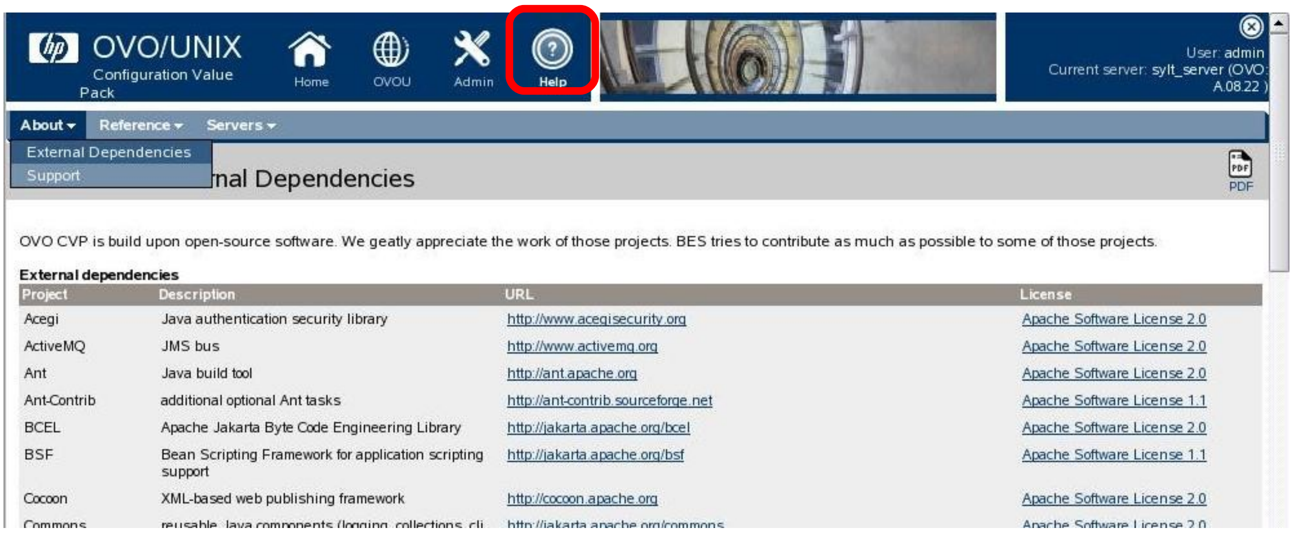
The web container used in the CVP is Jetty (http://www.mortbay.org). Standard deployment mechanisms of WAR files are used, therefore alternative web containers like Tomcat are theoretically possible.

This has not been tested yet and is currenly not supported therefore. However, supporting Tomcat is planned for future releases.

4.7.4 Open-Source software

Besides the components mentioned earlier already, CVP contains various other open-source software modules.

For an overview, the URLs referring to the open-source projects and the license terms can be reviewed as part of the CVP on-line help under `http://<WebApp>:<Port>/midas/help/en/external.html`:



nal Dependencies

OVO CVP is build upon open-source software. We geatly appreciate the work of those projects. BES tries to contribute as much as possible to some of those projects.

External dependencies

Project	Description	URL	License
Acegi	Java authentication security library	http://www.acegisecurity.org	Apache Software License 2.0
ActiveMQ	JMS bus	http://www.activemq.org	Apache Software License 2.0
Ant	Java build tool	http://ant.apache.org	Apache Software License 2.0
Ant-Contrib	additional optional Ant tasks	http://ant-contrib.sourceforge.net	Apache Software License 1.1
BCEL	Apache Jakarta Byte Code Engineering Library	http://jakarta.apache.org/bcel	Apache Software License 2.0
BSF	Bean Scripting Framework for application scripting support	http://jakarta.apache.org/bsf	Apache Software License 1.1
Cocoon	XML-based web publishing framework	http://cocoon.apache.org	Apache Software License 2.0
Commons	reusable Java components (logging, collections, cli	http://jakarta.apache.org/commons	Apache Software License 2.0

The list displayed above represents an extract only and is not complete – the actual product contains the complete list.

Alternatively, all open-source license files are available after installation under:

```
# ls -l <MIDAS_HOME>/docs/licenses
total 1236
drwxrwxr-x  2 root root  4104 2006-11-27 19:34 .
drwxrwxr-x  3 root root    72 2006-11-27 19:34 ..
-rwxrwxr-x  1 root root 26430 2006-11-27 18:00 LICENSE.3dm
-rwxrwxr-x  1 root root 11558 2006-11-27 18:00 LICENSE.acegi
-rwxrwxr-x  1 root root 11359 2006-11-27 18:00 LICENSE.activeio
[...]
```

5 Installation of CVP

The **CVP** installation allows you to install one or both of the following core software components:

- Server on the OVO platform (so-called ***BackEnd*** or ***BE***)
- Web Application (so called ***WebApp*** or ***WA***)

During installation, setup automatically detects which kind of the supported management framework is installed on the target system and will only allow installation of the matching components. For example, if you are installing **CVP** on an OVO for UNIX server, setup will only allow you to install OVO for UNIX or NNM components. For more information about installation requirements, see chapter 5.1 Installation Prerequisites.

5.1 Installation Prerequisites

The information in this section describes the steps you have to perform to prepare for the installation of **CVP**. The section contains information about the following topics:

-
-
- Control
-

5.1.1 Users and passwords

The information in this section applies to all types of installation except those in a high-availability (HA) cluster. If you are installing CVP on a system that is configured in a high-availability environment, make sure you review chapter 5.7 Installing in a HA cluster before starting the installation process.

Before starting to install CVP, it is important to verify the OVO passwords, for example:

● **opc_op**	OVO Oracle database user
● **opc_adm**	OVO administrator

TIP: If you do not know the passwords, see chapter 9.2 (Wrong or unknown OVO passwords) for information about how to determine and verify these settings.

Next, you need to create a local operating-system (OS) user called **`midas`** on both the OVO BackEnd server *and* on the system the WebApp will be installed on. The `midas` user is needed on both systems for version-control software CVS and for secure-data transfer using SSH (if you choose to use this option). Note that if either of the systems (BackEnd, WebApp) is configured in a high-availability environment, you need to add the user `midas` on each physical cluster node.

The `midas` user's home directory must be the CVP installation folder <MIDAS_HOME> (for example: `/opt/midas30`). Note that, although you *can* set a password for the `midas` user account, you do not *have* to; no password is needed for the midas user account to run CVP. If no password is set, the account is locked and no user will be able to use it to log in to the system illegally. However, depending on the OS, the user may need a password for SSH transfer, otherwise the SSH connection may fail.

To add the midas user on a UNIX-based system, run the following command:

```
# useradd -c „CVP user“ -d [CVP installation dir] midas
```

If you are installing CVP on a Windows system use the following command to add the user midas to the system:

```
> net user midas [password] /ADD /EXPIRES:NEVER /PASSWORDCHG:NO
 /HOMEDIR:[CVP installation dir]
```

The password-expiration option ensures that the account does not require any later maintenance.

ATTENTION: If you are using LDAP or NIS as user authentication mechanism, create a user account called "midas" with the correct home directory in that authentication system.

5.1.2 SSH data transfer

To transfer mass data between the systems hosting the CVP Backend and/or WebApp components, you can use either FTP or SSH. If you want to use SSH for secure communication, you will need to know the location of the ssh-keygen executable on the system you are currently installing. If SSH is already installed, determine the location of the ssh-keygen executable using the following command:

```
# which ssh-keygen
/usr/bin/ssh-keygen
```

If SSH is not installed yet, you can perform this set-up step later. See chapter 7.3 Secure Shell (SSH) for more information.

5.1.3 Version Control with CVS

You can configure CVP to store data in a revision-control system such as CVS. The default CVS repository resides on the WebApp system, which is where the CVS access will be managed.

If you plan to use a local CVS repository, the cvs command must be present on the WebApp system, too; if you plan to make use of a remote CVS repository, the cvs command is not required on the WebApp system. If you have not already done so, install the CVS software for the appropriate operating system. See chapter 7.2 Concurrent versioning system (CVS) for more information.

If CVS is already installed, determine the location of the cvs command using the following command:

```
# which cvs
/usr/local/bin/cvs
```

If CVS is not installed yet, you can perform this set-up step later. See 9.8 CVS configuration for more information.

5.1.4 Java runtime

The CVP installer includes a bundled JDK version 1.4.2 for all supported platforms. Setup installs the appropriate bundle for your platform. Only JDK version 1.4.2 and later are supported. JDK 1.5 can also be used.
Make sure to update the ./midas_env.sh or ./midas_env.bat configuration file accordingly with the correct JAVA_HOME path

5.2 System Requirements

The target systems should satisfy the following minimum requirements for running the individual CVP components:

	BackEnd	WebApp
Disk space	280 MB, location user-definable	600 MB, location user-definable
RAM	512 MB	512MB minimum, 1GB recommended

The numbers listed in the table above represent the minimum required to install and run the software, the more powerful the system, the better. Note that demand on system resources depends on the size and scale of the managed environment. Note, too, that the numbers displayed in the table above apply to CVP alone; other software such as OVO and Oracle will require additional resources.

During installation, about 600MB additional disk space will be needed temporarily to extract the **CVP** Installer itself.

To use the **CVP** Web Application, a recent version of a web browser is required. The following browsers have been tested and are supported:

- Microsoft Internet Explorer 6 and later
- Netscape 7 and later
- Mozilla 1.5 and later (including Mozilla Firefox 1.5)

Older browsers such as Netscape 4.7, 6, or Mozilla before 1.0 are not supported; these older browsers are known to have problems such as incomplete or wrong implementations of CSS and other technologies required by the **CVP** Web Application and will show a visually unattractive web page. Other browsers such as Opera have not been tested, but are likely to work with more or less restrictions.

NOTE: The built-in web browser which is bundled with the OVO Java GUI currently lacks some important capabilities and can be used only with some restrictions, particularly in the area of menu handling. If you are using the OVO Java GUI on Windows systems, it is recommended to enforce the use of the embedded Internet Explorer instead.

The following table shows which hardware platforms, operating systems, and high-availability software is supported with the current **CVP** release.

	OVO/UNIX 7	OVO/UNIX 8	WebApp
HP-UX PA-RISC	11.11, 11.23	11.11, 11.23,11.31 (later)	yes
HP-UX Itanium	n/a	11.23,11.31 (later)	yes
Solaris SPARC	8, 9	8, 9, 10	yes

Linux x86 32 bit	n/a	n/a	yes
Windows x86 32 bit	n/a	n/a	yes
ServiceGuard for HP-UX	11.15, 11.16	11.15, 11.16, 11.17	n/a
Veritas Cluster for HP-UX	3.5, 4.0	3.5, 4.0	n/a
Veritas Cluster for Solaris	3.5, 4.0	3.5, 4.0	n/a
Sun Cluster	3.0, 3.5	3.0, 3.5	n/a

5.3 OS and application patches

The following is a list of known problems which require certain OS patches or software packages to be installed.

- On RedHat Linux "Red Hat Enterprise Linux AS release 4 (Nahant Update 3)" and possibly similar OS versions the compatibility package "Legacy Software Development" is needed.
  Otherwise, the installer may be unable to open the DISPLAY even if set correctly. See chapter 5.8.1 Display problems.
- On Suse Linux 10.1, Suse Enterprise Server/Desktop 10 and FedoraCore5 systems, the following error may occur:

```
[...]
Configuring the installer for this system's environment...
awk: error while loading shared libraries: libdl.so.2: cannot open shared
object file: No such file or directory
[...]
```

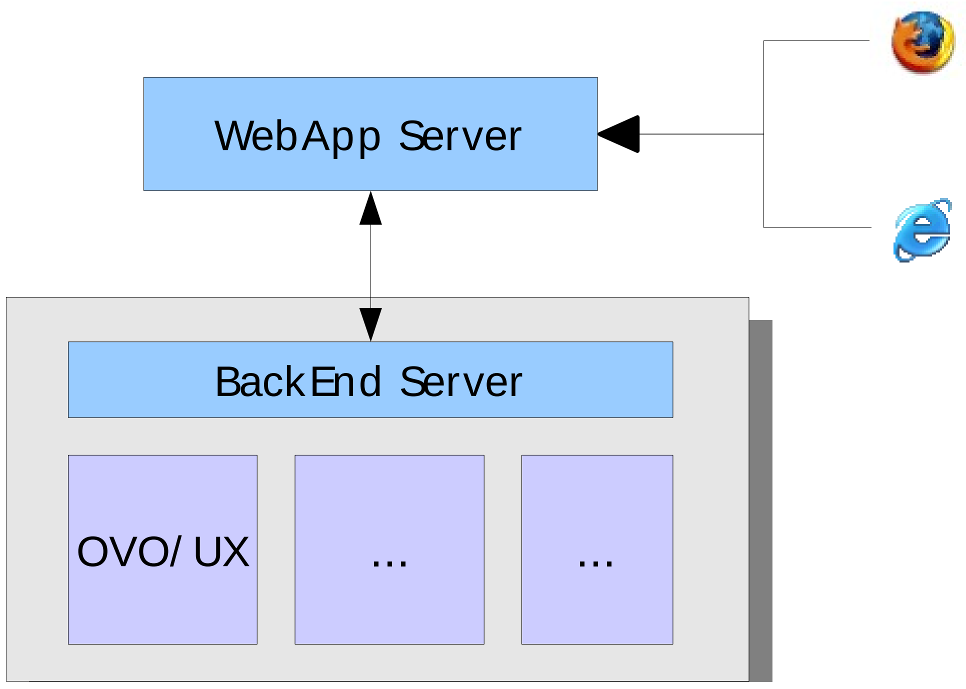
  The reason is, that the stated OS versions no longer provide the LD_ASSUME_KERNEL variable which is required by the InstallAnywhere installer.
  To work around this problem use the dedicated Linux Suse 10.1 installer.

5.4 Installing CVP

You install **CVP** using a GUI-based tool created with the InstallAnywhere technology. The **CVP** GUI-based setup installs the following core components:

- **CVP** Server (BackEnd)
- **CVP** Web Application (WebApp)

The **CVP** server must be installed on each management-framework server (e.g. an HP OVO management server) whose configuration data you want to manage with **CVP**; the **CVP** web application should be installed on a separate server to take system load off the OVO Server. For more information about system requirements, see 5.1 Installation Prerequisites. The following figure shows the basic architecture of the CVP modules:



The boxes underneath *BackEnd Server* (for example OVO/UX) stand for the IT Management products integrated in CVP. Currently **CVP 3.0** does not support any other products except OVO/UNIX. Future releases may also work with other products like OVO/W or NNM.

Although the interactive graphical installer attempts to discover the current values for the most important **CVP** Server configuration parameters during the installation process, you still need to perform some manual steps.

To install **CVP**, you have to perform the following high-level steps, which are explained in detail in the subsequent sections:

1. Login as **`root`** user and mount the product media or copy the installer image to the target system. If you are installing on a remote system, you might need to set and export the DISPLAY variable.
2. Start the installer program that matches the operating system installed on the target system.
3. Select the components to install
4. Choose and set the installation path for the <MIDAS_HOME> directory
5. Configure the CVP BackEnd server
    1. Oracle database configuration for OVO
    2. HP OpenView Operations configuration
    3. **CVP** BackEnd port configuration
6. Configure the **CVP** Web Application
    1. CVS repository configuration
    2. **CVP** Web Application HTTP port configuration
    3. **CVP** Document Server configuration

    4. **CVP** Web Application start-up configuration

7. Configure server start-up
8. Perform a pre-installation check of port numbers, server names, disk space, and installation path
9. Launch the automatic installation of the **CVP** software

The installer attempts to determine most parameters by itself and, in most cases, it should be sufficient to confirm the provided default values. However, you should carefully review the parameters that are suggested and check if they are correct. Note that some values, such as passwords, cannot be determined and have to be supplied manually during the course of the installation.

For preparation of the installation a check-list is provided in chapter .

If you enter an incorrect value for an important parameter and you need to change the parameter value later, most values can be modified easily after the installation has completed.

5.4.1 Starting the CVP installer

The installation of the **CVP** software has to be performed as user root on UNIX systems, because the installer enables system startup and shutdown configuration, which requires root permissions. Setup checks to ensure root privileges are available before starting the installation procedure. If you are installing on a Windows system, make sure the user has administrator privileges. For remote installations on UNIX systems, remember to export the DISPLAY to your workstation by using the `xhost +` command, as necessary

If a **CVP** product is already installed on the system, make sure that both the **CVP** Server and the **CVP** Web Application have been stopped before continuing with the installation.

- Use for MIDAS 2.x
  `# <MIDAS_HOME>/midas.sh stop all`
- Use for CVP 3.x
  `# <MIDAS_HOME>/midas.sh stop`

You should also use the following command on UNIX systems to check that no CVP processes such as tomcat or java that belong to CVP are running:

```
# ps -ef | grep -i midas
```

Make sure you remove the previous version of CVP before proceeding with the installation of the new software version. For more information about removing and upgrading CVP, see chapter 5.9 Upgrading MIDAS 2.x and 5.10 De-installing CVP.

Make sure that the product media is mounted (e.g. on UNIX systems to `/mnt`) or the installer image has been copied to some temporary location. Then, execute the installation program as follows:

- UNIX: `# /mnt/<OS>/VM/install.bin`
- Windows: `...\install.exe`

where <OS> stands for the operating system running on the machine where you start the installer.

If you download the installer from the internet, make sure you name the binary appropriately (e.g. `MIDAS3.0-Configurator-HPUX11-PA.bin`). Although the name is not important for technical reasons, it will simplify later identification.

Make sure there is enough disk space in the `/tmp` directory (on UNIX systems) and the `Temp` folder (on Windows systems). If there is not enough free disk space, you will get the following message:

```
bash-2.05b# ./install.bin Preparing to install...
WARNING: /tmp does not have enough disk space!
       Attempting to use / for install base and tmp dir.
WARNING! The amount of / disk space required to perform this
installation is greater than what is available. Please free up at least
512262 kilobytes in / and attempt this installation again. You may also
set the IATEMPDIR environment variable to a directory on a disk
partition with enough free disk space. To set the variable enter one of
the following commands at the UNIX command line prompt before running
this installer again:
- for Bourne shell (sh), ksh, bash and zsh:
   $ IATEMPDIR=/your/free/space/directory
   $ export IATEMPDIR
- for C shell (csh) and tcsh:
   $ setenv IATEMPDIR /your/free/space/directory
```

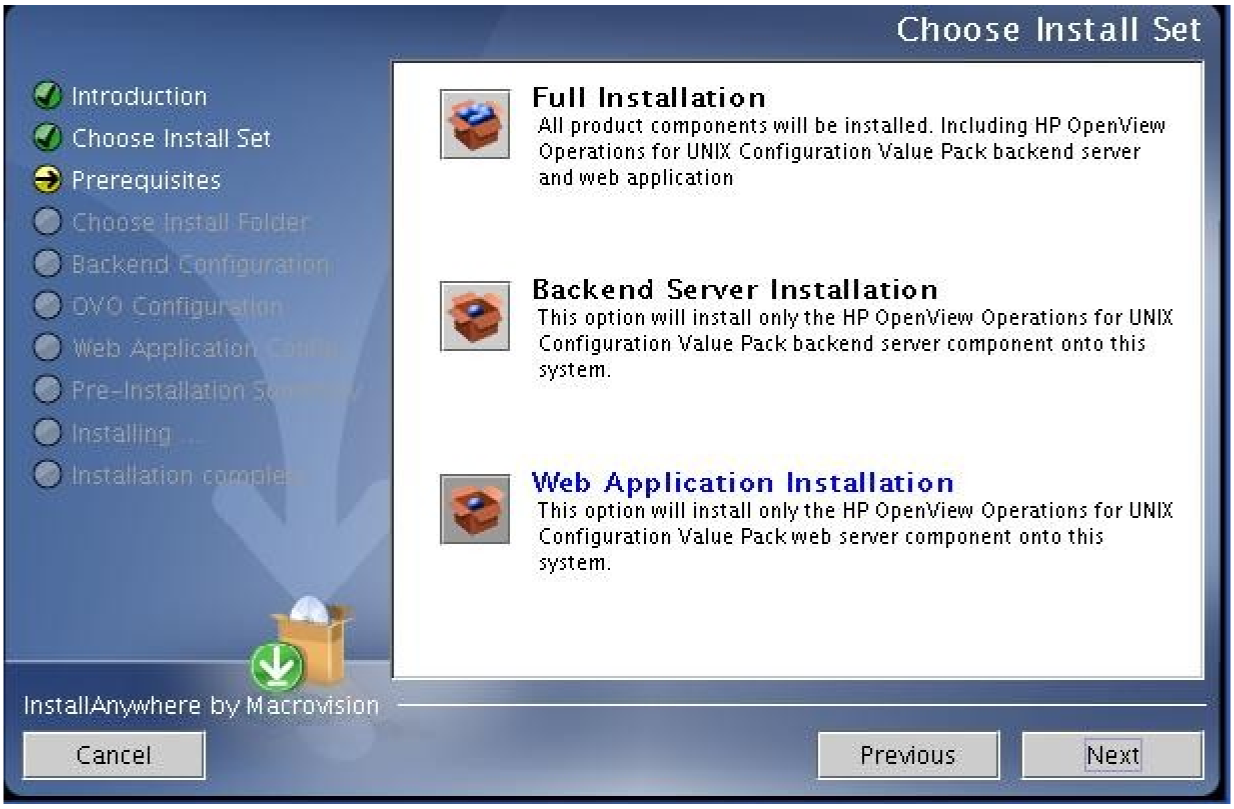
If you the message illustrated in the figure above, set the IATEMPDIR variable to a location with enough disk space for the installation process and restart the installer, for example:

```
# export IATEMPDIR=/your/free/space/directory
# /mnt/<OS>/VM/install.bin
```

The installer unpacks itself and after a few moments the actual installer GUI starts. Note that this may take some time, please be patient and do not interrupt the execution.

5.5 Install Set Selection

The installer displays the installable **CVP** options and requires you to choose one:



Full Installation
All product components will be installed. Including HP OpenView Operations for UNIX Configuration Value Pack backend server and web application

Backend Server Installation
This option will install only the HP OpenView Operations for UNIX Configuration Value Pack backend server component onto this system.

Web Application Installation
This option will install only the HP OpenView Operations for UNIX Configuration Value Pack web server component onto this system.

The items displayed depend on the system you are installing on and which components (OVO/UNIX) are available.

- **Full Installation**
  Installs both a **CVP** Server and a CVP Web Application on the same system with all available components
- **Backend Server installation**
  Installs only the CVP Server on the OVO BackEnd server with its typical components.
- **Web Application installation**
  Installs only a **CVP** Web Application with all typical components (CVS adapter, etc.).

In productive environments it is recommended to install the WebApp on a separate system and to avoid Full Installations to take system load off the OVO Management Server.

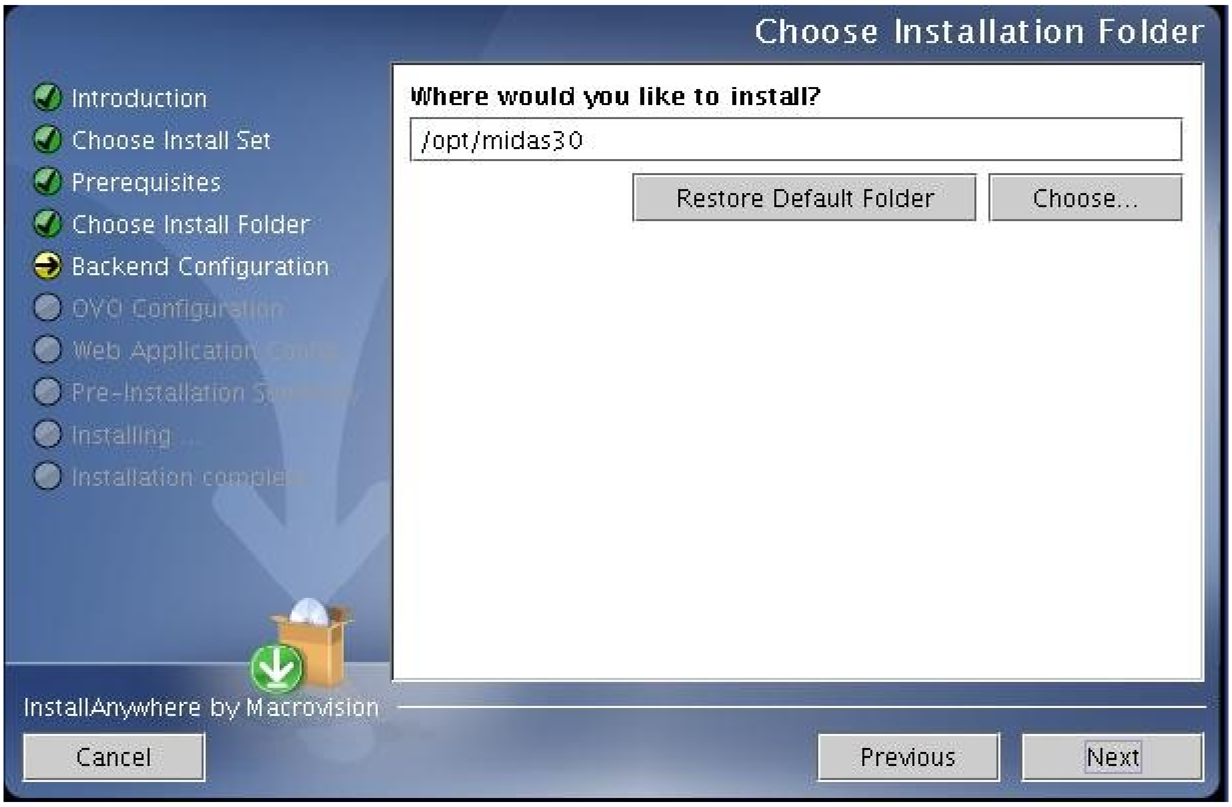
NOTE: The CVP BackEnd Server can only be installed on systems where HP OVO is present.

Example: If you run the installer on a Windows or Linux system, then the Full Installation and BackEnd Server Installation items will not be available, since there is no such OVO component on these platforms.

NOTE: The order of the screenshots presented here, follows the Full Installation.

5.5.1 Installation directory

In this step of the installation procedure, you either confirm the default location for installation or, alternatively, enter the directory path, where the **CVP** components should be installed:



Where would you like to install?

/opt/midas30

The default locations for installation are as follows:

- UNIX: `/opt/midas30`
- Windows: `C:\midas30`

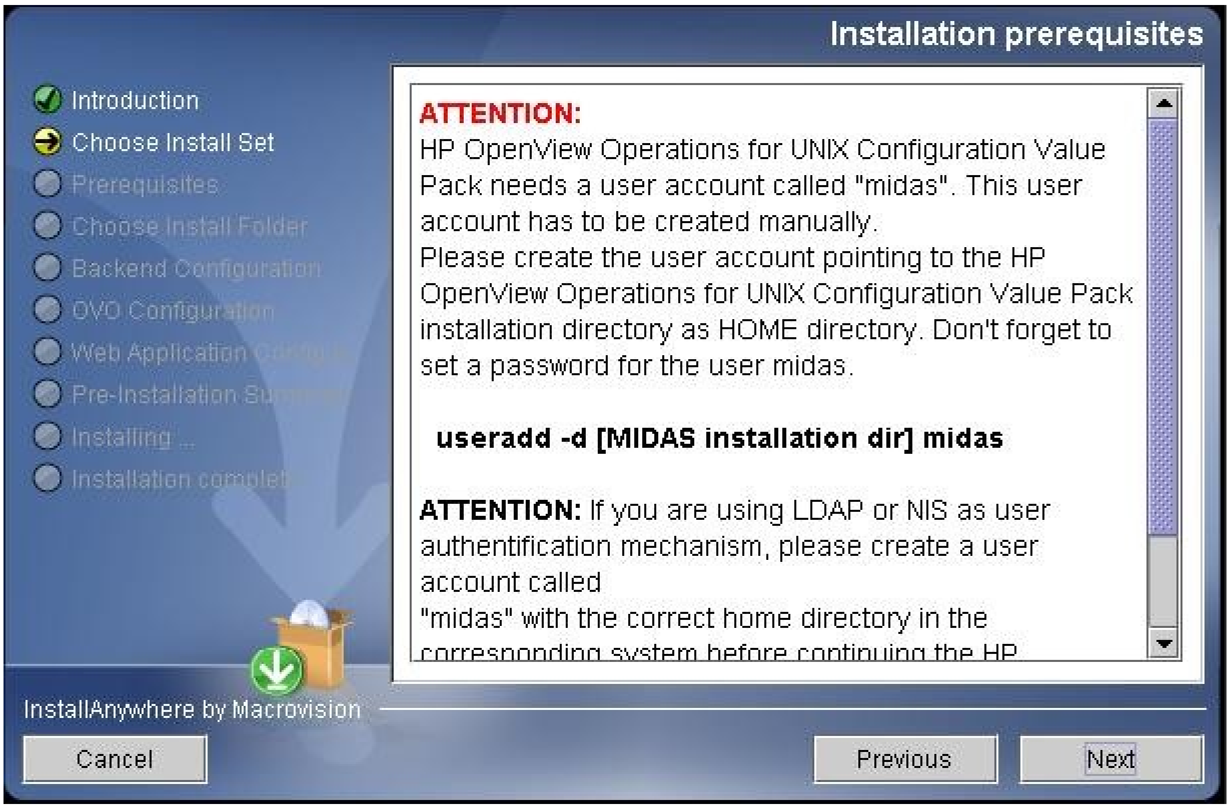
It is recommended to accept the suggested value. Installations on Windows to D:\ etc. are possible.

CAUTION: Do NOT use blanks in the program paths.

NOTE: When installing in a HA cluster, make sure this path is located on a shared disk which is part of the OVO cluster package.

5.5.2 Installation pre-rerequisites

As explained in the pre-installation requirements (chapter 5.1 Installation Prerequisites) you must have created an user `midas` on all **CVP** servers (both BackEnd and WebApp).

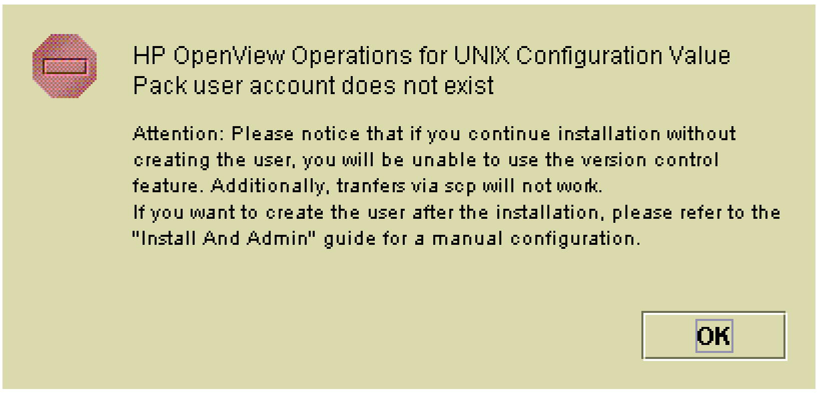


ATTENTION:
HP OpenView Operations for UNIX Configuration Value Pack needs a user account called "midas". This user account has to be created manually.
Please create the user account pointing to the HP OpenView Operations for UNIX Configuration Value Pack installation directory as HOME directory. Don't forget to set a password for the user midas.

```
useradd -d [MIDAS installation dir] midas
```

ATTENTION: If you are using LDAP or NIS as user authentification mechanism, please create a user account called
"midas" with the correct home directory in the corresponding system before continuing the HP

The installer checks whether this user exists – if not an error message is displayed:



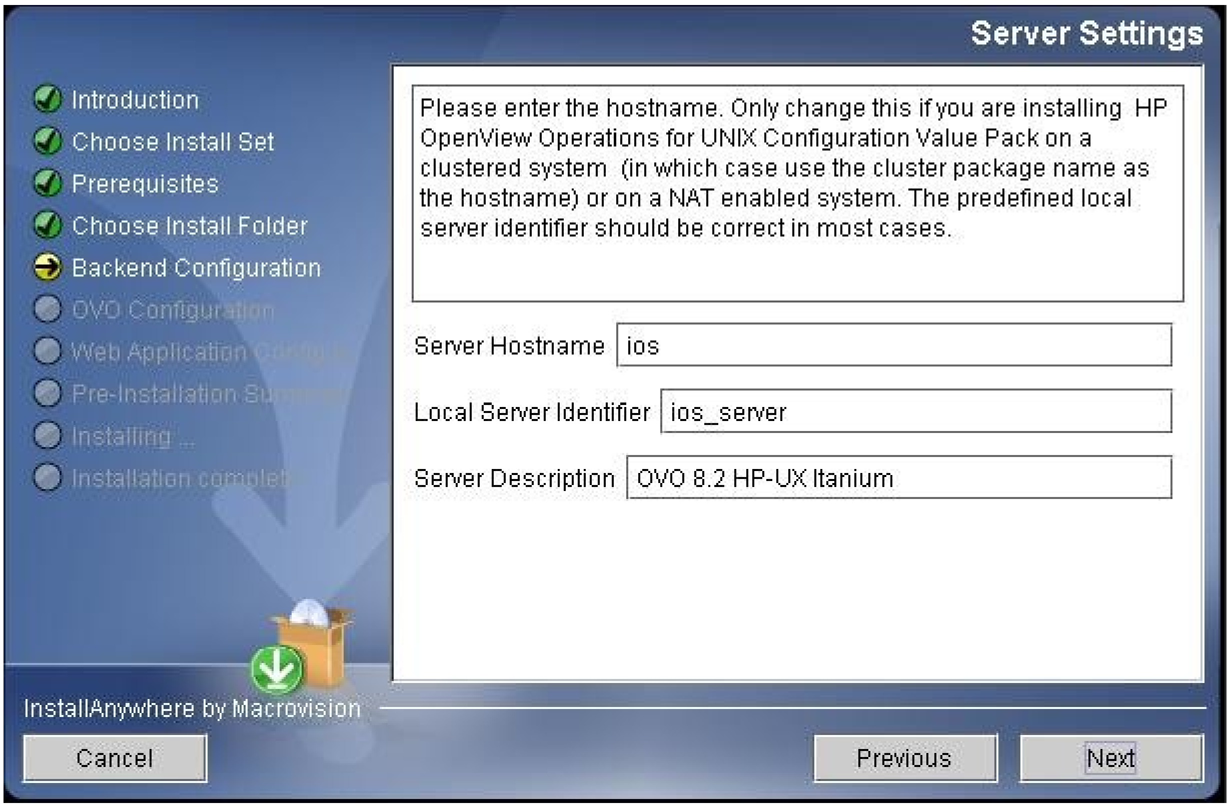
HP OpenView Operations for UNIX Configuration Value Pack user account does not exist

Attention: Please notice that if you continue installation without creating the user, you will be unable to use the version control feature. Additionally, tranfers via scp will not work.
If you want to create the user after the installation, please refer to the "Install And Admin" guide for a manual configuration.

You can proceed and create the `midas` user account later if needed. If neither CVS nor SSH will be used on the local CVP server, the `midas` user account is not needed at all and you can ignore this error.

5.5.3 The CVP server identifier

In this step of the installation procedure you have to confirm the name of the system hosting the CVP server. Each CVP server has an ID which can be specified together with a host name and a description. Setup is usually able to determine these values (except description) without user input.



Please enter the hostname. Only change this if you are installing HP OpenView Operations for UNIX Configuration Value Pack on a clustered system (in which case use the cluster package name as the hostname) or on a NAT enabled system. The predefined local server identifier should be correct in most cases.

Server Hostname: ios

Local Server Identifier: ios_server

Server Description: OVO 8.2 HP-UX Itanium

The parameters are:

- ***Server Hostname***: This is the local host name. Specify a name or IP address which can be resolved and reached from all involved systems.
  Default is the output of the command hostname, for example:

```
# hostname
ios
```

IMPORTANT: On HA clusters use the virtual host name, which represents the HA package running the OVO Server (which includes **CVP**)!

- ***Local Server Identifier***: This is used later within the WebApp to access the different BackEnd servers. You have to give the server you are currently installing a unique name. Each server (whether a WebApp or BackEnd) must have its own local identifier; the value you assign as a local identifier must not contain any spaces.
  Example: shortname_server; OVO_Production_server; Test_server; Webapp

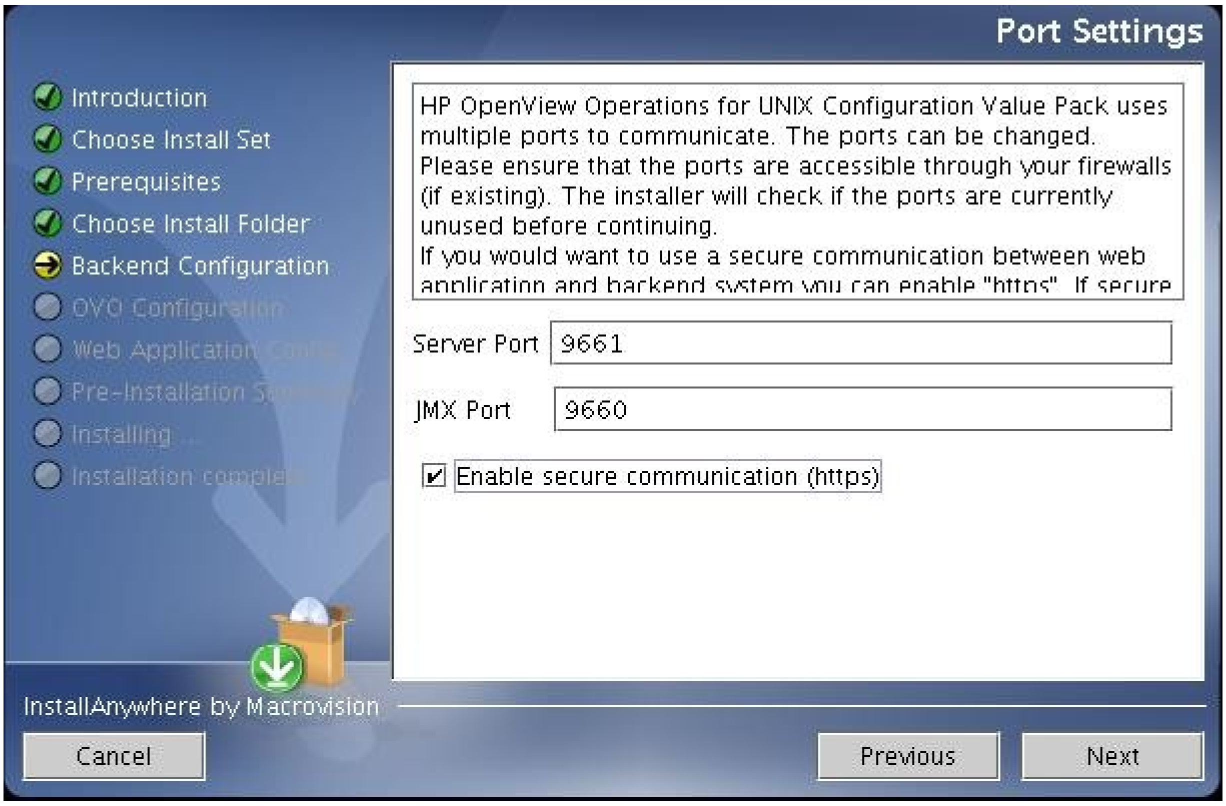
NOTE: The recommended naming schema is the *Server Hostname* as entered in the installer screen extended by "*_server*".

- ***Server Description***: Enter a short description, for example: "Germany HQ,

Production Server (OVO 8.2)”

5.5.4 Server ports

In this step of the installation procedure you have to confirm the default assignment of port numbers for the **CVP** server or specify new ones. The setup process assigns ports to each **CVP** server (both WebApp and BackEnd).



Port Settings

HP OpenView Operations for UNIX Configuration Value Pack uses multiple ports to communicate. The ports can be changed. Please ensure that the ports are accessible through your firewalls (if existing). The installer will check if the ports are currently unused before continuing.
If you would want to use a secure communication between web application and backend system you can enable "https". If secure

Server Port: 9661

JMX Port: 9660

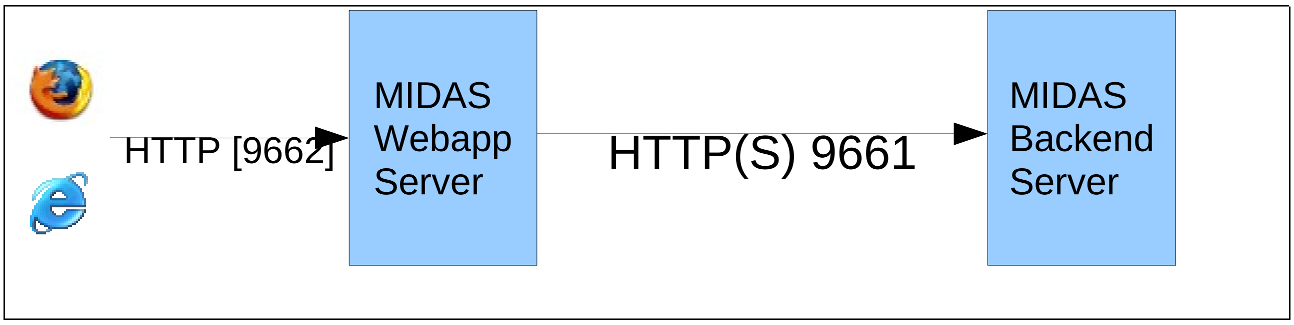
☑ Enable secure communication (https)

The required ports are:

- The ***Server Port*** (default 9661) is used for the main HTTP(S) communication between WebApp and BackEnd systems.
- The ***JMX Port*** (default 9660) is used for troubleshooting purposes only. It is not referenced by any remote CVP components.
- For HTTPS communication between CVP servers, enable the check-box labeled ***Enable secure communication (https)***. This affects
  - the registration of a BackEnd at WebApps (choose SSL in this case. See chapter 6.3 Registering a BackEnd for details). This applies to *all* WebApps connecting to this BackEnd
  - local communication through the `midas.sh` command. No extra configuration is necessary for this purpose.

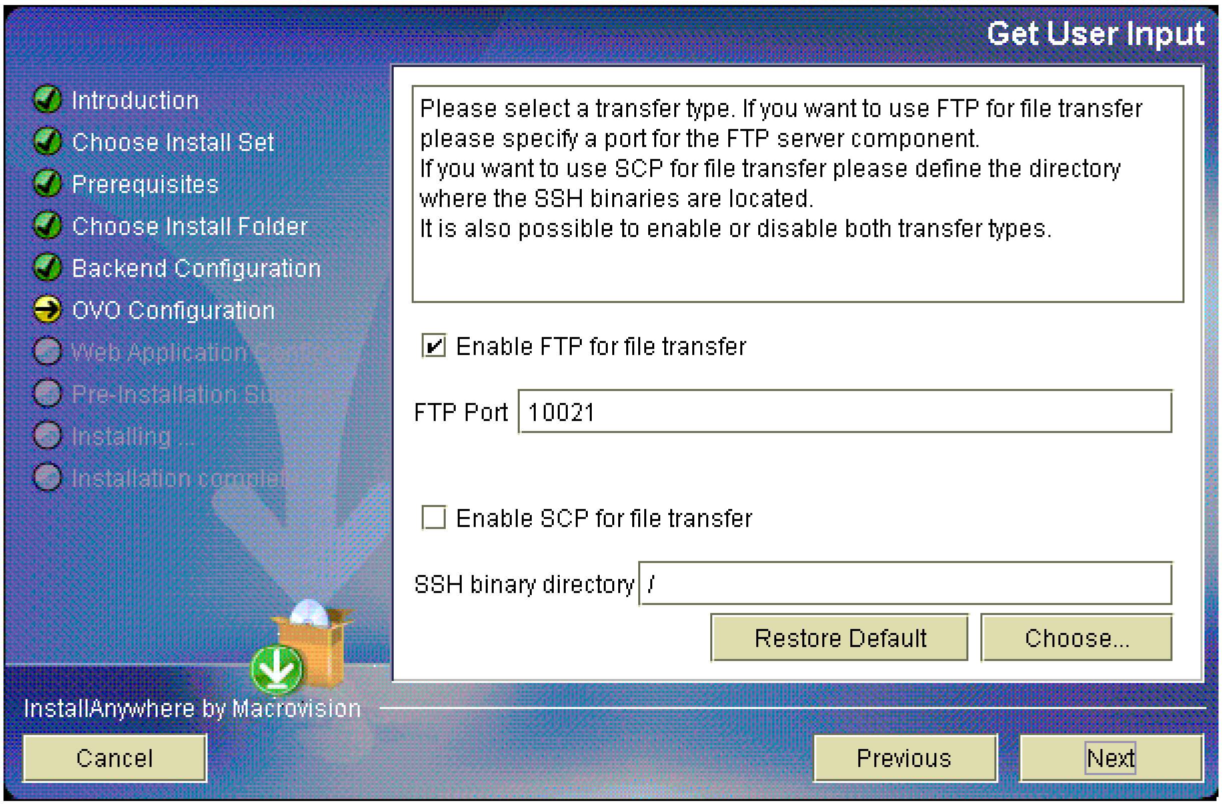
  For more details about HTTPS communication see chapter 9.7.2 Using HTTPS.

  If a firewall exists between any related WebApp and BackEnd systems, make sure the appropriate firewall rules are configured to allow this traffic. See chapter 9.7.4 Using CVP in firewall environments for details:



5.5.5 FTP / SSH transfer type

In this step of the installation procedure you have to specify the tool you want to use to transfer the large amounts of configuration data between systems. All main CVP communication is performed using HTTP(S). However, large volumes of data traffic between any WebApp or BackEnd system, for example: to exchange downloaded configuration data between a development and a production OVO server, are moved with FTP or SSH. Select which transfer types you want to enable for the BackEnd system currently being installed; FTP is the default method, but you can also use SSH.



Please select a transfer type. If you want to use FTP for file transfer please specify a port for the FTP server component.
If you want to use SCP for file transfer please define the directory where the SSH binaries are located.
It is also possible to enable or disable both transfer types.

[✓] Enable FTP for file transfer

FTP Port 10021

[ ] Enable SCP for file transfer

SSH binary directory /

If you are unsure about making the required choice, you can accept the default values and configure this later.

If you choose the SCP option for secure copy via SSH, you need to enter the location, where the `ssh-keygen` executable can be found. Note that the `ssh-keygen` command is needed to generate SSH keys; it is not required for performing the data transfers themselves.

To locate the `ssh-keygen` command on a UNIX system, use the `which` utility:

```
# which ssh-keygen
```

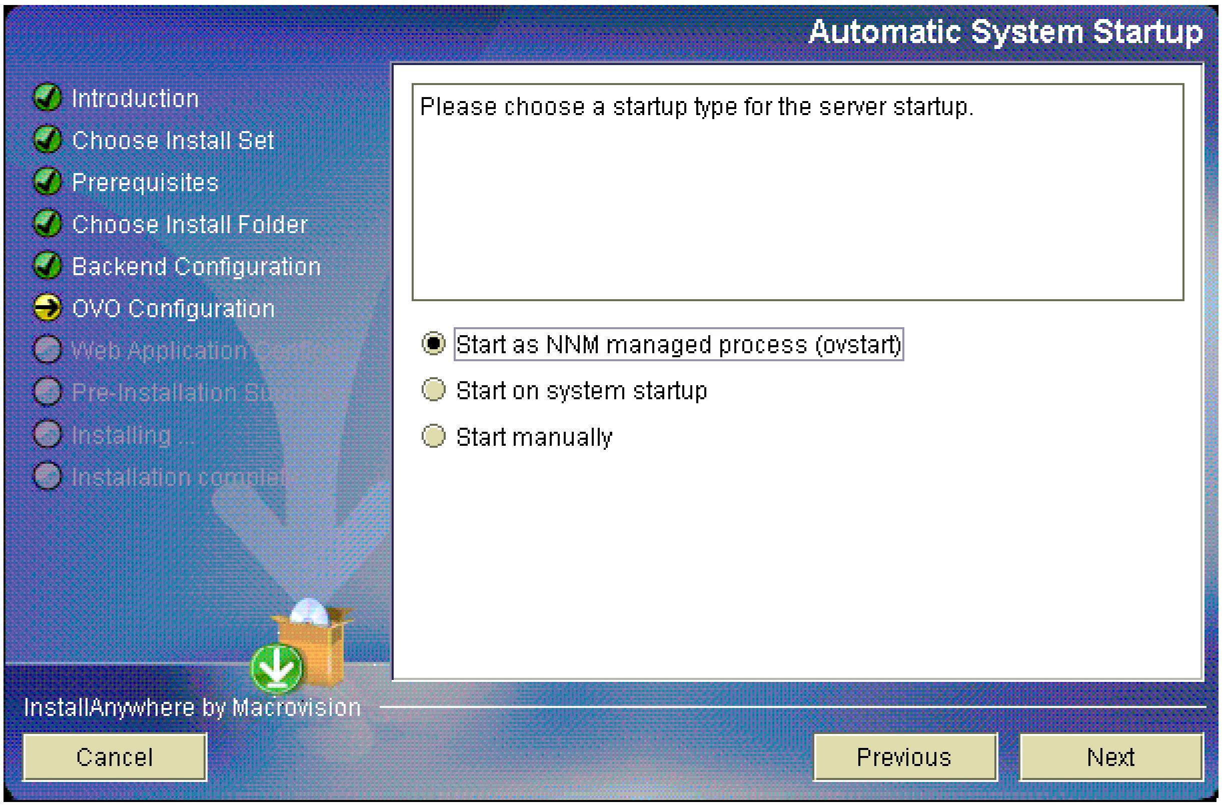
```
/usr/bin/ssh-keygen
```

In this example, the path to be entered in the SSH binary directory in the installer GUI is: `/usr/bin`

See chapter 9.9.2 Configuring SSH transfer for more detailed information about configuring SSH.

5.5.6 Automatic server startup

In this step of the installation procedure you have to select the way in which you want the CVP server to start-up:



Please choose a startup type for the server startup.

[✓ Start as NNM managed process (ovstart)]
Start on system startup
Start manually

You can choose to start the CVP server by one of the following methods:

- ***Start as NNM managed process (ovstart)*** – standard `ovstart/ovstop` integration. This will be available only, if `ovstart` exists (e.g. not for standalone WebApps on systems without the OV Platform product present).
- ***Start on system startup*** – some OS boot scripts (typically in `/etc/init.d`) will be created. On Windows, CVP will be registered as Windows service.
- ***Start manually*** – no automated startup for CVP; this may be required for cluster installations.

NOTE: In a CLUSTER set-up do not use the OS-integrated startup. Configure CVP in a way that CVP starts after Oracle and OVO. This can be done either with the `ovstart/ovstop` commands or by providing some other cluster-control script.

See chapter 9.6 Running CVP in a HA cluster for details.

5.5.7 Oracle settings (BackEnd only)

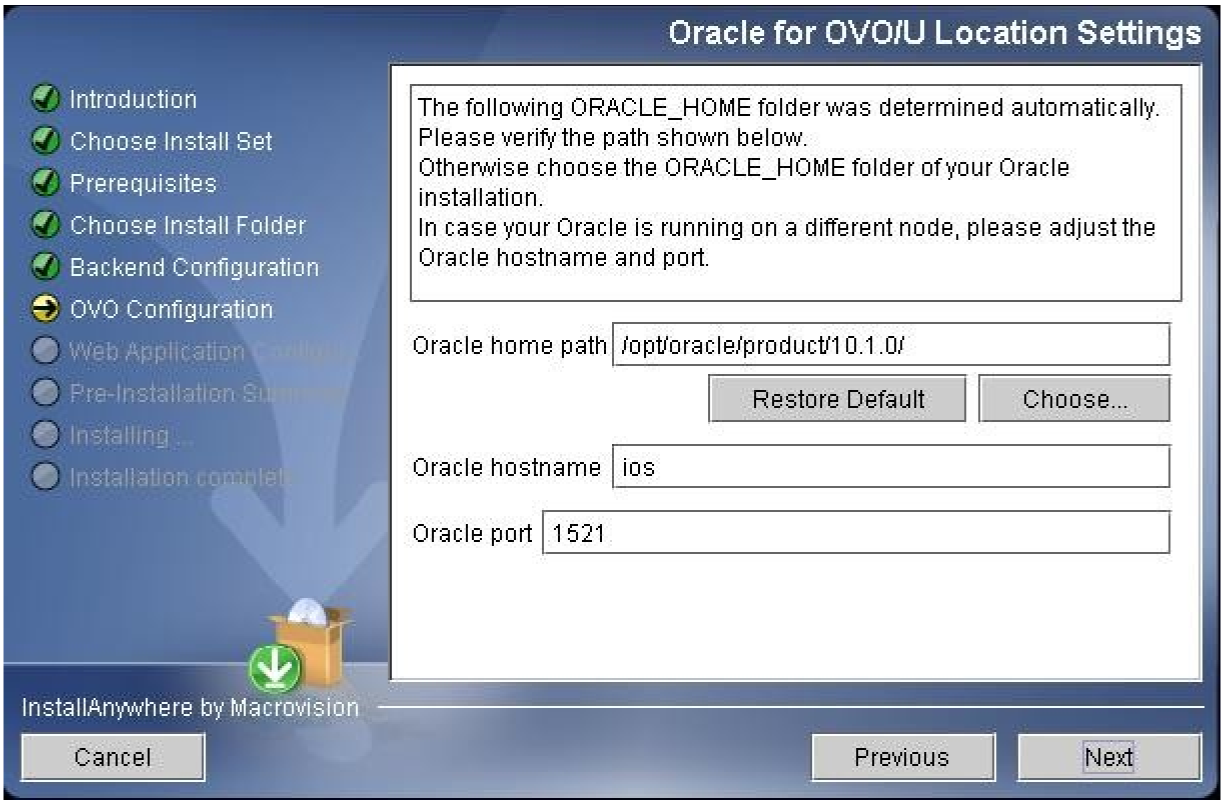
In this step of the installation procedure you have to confirm one or two details about the Oracle database that OVO uses. The initial CVP installation dialog allows you to choose between installing either a CVP BackEnd server, a CVP WebApp or both. If you choose to install a CVP BackEnd server (either on its own or as part of a full installation) in the install set dialog, the installer displays dialogs that ask for the HP OVO server configuration parameters.

HP OVO uses an Oracle instance to store its configuration data. CVP retrieves the

OVO configuration data from that Oracle database instance (read-only). The installer attempts to detect all Oracle settings automatically by reading the file `/etc/opt/OV/share/conf/ovdbconf`. The `ovdbconf` file is the main OVO configuration file defining database access.

Generally, there should no need to change these settings, unless Oracle is located on a remote DB server. However, you should carefully verify the detected directory and change appropriately, if required.

The following figure shows the dialog used to specify the location of the OpenView database:



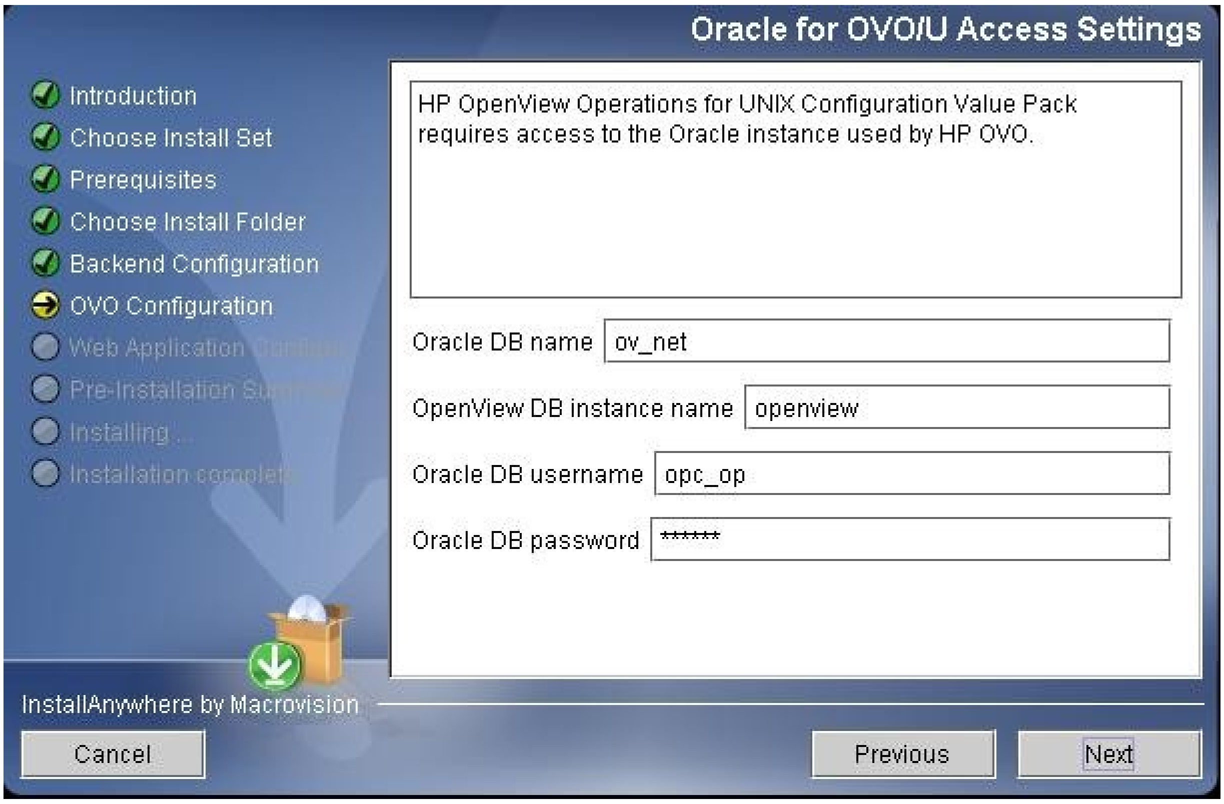
The following ORACLE_HOME folder was determined automatically.
Please verify the path shown below.
Otherwise choose the ORACLE_HOME folder of your Oracle installation.
In case your Oracle is running on a different node, please adjust the Oracle hostname and port.

Oracle home path: /opt/oracle/product/10.1.0/

Oracle hostname: ios

Oracle port: 1521

The subsequent screen asks for parameters required to access the Oracle database instance:


Oracle for OVO/U Access Settings

HP OpenView Operations for UNIX Configuration Value Pack requires access to the Oracle instance used by HP OVO.

Oracle DB name: ov_net

OpenView DB instance name: openview

Oracle DB username: opc_op

Oracle DB password: ******

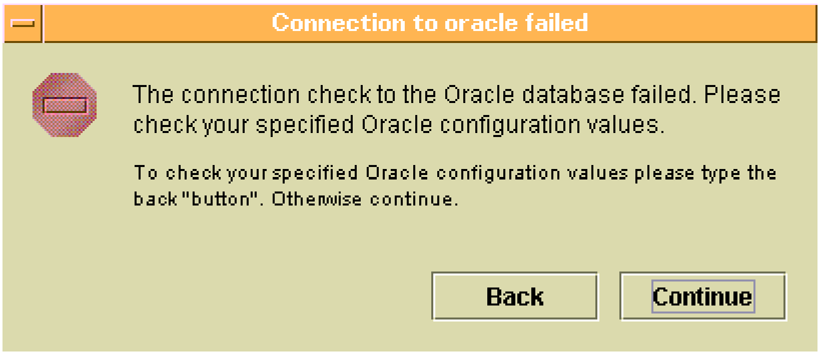
Check that the Oracle DB name and OpenView DB instance name are correct.

You also need to specify the user name and password for the Oracle user which will be used to connect to the database. By default, the user name is **`opc_op`**; alternatively, the Oracle user **`opc_report`** (has read-only rights only) is fine as well with marginal restrictions. See chapter 9.1.1 Oracle connectivity for details. Generally, any Oracle user can be specified as long as the Oracle user has sufficient privileges to read OVO database objects.

The passwords for the both Oracle users (`opc_op` and `opc_report`) are set during HP OVO server installation. In the installer screen you must supply exactly this password.

CAUTION: Do not confuse the Oracle opc_op account with the accounts of the same name in OVO for UNIX and the UNIX operating system itself. All three exist and are completely different accounts!

The installer checks if the password you supply for the Oracle user is correct and, if it is not, displays the following warning:



Connection to oracle failed

The connection check to the Oracle database failed. Please check your specified Oracle configuration values.

To check your specified Oracle configuration values please type the back "button". Otherwise continue.

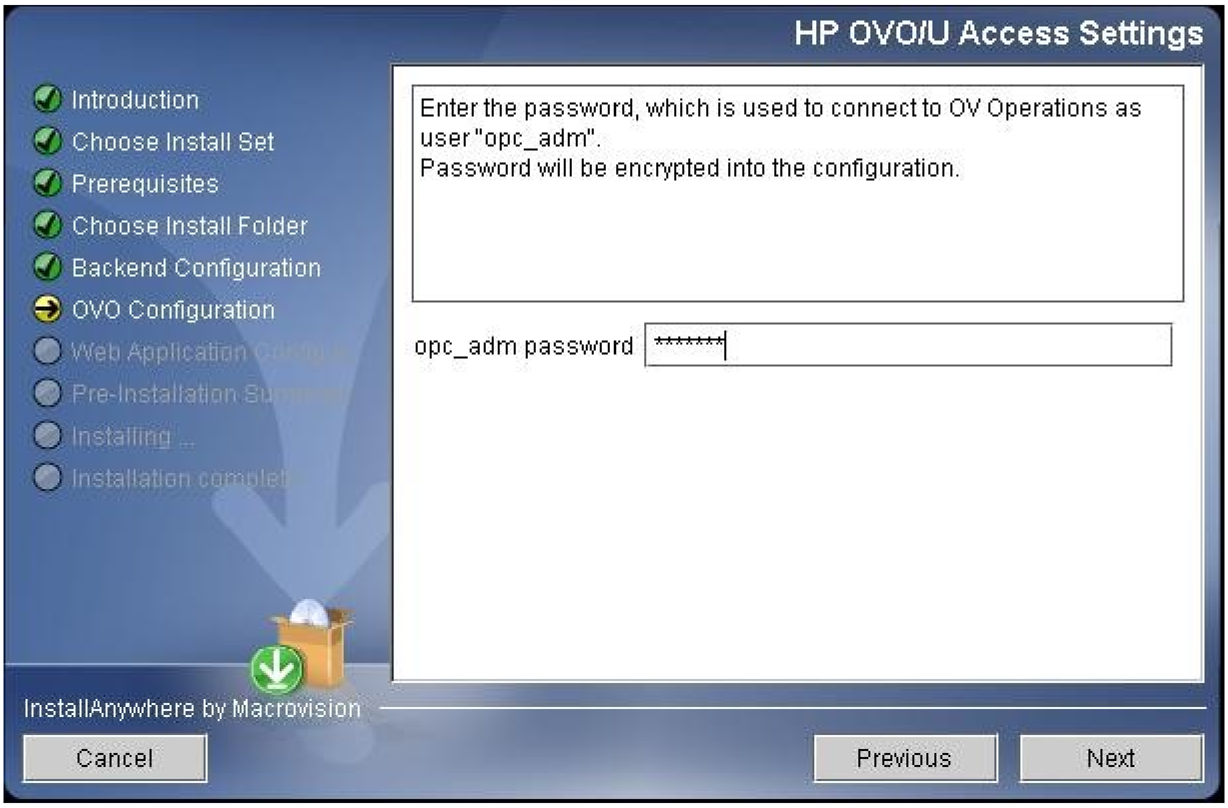
If you ignore this warning and proceed anyway, the CVP BackEnd server will fail to connect to the database and major CVP functionality will be unavailable. However, it is possible to finish the installation first and then re-configure the DB access with the correct information.

The password will not be shown in the installer GUI as you type it; the password is stored in encrypted form in the CVP Server configuration for authentication with the HP OVO server at runtime.

IMPORTANT: If password of the Oracle user is unknown, check chapter 9.2 Wrong or unknown OVO passwords. If the password is incorrect you can continue with the installation, but the CVP will not start up correctly.
For more information about configuring passwords later, see chapter 9.1.2.1 Encrypting and configuring passwords.

5.5.8 OVO API access (BackEnd only)

To perform many tasks, the CVP BackEnd connects to the OVO configuration API. In order to access the API, CVP BackEnd has to log on to OVO as the OVO administrative user **opc_adm** with the correct password as configured in OVO. You have to supply the password manually at this point.

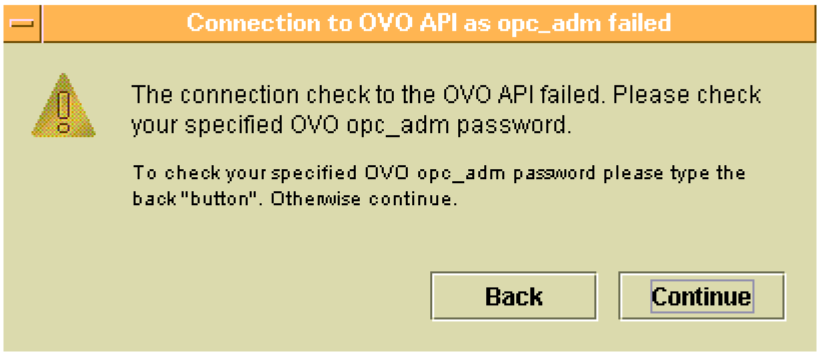


Enter the password, which is used to connect to OV Operations as user "opc_adm".
Password will be encrypted into the configuration.

opc_adm password *******

The password will not be shown in the installer GUI as you type it; the password is stored in encrypted form in the CVP Server configuration for authentication with the HP OVO server at runtime.

Note that the setup process checks the opc_adm password; if the supplied password is not correct, the following warning appears:


Connection to OVO API as opc_adm failed

The connection check to the OVO API failed. Please check your specified OVO opc_adm password.

To check your specified OVO opc_adm password please type the back "button". Otherwise continue.

If you ignore this warning and proceed anyway, the CVP BackEnd server will fail to connect to the OVO server and major CVP functionality will be unavailable. However, it is possible to finish the installation first and then re-configure the OVO access with the correct information.

IMPORTANT: If password of the OVO user opc_adm is unknown, check chapter 9.2 Wrong or unknown OVO passwords. If the password is incorrect you can continue

with the installation, but the CVP server will not start up at correctly.
For more information about configuring passwords later, see chapter 9.1.2.1 Encrypting and configuring passwords.

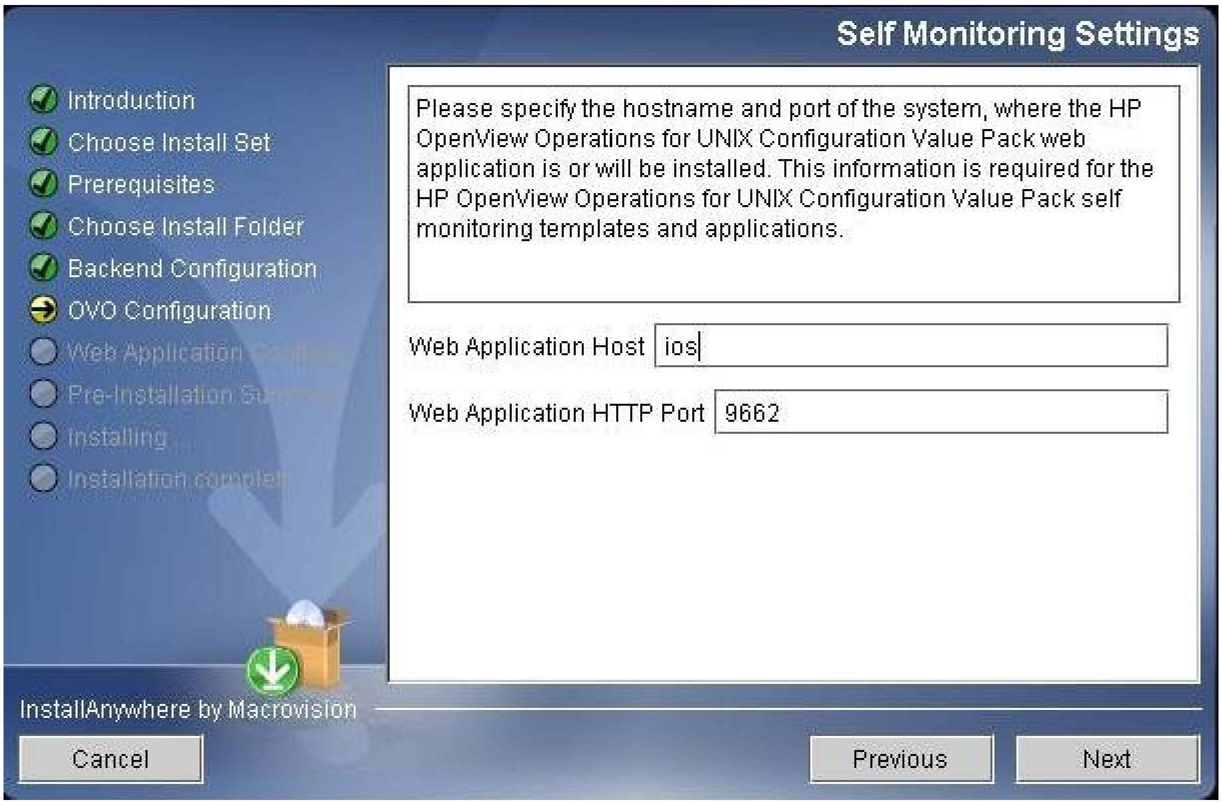
5.5.9 Initial WebApp (BackEnd only)

When installing a standalone BackEnd, an initial WebApp has to be specified (in a full installation this screen will not appear, since the local system acts in this role). This information will be used for

- self monitoring (see below)
- access permission to the local built-in FTP Server (for details see chapter 9.9.1 Configuring FTP transfer).

CVP contains something like a SMART Plug-In providing some self-management capabilities. It consists of the usual elements (policies, policy groups, node groups, profiles, applications, ...).

This set of OVO configuration objects will be automatically loaded during installation. To make them work immediately, please specify the required parameters. The host name and port of the CVP WebApp (same as used for working with the regular CVP GUI) will be used to construct URL Applications in OVO, so the CVP GUI can be started from operator's Application Desktop.



Please specify the hostname and port of the system, where the HP OpenView Operations for UNIX Configuration Value Pack web application is or will be installed. This information is required for the HP OpenView Operations for UNIX Configuration Value Pack self monitoring templates and applications.

Web Application Host: ios
Web Application HTTP Port: 9662

If unsure, you may leave these values unchanged and update the OVO Applications manually later.

- ***Web Application Host*** – Enter here the host name of the system where the CVP WebApp is installed. Specify a host name which is resolvable and reachable from all systems, where such an OVO Application will be launched by an OVO user.

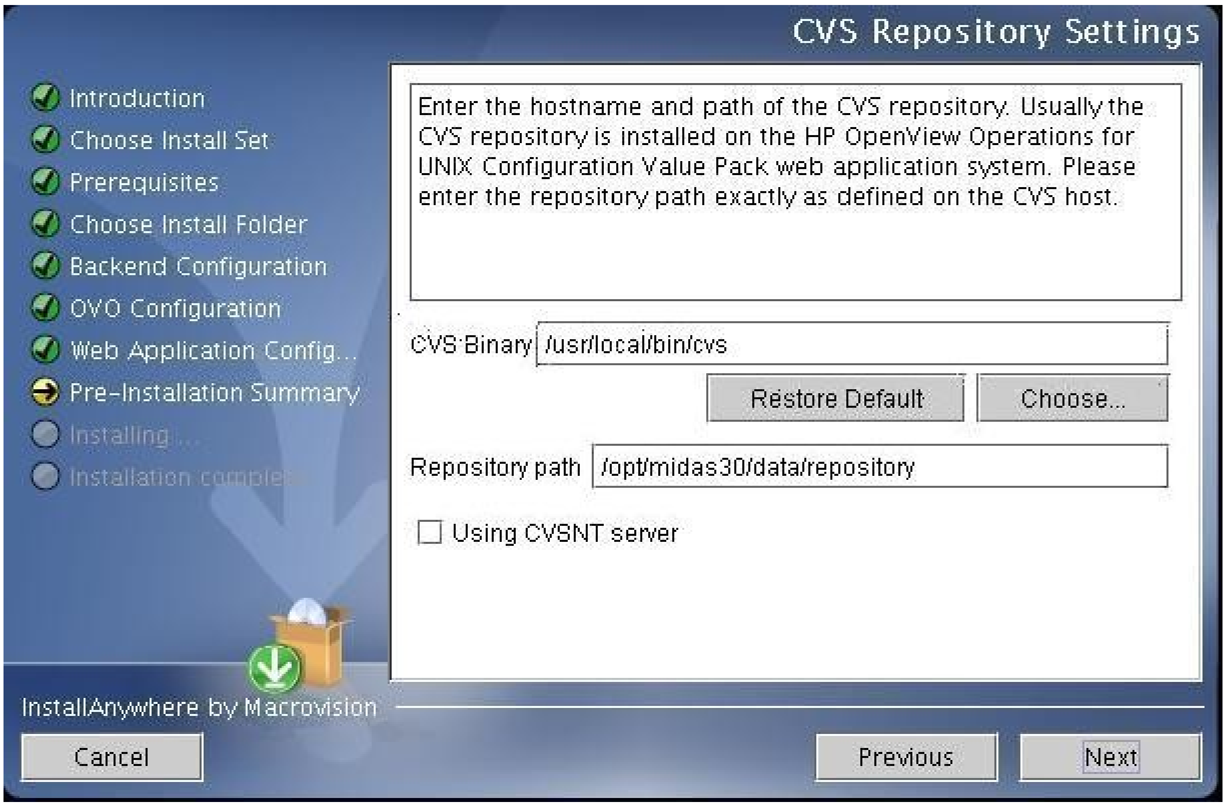
- ***Web Application HTTP Port*** – The port where the WebApp to be used is listening.

NOTE: If you choose the Full-Installation option, the self-monitoring screen will not appear – the local system runs a WebApp, which will be used for this purpose.

5.5.10 CVS configuration (WebApp only)

To control versions of configuration data you save and download, you can use the open-source software CVS (concurrent versioning system - http://www.cvshome.org). By default, a CVS repository is installed as part of the CVP Web Application. The CVS software itself must be installed by the user as described in chapter 7.2 Concurrent versioning system (CVS).

Parameters related to CVS can be specified here. If CVS is not to be used initially, just leave the values unchanged and proceed by clicking *Next*.



Enter the hostname and path of the CVS repository. Usually the CVS repository is installed on the HP OpenView Operations for UNIX Configuration Value Pack web application system. Please enter the repository path exactly as defined on the CVS host.

CVS Binary: /usr/local/bin/cvs

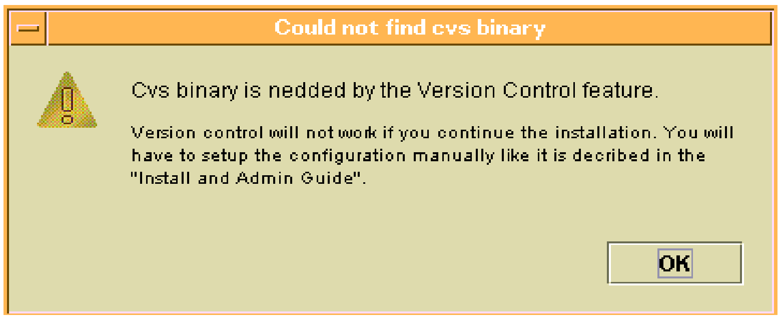
Repository path: /opt/midas30/data/repository

☐ Using CVSNT server

The CVS repository can be located locally on the CVP WebApp or anywhere else (for non-default repositories see chapter 9.8 CVS configuration for details). If a local repository is to be used, the CVS software is needed on the CVP WebApp system. Make sure, this is installed correctly.

- ***CVS Binary*** – enter the path to the CVS binary (e.g. `/usr/bin/cvs`).
- ***Repository path*** – the CVS repository will be created here. The default location is within the CVP installation directory `<MIDAS_HOME>/data/repository`. Usually there is no need to change this.
- ***Using CVSNT server*** – enable this option if you are using CVSNT as CVS implementation.

If the specified cvs command does not exist the following will be displayed:



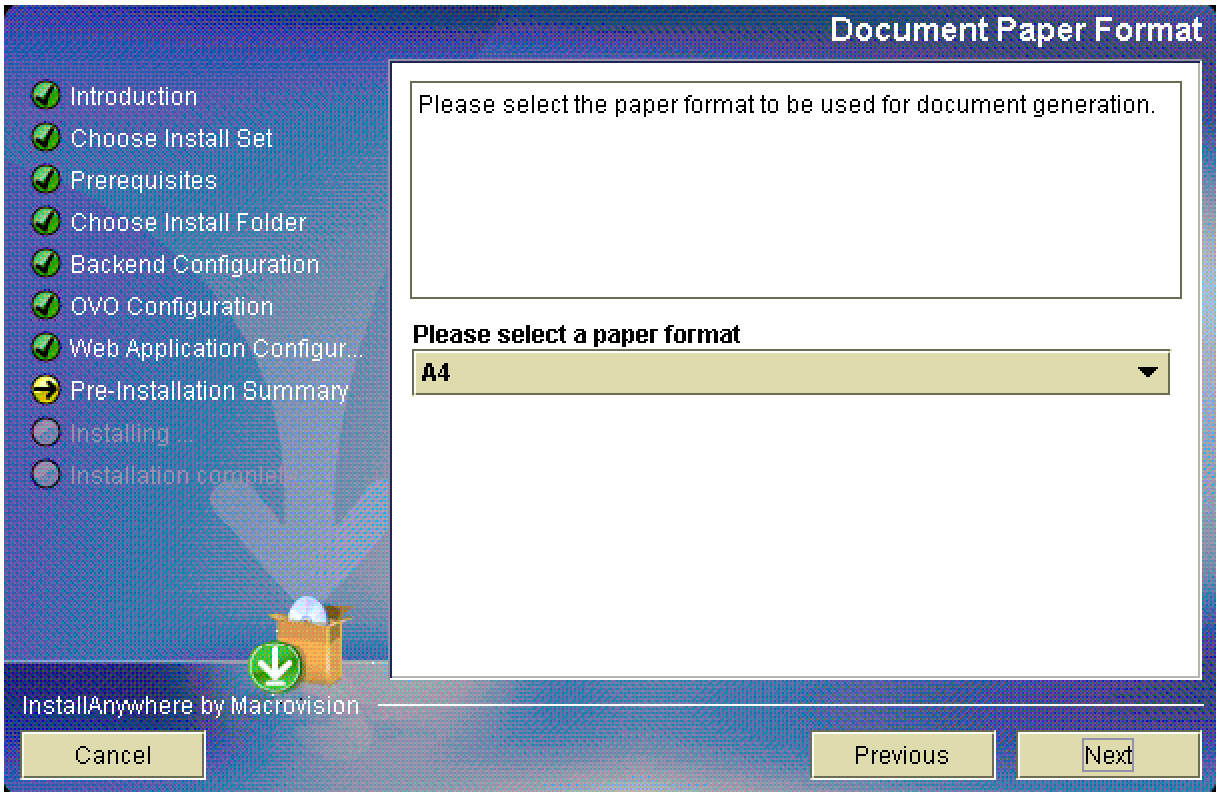
Cvs binary is nedded by the Version Control feature.

Version control will not work if you continue the installation. You will have to setup the configuration manually like it is decribed in the "Install and Admin Guide".

You can continue with the installation, but you will have to configure CVS later if you want to make use of it to manage the configuration data you store.

5.5.11 Document paper format (WebApp only)

Choose the default paper format you want to use for the documents you generate with CVP; you can choose between A4 or US letter.



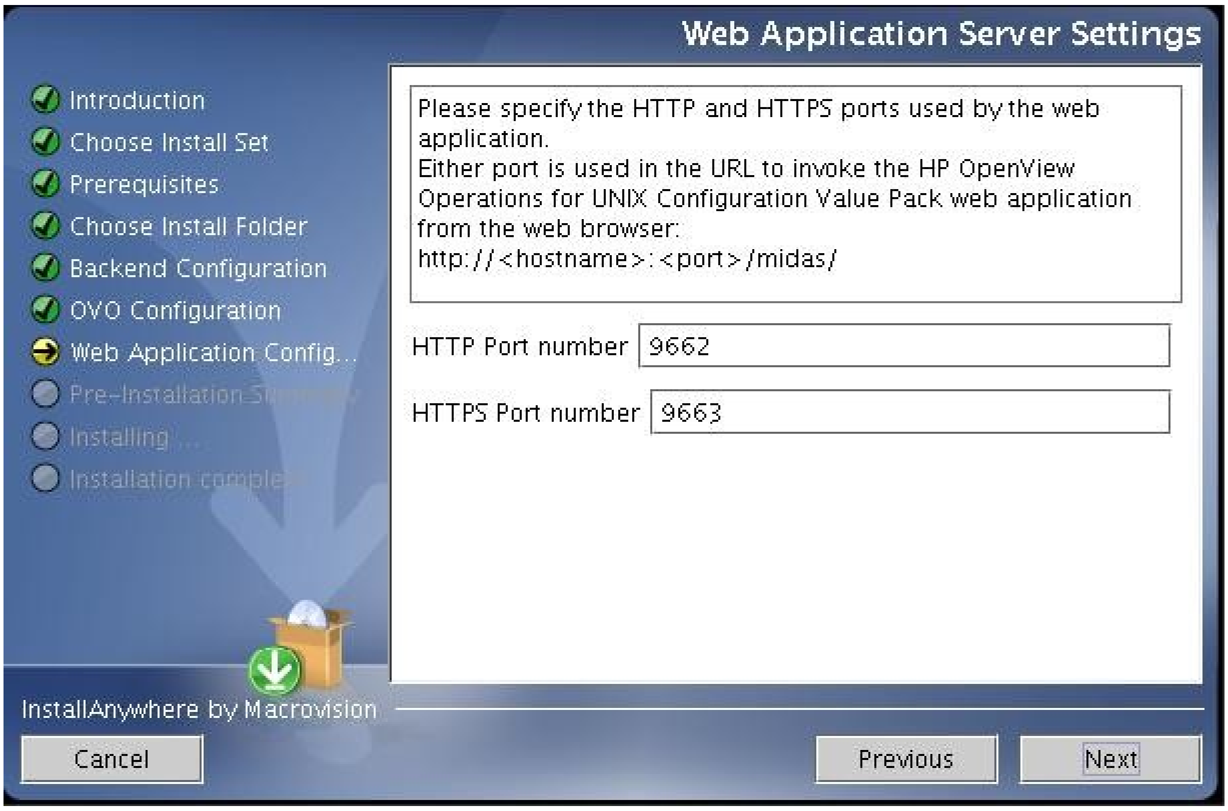
Please select the paper format to be used for document generation.

Please select a paper format

A4

5.5.12 WebApp port configuration (WebApp only)

If the CVP Web Application has been selected for installation, the installer will ask you for the port on which the CVP Web Application can be reached by a browser:



Please specify the HTTP and HTTPS ports used by the web application.
Either port is used in the URL to invoke the HP OpenView Operations for UNIX Configuration Value Pack web application from the web browser:
http://<hostname>:<port>/midas/

HTTP Port number 9662

HTTPS Port number 9663

The default ports are the following: 9662 for HTTP ; 9663 for HTTPS.

The installer tries to check if the specified ports are already in use. If the ports are already in use, the installer will ask for a different port before proceeding with the installation. Choose another port number.

If you enter a non-default port number, you also have to specify the alternate port number in the URL, which is used to invoke the CVP Web Application from the web browser. With the default ports, this is:

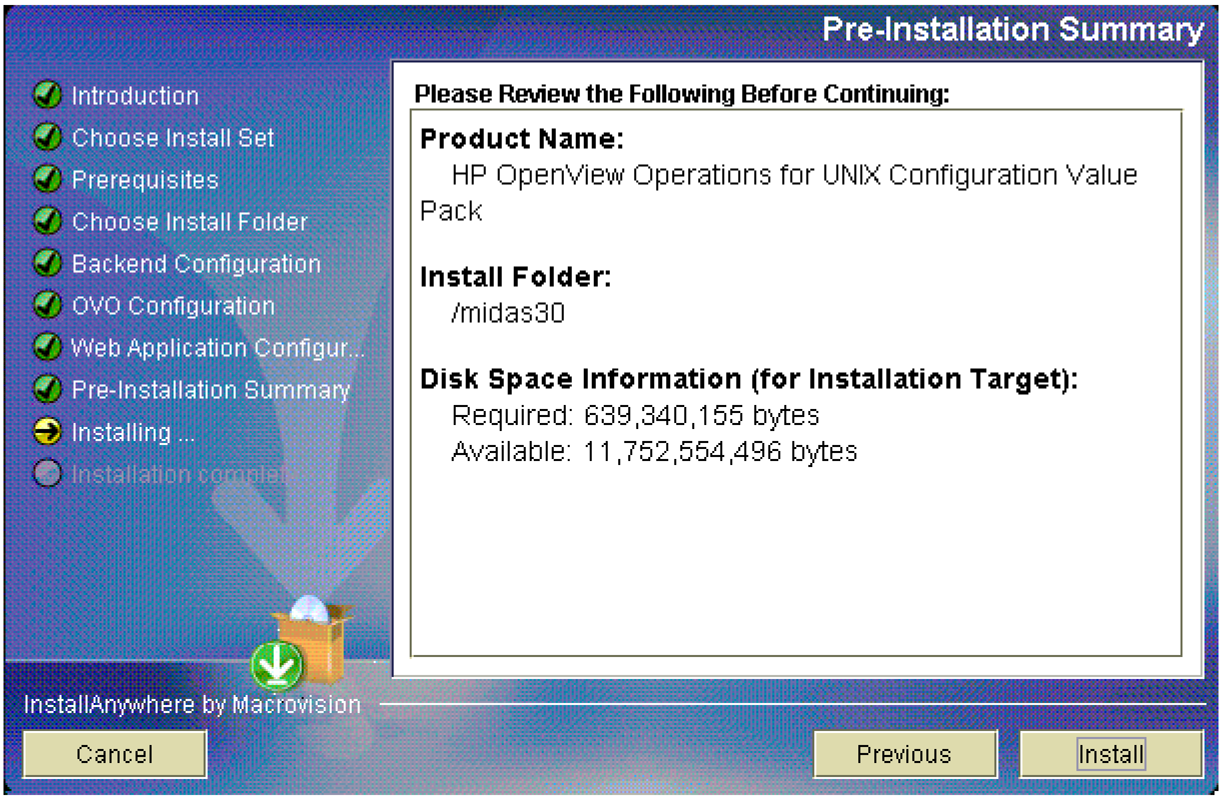
```
http://<webapphost>:9662/midas
```

or, if HTTPS is desired:

```
https://<webapphost>:9663/midas
```

5.5.13 Pre-Installation summary

Before the actual installation process takes place you will be presented with a summary of all the choices you have made and the information you have provided in the various screens and dialogs so far. Take this opportunity to check the details you supplied, in particular the installation folder and disk space requirements:



Please Review the Following Before Continuing:

Product Name:
HP OpenView Operations for UNIX Configuration Value Pack

Install Folder:
/midas30

Disk Space Information (for Installation Target):
Required: 639,340,155 bytes
Available: 11,752,554,496 bytes

Click *Install* to start the installation; click *Previous* to return to any dialog where you need to check or change parameters.

5.5.14 Installation of software

If you click Install, the installer starts the actual installation and displays the progress so that you can see what is happening and why.

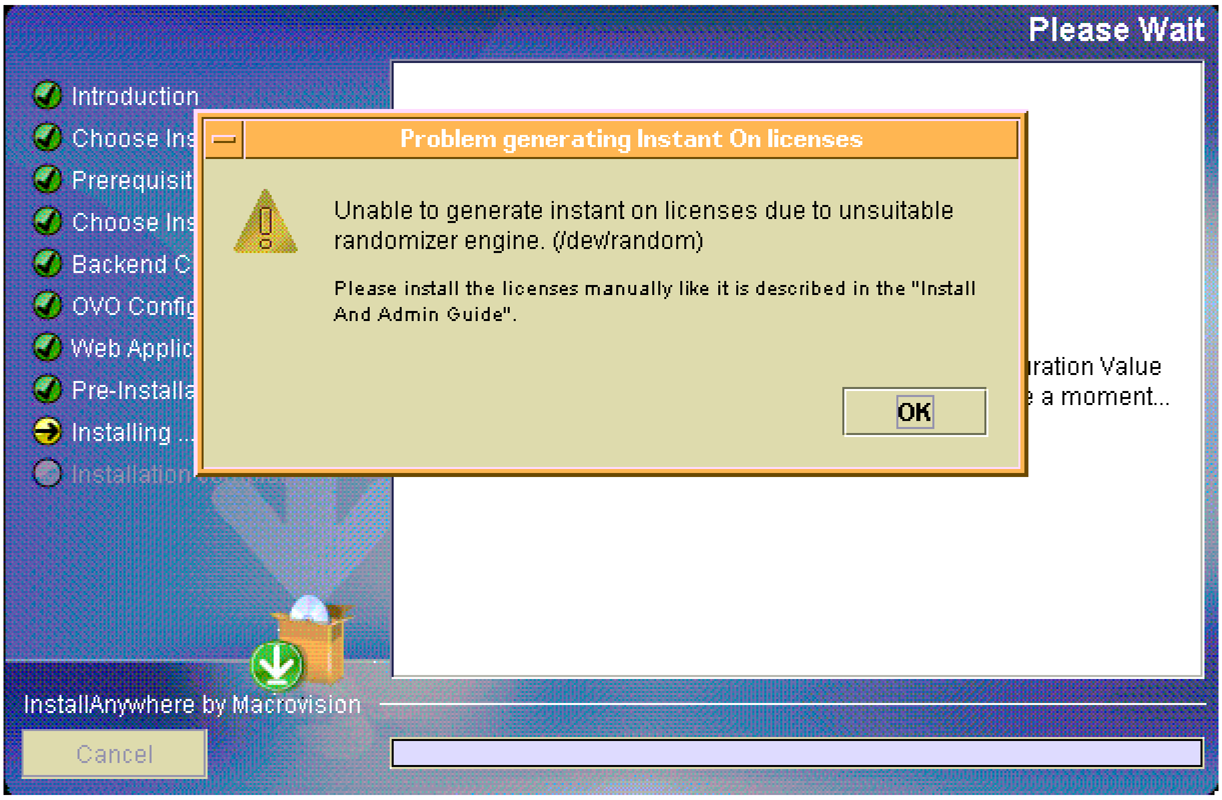
The installation procedure consists of two phases:

- installation of the software itself (binaries, configuration files, documentation, ...)
- configuration of the CVP components based on the parameters provided during the installation

The configuration phase of the installation includes the execution of various scripts and commands. The progress of the CVP software installation is shown in the installer panel, using progress-meter boxes.

5.5.14.1 /dev/random bug

A known defect in the Java JRE on HP-UX 11.x systems (typically PA-RISC only) may cause the generation of the instant-on license fail. If this problem occurs, the following dialog indicates what has happened.



Problem generating Instant On licenses

Unable to generate instant on licenses due to unsuitable randomizer engine. (/dev/random)

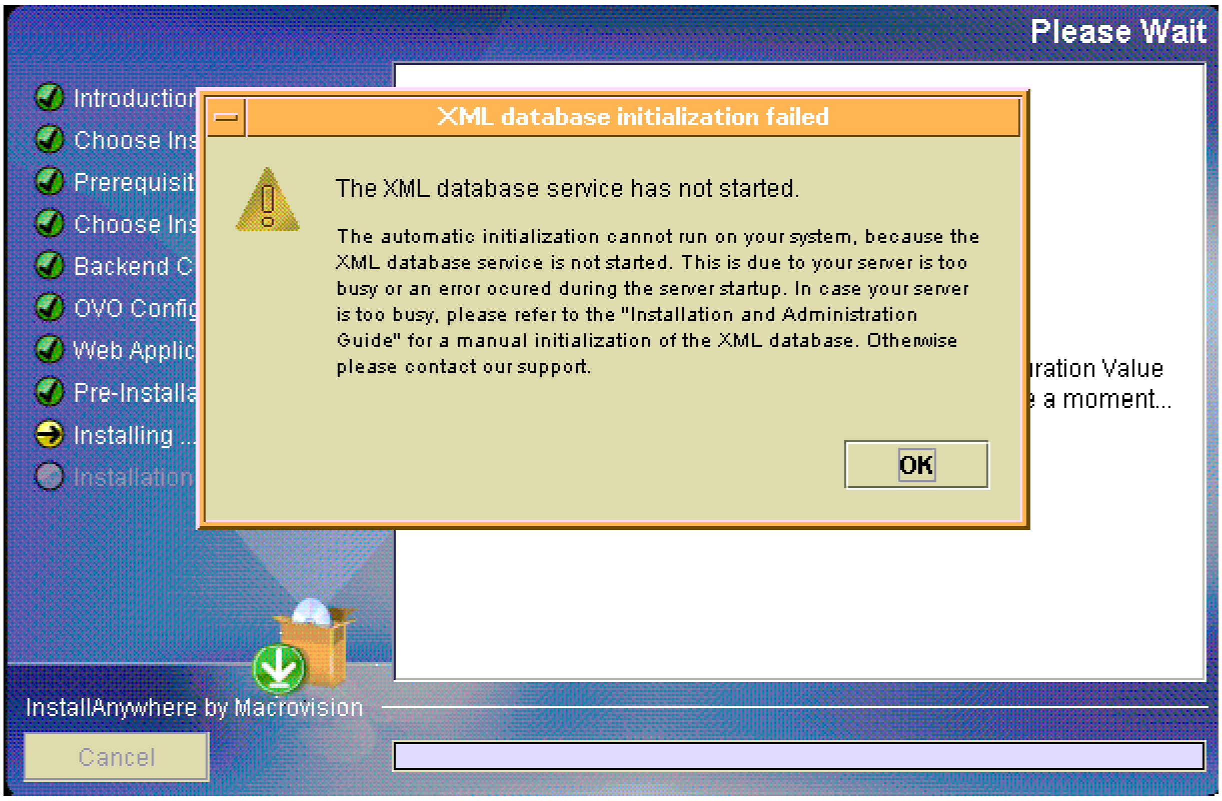
Please install the licenses manually like it is described in the "Install And Admin Guide".

This error message is displayed if the generation of the instant-on licenses has failed and `/dev/random` is detected. The pseudo-device `/dev/random` is used to create random keys for SSH.

Currently, there is no known solution for this problem. If this error occurs during the installation, contact the product support team to request an instant-on license for your system.

5.5.14.2 XMLDB error (WebApp only)

If you are installing CVP on slower systems or machines with a high workload during installation, the XML database startup might fail and display the error illustrated in the following figure:



The XML database service has not started.

The automatic initialization cannot run on your system, because the XML database service is not started. This is due to your server is too busy or an error ocured during the server startup. In case your server is too busy, please refer to the "Installation and Administration Guide" for a manual initialization of the XML database. Otherwise please contact our support.

If this error occurs during the installation, click OK and continue with the installation.

After the installation has completed, run the following command to re-initialize the XML database:

```
# <MIDAS_HOME>/midas.sh init
```

On Windows systems, use midas.bat instead, for example:

```
# <MIDAS_HOME>\midas.bat init
```

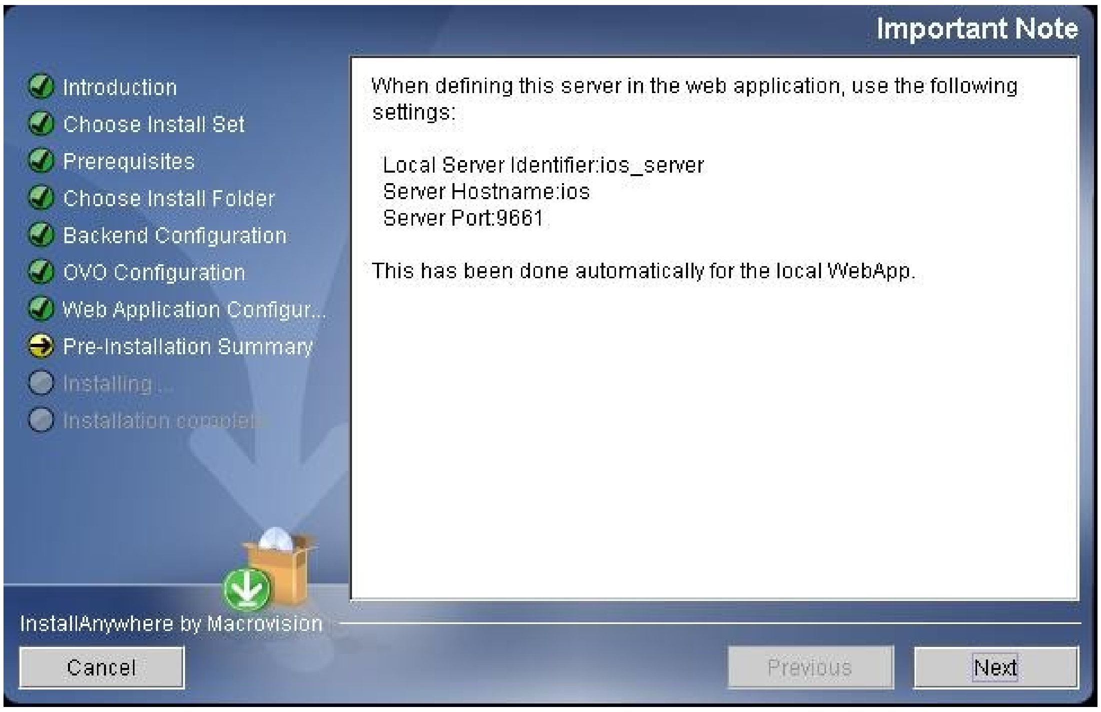
The `midas.sh init` command will load the default BackEnd and user definitions into the XML database.

CAUTION: Do not perform this initialization before the CVP WebApp server has started up completely. Otherwise the operation will fail because the `midas.sh` script cannot connect to the XML DB.

Also note, that this command applies only to CVP WebApp servers.

5.5.15 Post-Installation summary

After the installation has completed, the following two screens will appear containing a port configuration summary:

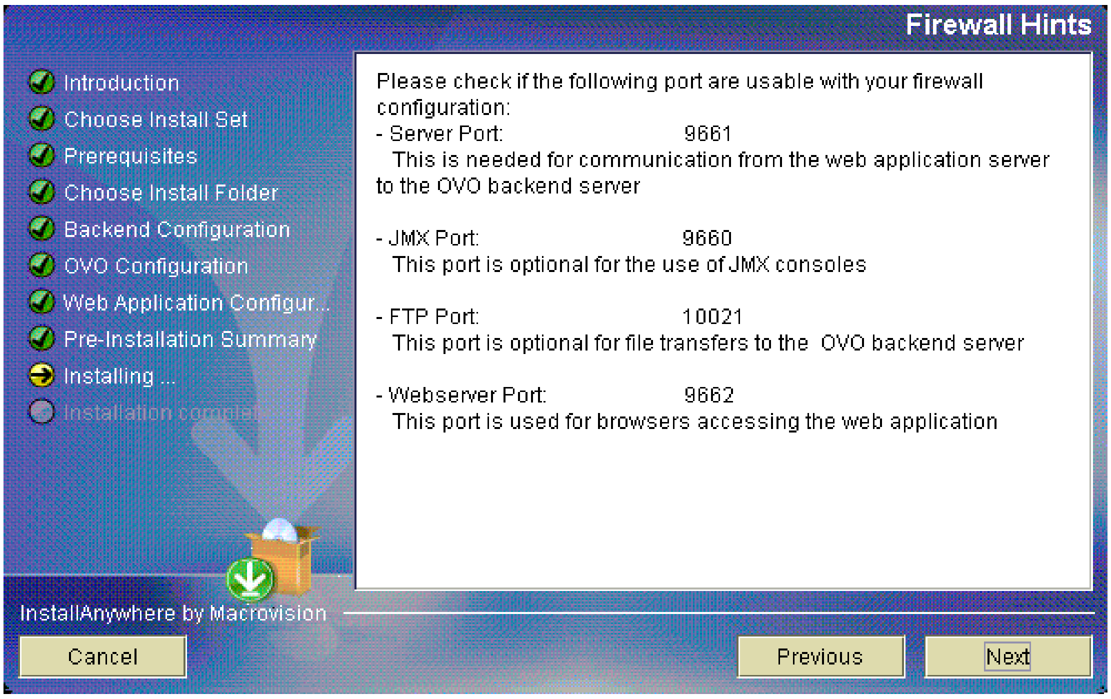


When defining this server in the web application, use the following settings:

Local Server Identifier:ios_server
Server Hostname:ios
Server Port:9661

This has been done automatically for the local WebApp.

NOTE: It is a good idea to print or save this information. If you are unable to print or save the configuration summary, you can view the information at any later time, too. The command `midas.sh backend` displays the same information.



```
Please check if the following port are usable with your firewall
configuration:
- Server Port:                    9661
  This is needed for communication from the web application server
to the OVO backend server

- JMX Port:                       9660
  This port is optional for the use of JMX consoles

- FTP Port:                       10021
  This port is optional for file transfers to the  OVO backend server

- Webserver Port:                 9662
  This port is used for browsers accessing the web application
```

5.6 Testing the installation

If the installation completes successfully, you can test it by logging on to the WebApp.

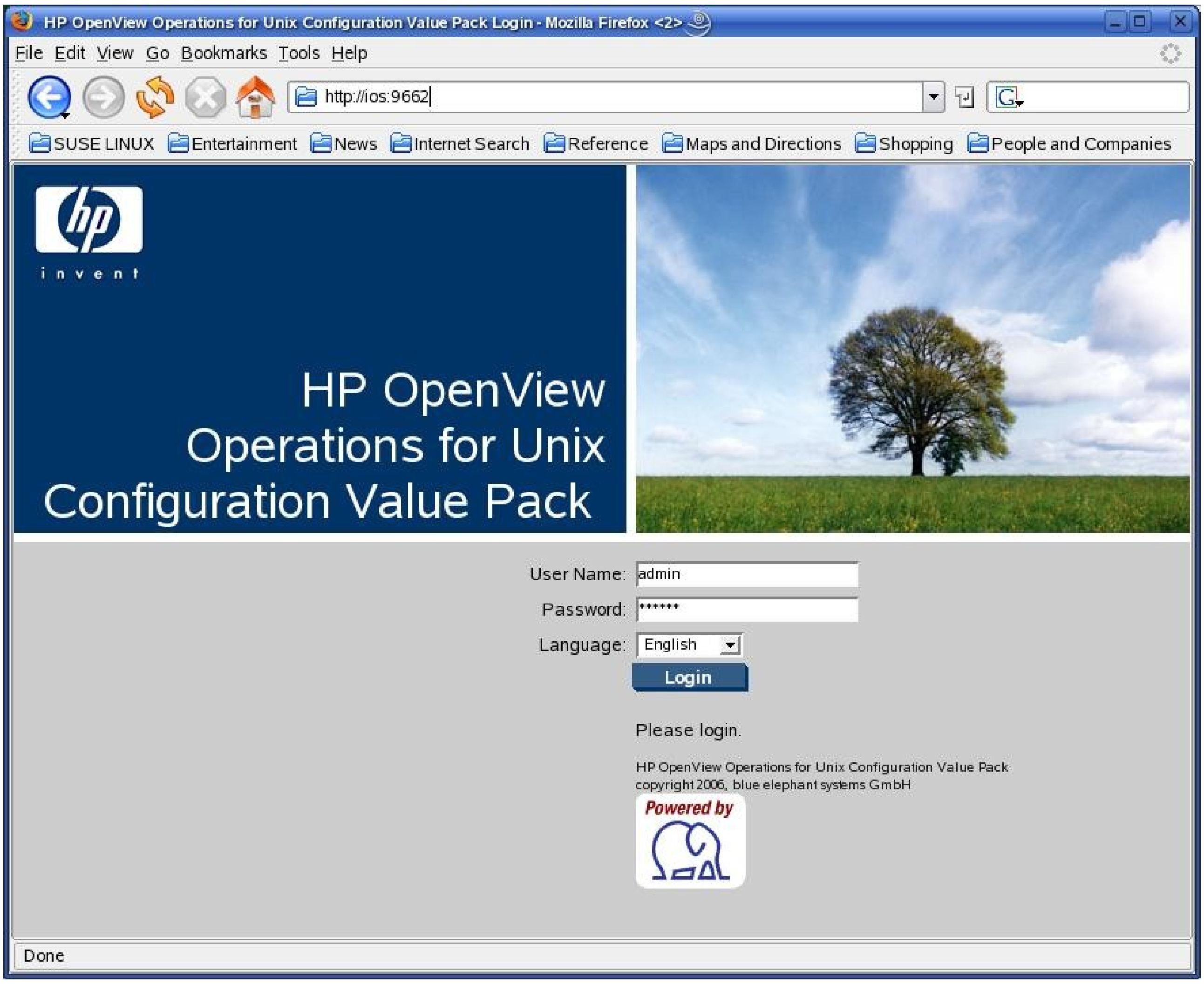
To log onto the WebApp, enter the following URL into a supported web browser:

```
http://<WebApp>:<port>/midas
```

Note that <WebApp> is the host name of the system where the CVP Web Application was installed and <port> is the port number that you selected during the pre-installation phase of the installation procedure. The default port for the WebApp is 9662. The trailing "/midas" can be omitted.

NOTE: At least one CVP WebApp must be available either stand-alone of as part of a full installation, where both the WebApp and the Backend are installed. A CVP BackEnd cannot be used as a stand-alone component.

The URL you use to log in to the WebApp displays the welcome page illustrated in the following figure:


HP OpenView Operations for Unix Configuration Value Pack Login - Mozilla Firefox <2>
File Edit View Go Bookmarks Tools Help
http://ios:9662
SUSE LINUX
Entertainment
News
Internet Search
Reference
Maps and Directions
Shopping
People and Companies
invent
HP OpenView Operations for Unix Configuration Value Pack
Login
Powered by
Done

User Name: admin
Password: ******
Language: English

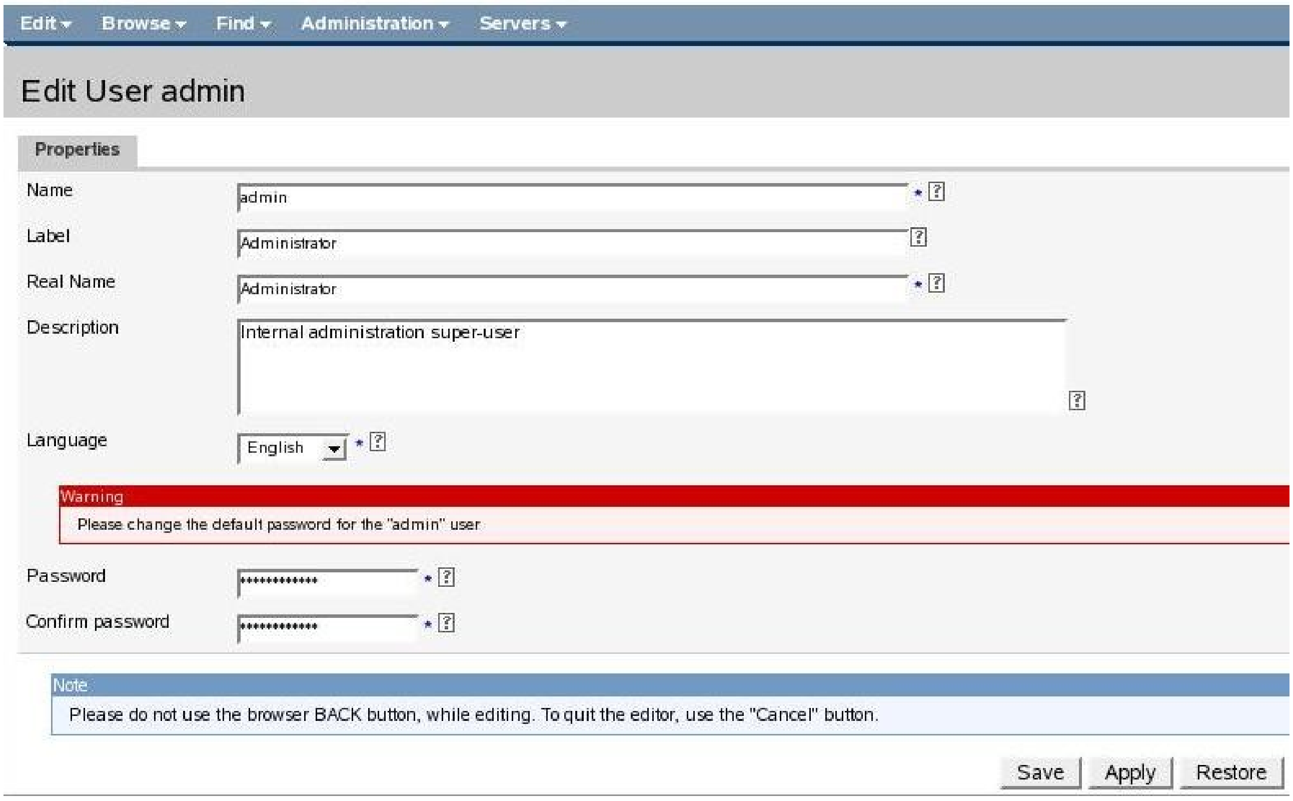
Please login.

HP OpenView Operations for Unix Configuration Value Pack
copyright 2006, blue elephant systems GmbH

To log in to WebApp, use the initial user name **admin** and the password **secret**. If this step succeeds, the WebApp is working correctly.

Currently the only supported language is English.

For security reasons you will have to change the default password of the user **admin** as shown in the following picture:



Edit User admin

Name	admin *
Label	Administrator
Real Name	Administrator *
Description	Internal administration super-user
Language	English *

Warning
Please change the default password for the "admin" user

Password	•••••••••••• *
Confirm password	•••••••••••• *

Note
Please do not use the browser BACK button, while editing. To quit the editor, use the "Cancel" button.

NOTE: Although you are logged into the WebApp, you are not yet connected to any OVO management server. For information about using CVP, see chapter 6 Using CVP.

If you want to test the login using HTTPS, see chapter 9.7.2.5 HTTPS between web browser and WebApp.

5.7 Installing in a HA cluster

CVP can be installed in a high-availability (HA) cluster. You can install either the BackEnd server or the WebApp as independent components or you can install both components together in a full installation. In all cases, you should bear in mind the information described in the following sections which describe tested and supported scenarios. Although other set-ups are possible, they have not yet been tested.

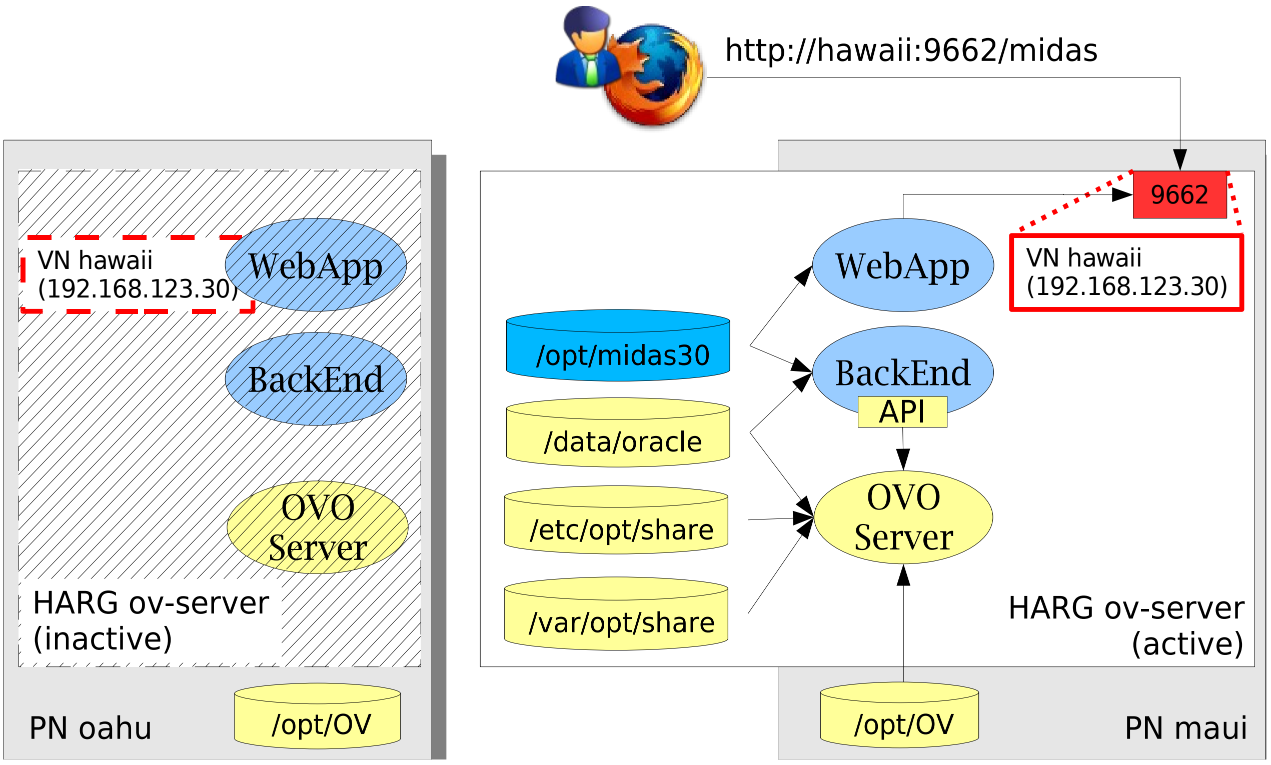
Whatever the actual structure of the HA set-up is, the following rules **must** be obeyed:

- The CVP BackEnd must run on the same physical node as the OVO Server. Remote communication between both is not possible.
- The virtual name and IP address of the HA resource group containing a CVP server must be resolvable with the name service (DNS, /etc/hosts, ...) at all locations where needed by other CVP components, for example: GUIs, other CVP WebApp or BackEnd servers.
- The virtual host name of the HA resource group must be specified during installation of CVP or, alternatively, when registering a BackEnd or logging in to CVP with a web browser.

For a list of supported HA software products, refer to chapter 5.1 Installation Prerequisites.

5.7.1 Full installation

In a Full Installation both the CVP WebApp and BackEnd reside on the same system. Since the BackEnd must run on the same node as the OVO Management Server, the resulting situation can be illustrated with the following picture:



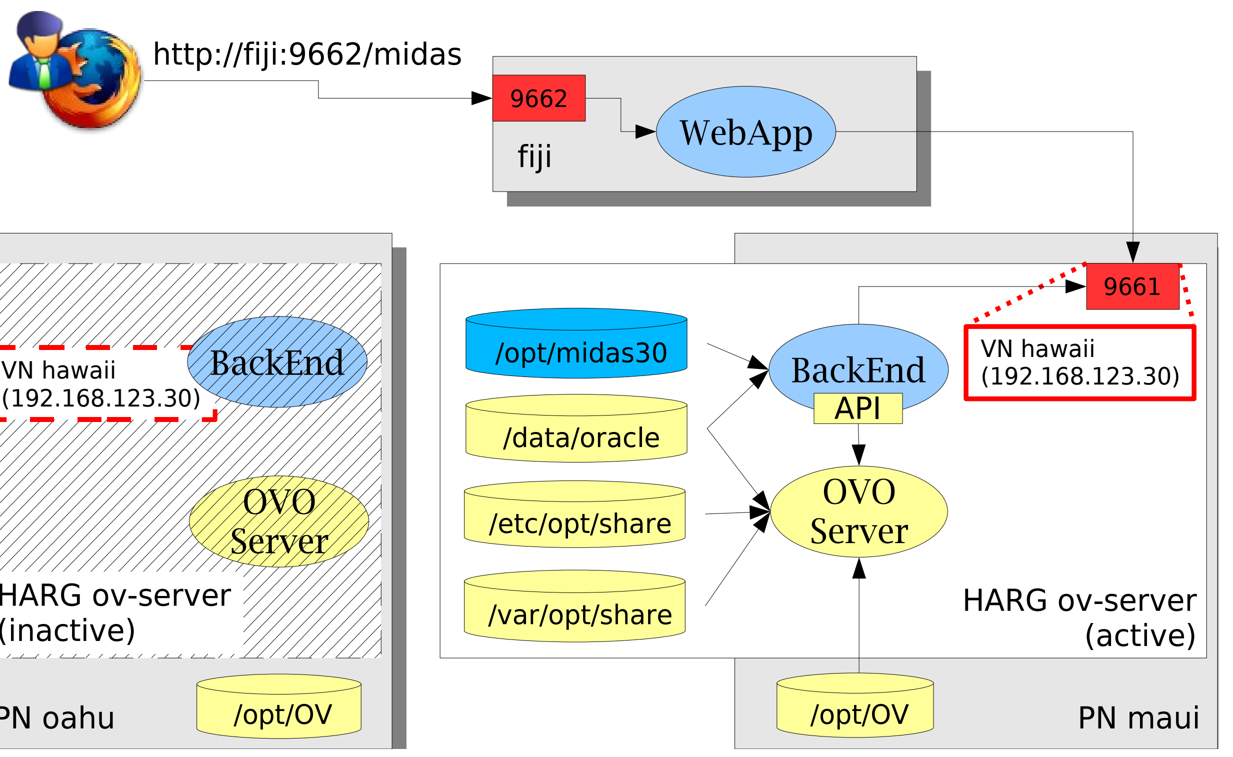
This example shows the following characteristics:

- CVP is installed into a dedicated shared file system is mounted to `/opt/midas30`. This file system is defined as resource belonging to the HA *resource group* (HARG, also referred to as HA *package*) **ov-server**.
- The CVP server (including BackEnd and WebApp) is configured as application resource into the same HARG ov-server as the OVO Management Server itself.
- The names of the *physical nodes* (PN) are: **oahu** and **maui**. Currently the HARG is active on maui. In case of a fail-over the entire HARG including the OVO Server and both the CVP WebApp and BackEnd moves to oahu.
- The *virtual host name* of the HARG is **hawaii.** The virtual IP address of hawaii is 192.168.123.60 (and must be resolvable by name services). In case of a fail-over this name and address moves along with the HARG from maui to oahu.

The WebApp front-end port (default 9662) is allocated by the running WebApp. The GUI session always connects to the CVP WebApp using the URL http://hawaii:9662/midas. The physical node, on which the WebApp is actually running, is completely transparent to the user.

5.7.2 Pure BackEnds

Another scenario is a clustered installation of the OVO Management Server with an associated CVP BackEnd server. The CVP WebApp exists standalone on system **fiji** as shown in the following picture:



In this scenario, the GUI session always connects to the CVP WebApp using the URL http://fiji:9662/midas. On the WebApp fiji, the BackEnd representing the clustered OVO Server must be registered with the virtual host name **hawaii** of the HARG ov_server.

All other characteristics are the same as in a Full Installation.

5.7.3 Pure WebApps

A standalone CVP WebApp can be configured as HARG too. However, this scenario is not considered as a common use case and has not been tested.

Since there are no dependencies on OVO or other components, the HA set-up should be fairly straightforward.

NOTE: It is currently not supported to configure a BackEnd as one HARG and a WebApp as another HARG within the same HA cluster.

5.7.4 Installation location

If you are installing CVP in a high-availability cluster, the complete software (including binaries, configuration and runtime data) must be installed into a file system located on a shared disk. This file system must be available on all physical cluster nodes, where the CVP server package is active.

This implies, that you need to install CVP only once per HA cluster.

5.7.5 Defining the HA package

The CVP server must be configured as part of an HA package (or resource group, HARG). This HARG should be the same HARG as the one used by the OVO Server (it is possible to configure CVP as an individual HARG with a dependency on the OVO HARG – however, this is not recommended).

NOTE: The definition of the HA package must be performed manually; it is not part of the CVP installation.

You can configure a package switch as a result of a CVP failure depending on how critical you think CVP is in your integrated environment. If CVP is not critical to your environment, remember to configure the OVO HA package not to perform a fail-over switch, if CVP fails.

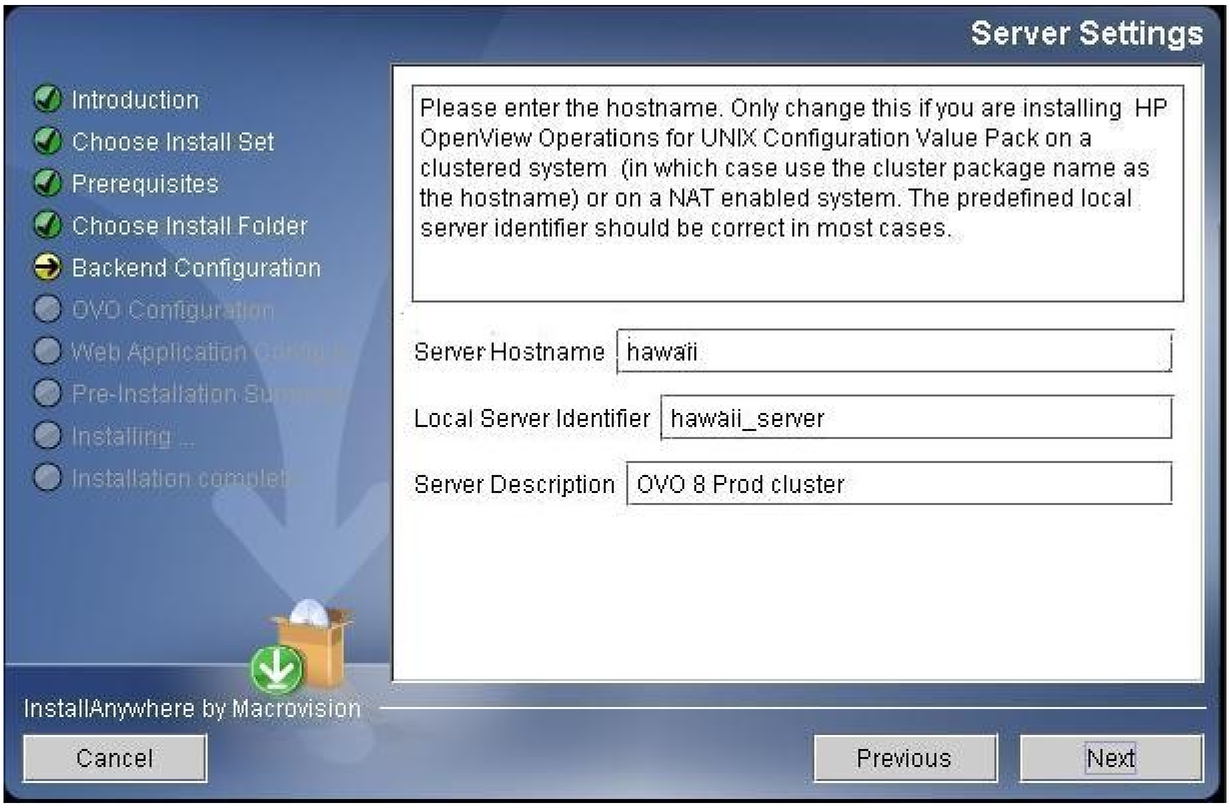
The following example of a Veritas cluster configuration defines the scenario presented above:

```
[...]
system oahu ()
system maui ()

group ov-server (
     SystemList = { oahu = 1, maui = 2 }
     )

Application app-midas (
     AutoStart = 0
     Critical = 0
     User = root
     StartProgram = "/opt/midas30/midas.sh start"
     StopProgram = "/opt/midas30/midas.sh stop"
     PidFiles = "/var/tmp/midas.pid"
     )
Application app-ovo (
     StartProgram = "/opt/OV/lbin/ovharg -start ov-server"
     [...]
     )
[...]
IPMultiNICB ovo-ip (
     BaseResName = ovo-nic
     Address = "192.168.123.60"
     NetMask = "255.255.255.0"
)
Mount mount-midas (
     AutoStart = 0
     Critical = 0
     MountPoint = "/opt/midas30"
     BlockDevice = "/dev/vx/dsk/dgOVO/volume-midas"
     FSType = vxfs
     FsckOpt = "-y"
     )
[...]
app-midas requires app-ovo
app-midas requires mount-midas
app-ovo requires ovo-ip
[...]
mount-midas requires dgOVO
```

```
[...]
// resource dependency tree
//
//   group ov-server
//   {
//     Application app-midas
//     {
//       Application app-ovo
//       {
//         [...]
//       }
//       Mount mount-midas
//       {
//         DiskGroup dgOVO
//       }
//     }
//     [...]
//   }
```

5.7.6 Server ID and host name

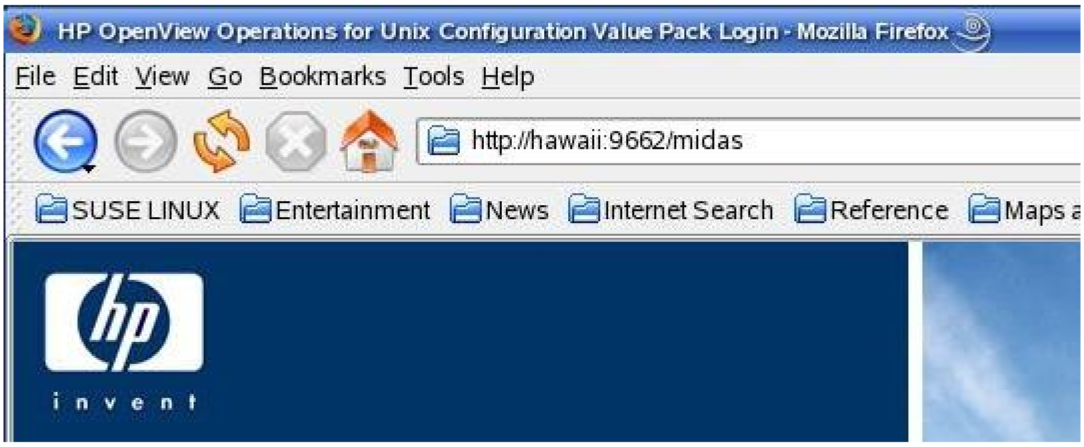
During CVP installation, it is crucial to specify the virtual host name of the HA package; this is the host where the CVP runs. If the name of the virtual host is *hawaii*, enter *hawaii* in the ***Server Hostname*** field, as illustrated in the following figure:



Please enter the hostname. Only change this if you are installing HP OpenView Operations for UNIX Configuration Value Pack on a clustered system (in which case use the cluster package name as the hostname) or on a NAT enabled system. The predefined local server identifier should be correct in most cases.

Server Hostname: hawaii
Local Server Identifier: hawaii_server
Server Description: OVO 8 Prod cluster

If you want to log in to a WebApp running with a HARG, you also enter the name of the virtual host in the URL you use to log in, as illustrated in the following figure:


HP OpenView Operations for Unix Configuration Value Pack Login - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://hawaii:9662/midas
SUSE LINUX
Entertainment
News
Internet Search
Reference
Maps

If the *hawaii* BackEnd has to be registered with a WebApp, use the virtual host name of the HA package and the server ID, as specified in the installer.

IMPORTANT: Whenever you have to refer to a CVP server running as part of an HA package, specify the host name, IP address, and CVP server ID of the virtual host running the HA package.

5.7.7 Start-up/shutdown

The start-up and shut-down mechanism you choose for CVP depends on whether the `ovstart` command is available or not. If the `ovstart` and `ovstop` commands are available, you should consider using it, as it is the easiest solution for automatic start-up and shut-down. If the `ovstart` and `ovstop` commands are not available, you have to integrate the CVP main control script `midas.sh` into the cluster fail-over procedure if you want CVP to start and stop automatically whenever the package switches host due to a fail-over.

5.7.8 Licensing

You need only one set of licenses (including the relevant CVP components) for a CVP server running in a HA cluster. The license must be issued to the virtual IP address and will be installed on the shared disk. The license will switch between physical nodes along with the software itself, and no other special precautions are necessary.

5.7.9 External resources

When installing CVP in a HA cluster, make sure that all resources, which are not part of the HA package, are present on all involved physical nodes. This applies particularly to the following points:

- The operating-system user, `midas`, must be present on all cluster nodes. For more information, see 5.15.1 Installation Prerequisites
- External software such as CVS, and SSH must be present on all cluster nodes.

5.8 Installation Troubleshooting

5.8.1 Display problems

The CVP installer attempts to open a GUI after unpacking itself. On UNIX system, this requires the correct setting of the DISPLAY environment variable and permission to access the X server on the workstation where the installation is initiated.

If not set correctly, the installer will print an error similar to the one shown in the following example (particularly watch out for the text talking about the *LocalGraphicsEnvironment*):

```
# /tmp/install.bin
Preparing to install...
Extracting the JRE from the installer archive...
Unpacking the JRE...
Extracting the installation resources from the installer archive...
Configuring the installer for this system's environment...
Launching installer...
Invocation of this Java Application has caused an
InvocationTargetException. This application will now exit. (LAX)
Stack Trace:
java.lang.NoClassDefFoundError
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Class.java:141)
        at
java.awt.GraphicsEnvironment.getLocalGraphicsEnvironment(GraphicsEnviro
nment.java:62)
        at java.awt.Window.init(Window.java:231)
        at java.awt.Window.<init>(Window.java:275)
        [...]
```

To point the installer to the correct target, prefix the installer command by the correct DISPLAY variable as shown in the following example (the example assumes that CVP is to be installed on host *src* whereas the user is logged on at the system *tgt*, i.e. the display must be redirected from *src* to *tgt*):

```
# [root@src]> DISPLAY=tgt:0 /tmp/install.bin
```

Allow access on the target *tgt* workstation by executing the following command:

```
# [user@tgt]> xhost +
```

To test general X connectivity, execute any other X application. For example, open a clock using the command `xclock`:

```
# [root@src]> DISPLAY=tgt:0 xclock
```

If this works correctly, also the CVP installer must be able to open the display properly.

NOTE: On certain RedHat Linux versions a known problem makes the installer fail to open the display even if everything is set correctly. See chapter 5.3 OS and application patches for details.

5.8.2 Installation log files

The CVP installation procedure creates two log files in the installation root directory `<MIDAS_HOME>` as specified during installation:

- `CVP_InstallLog.log` This file contains information about the individual installation steps and whether they where successful
- `midas_install.log` This file contains information about the values of variables used during installation, the JRE etc. In addition, any output created by the various scripts called during the installation process, for example, to determine the Oracle installation directory or to create configuration elements, will be logged to this file.

While the installation is running, these two log files are created and written in the temporary installer directory. At the very end of the installation procedure the log files are copied to `<MIDAS_HOME>`.

This temporary installation directory is by default:

UNIX:	`/tmp/install.dir.<some number>`
Windows:	`C:\Temp\<some number>`

If the IATEMPDIR environment variable has been set to an alternative location, also the installation log files will reside in this directory.

You can trace the installation progress on a UNIX machine using the `tail` command, for example:

```
# tail -f /tmp/install.dir.124211/midas_install.log
```

It is also possible to redirect the CVP Installer debug output to the console. To do this on a UNIX operating systems, set the environment variable `LAX_DEBUG` to true, before starting the installer, for example:

```
# LAX_DEBUG=true <Media>/install.bin
```

To trace installation progress on Windows operating systems, hold down the <CTRL> key immediately after launching the installer and hold it down until a console window appears.

NOTE: If you redirect the output to your screen by setting the LAX_DEBUG variable, the installer will **not** create the `midas_install.log` log file.

If there is a persistent problem with the CVP installation which you cannot resolve yourself, send the installation log files to the product support together with the support archive you may be able to create with the following command:

```
# <MIDAS_HOME>/midas.sh support
```

5.8.3 The midas.properties file

If a previous version of the product has been de-installed in the past and is to be reinstalled again, the Installer sometimes fails and exits due to the existence of the `midas.properties` files which you can find in the following location:

Windows:	`C:\Windows\midas.properties`
UNIX:	`/var/opt/midas/midas.properties`

Remove the `midas.properties` file and start the installation again.

5.9 Upgrading MIDAS 2.x

A direct upgrade from MIDAS 2.x is not possible due to major internal changes. To help migrate existing MIDAS environments, there are several custom migration tools available. Please contact support@blue-elephant-systems.com for more information.

CVP coexists with MIDAS 2.x, even if installed on the same system. This scenario might be helpful during a migration.

5.10 De-installing CVP

Before you start the de-installation, be sure to stop all running CVP servers. To stop the CVP servers on UNIX systems, us the following command:

```
# <MIDAS_HOME>/midas.sh stop
```

On Windows systems, stop the CVP servers using the Service Control Panel or the following command:

```
> midas.bat stop
```

Make sure that no CVP-related wrapper or Java processes are running before you start the removal process. If any CVP-related `wrapper` or `java` processes are running, stop or kill them as necessary.

The program you use to remove CVP is located in the CVP home directory. Start the de-installer as follows:

```
# <MIDAS_HOME>/Uninstall_MIDAS_3.0/uninstall.bin
```

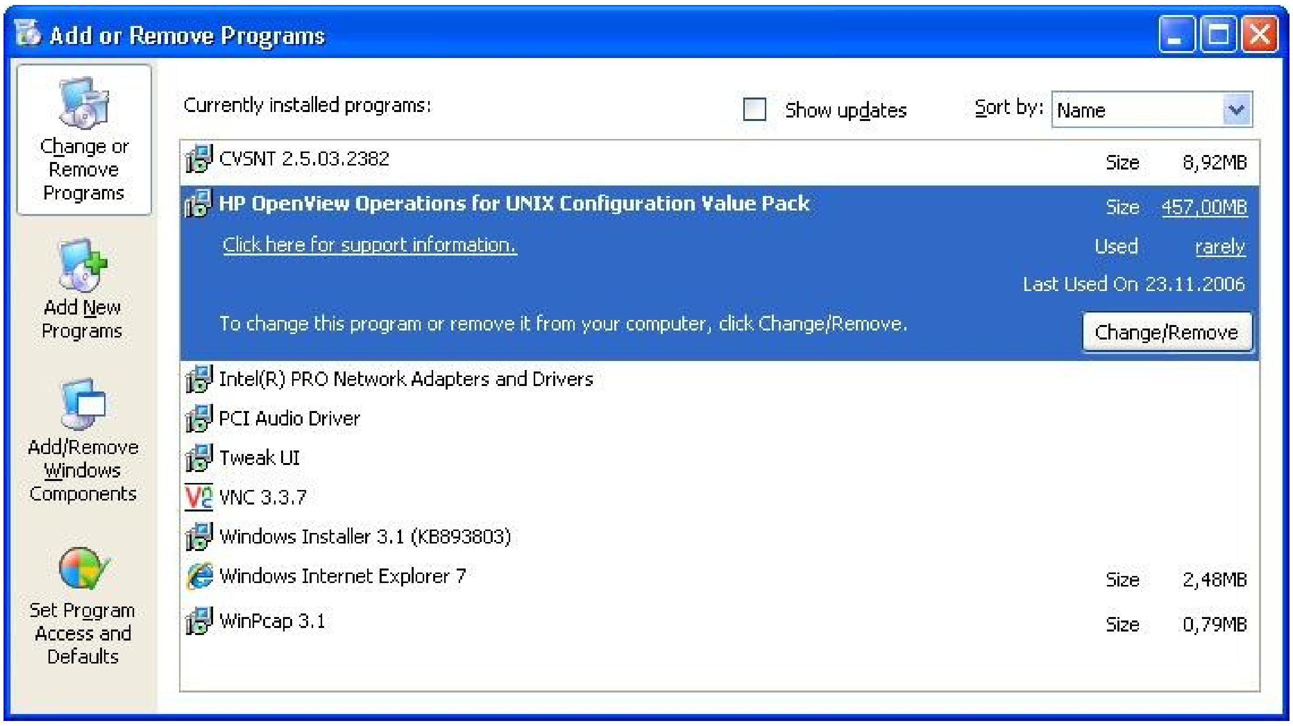
On Suse 10.1, Suse Enterprise Server/Desktop 10 and FedoraCore5 system, the de-installation may fail with the following error:

```
[...]
Configuring the installer for this system's environment...
awk: error while loading shared libraries: libdl.so.2: cannot open shared
object file: No such file or directory
[...]
```

To avoid this, perform the following steps before starting the de-installation:

```
# cd <MIDAS_HOME>/Uninstall_MIDAS
# cat uninstall.bin |
  sed 's/export LD_ASSUME_KERNEL/#xport LD_ASSUME_KERNEL/g' >
  uninstall.bin2; mv uninstall.bin2 uninstall.bin
# chmod 755 uninstall.bin
# ./uninstall.bin
```

On Windows systems, you can as well remove CVP via the Add/Remove Software function of Windows:

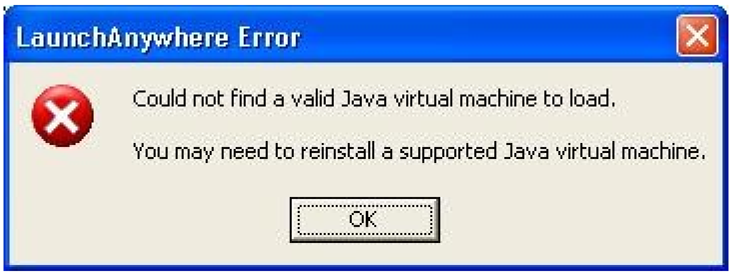

Add or Remove Programs
Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults
Show updates
Change/Remove

Currently installed programs: Sort by: Name

CVSNT 2.5.03.2382	Size	8,92MB
HP OpenView Operations for UNIX Configuration Value Pack	Size	457,00MB
Click here for support information.	Used	rarely
	Last Used On	23.11.2006
To change this program or remove it from your computer, click Change/Remove.		
Intel(R) PRO Network Adapters and Drivers		
PCI Audio Driver		
Tweak UI		
VNC 3.3.7		
Windows Installer 3.1 (KB893803)		
Windows Internet Explorer 7	Size	2,48MB
WinPcap 3.1	Size	0,79MB

This will launch the same graphical de-installer.

If de-installing CVP on a Windows systems a known problem may cause the following error:



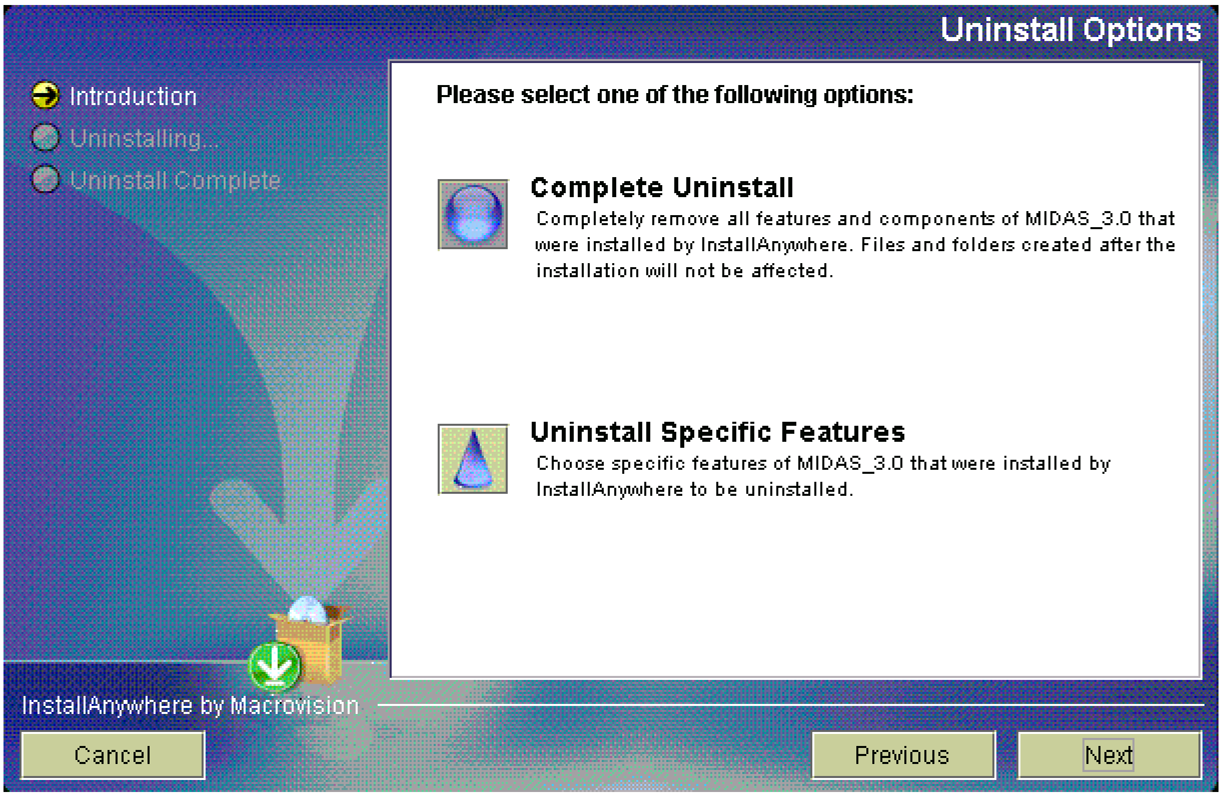
Could not find a valid Java virtual machine to load.

You may need to reinstall a supported Java virtual machine.

To work around this, make sure that a `java` executable can be found in the PATH of the environment where the de-installation is performed (in the current shell if using the de-installer from the command line, or as system variable if starting the de-installation from the Windows control panel).

CAUTION: Do not just remove the product files from the file system, for example: using the UNIX `rm` command. The CVP installer tool, InstallAnywhere, maintains a software inventory which will not be updated and subsequent re-installations may fail because it appears as if the software is already installed.

After launching the de-installation a graphical interface starts, which asks you to choose the de-installation method; you can choose to remove the complete software or selected components.

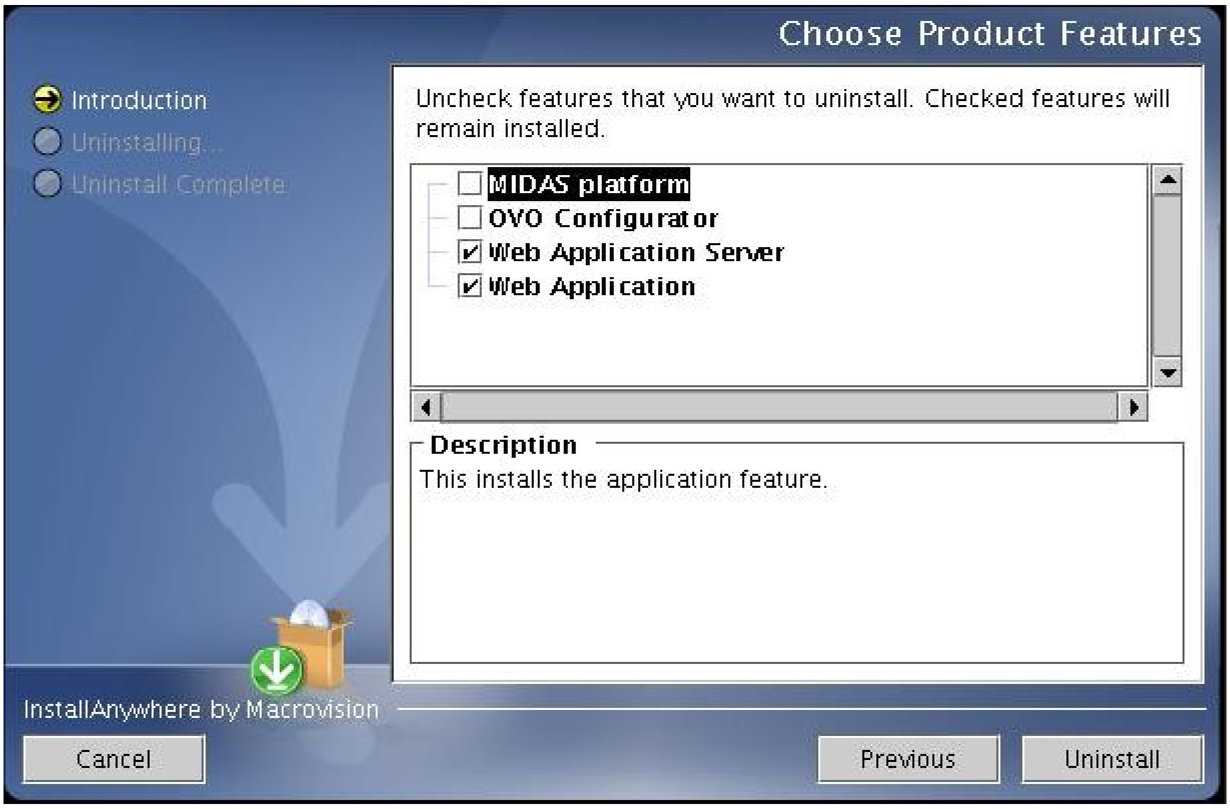


Please select one of the following options:

Complete Uninstall
Completely remove all features and components of MIDAS_3.0 that were installed by InstallAnywhere. Files and folders created after the installation will not be affected.

Uninstall Specific Features
Choose specific features of MIDAS_3.0 that were installed by InstallAnywhere to be uninstalled.

If you choose to remove specific components, the Choose Product Features screen displays a list of the components you can remove. Uncheck the components you want to remove as illustrated in the following screen; check the components you want to remain installed.


Choose Product Features

Uncheck features that you want to uninstall. Checked features will remain installed.

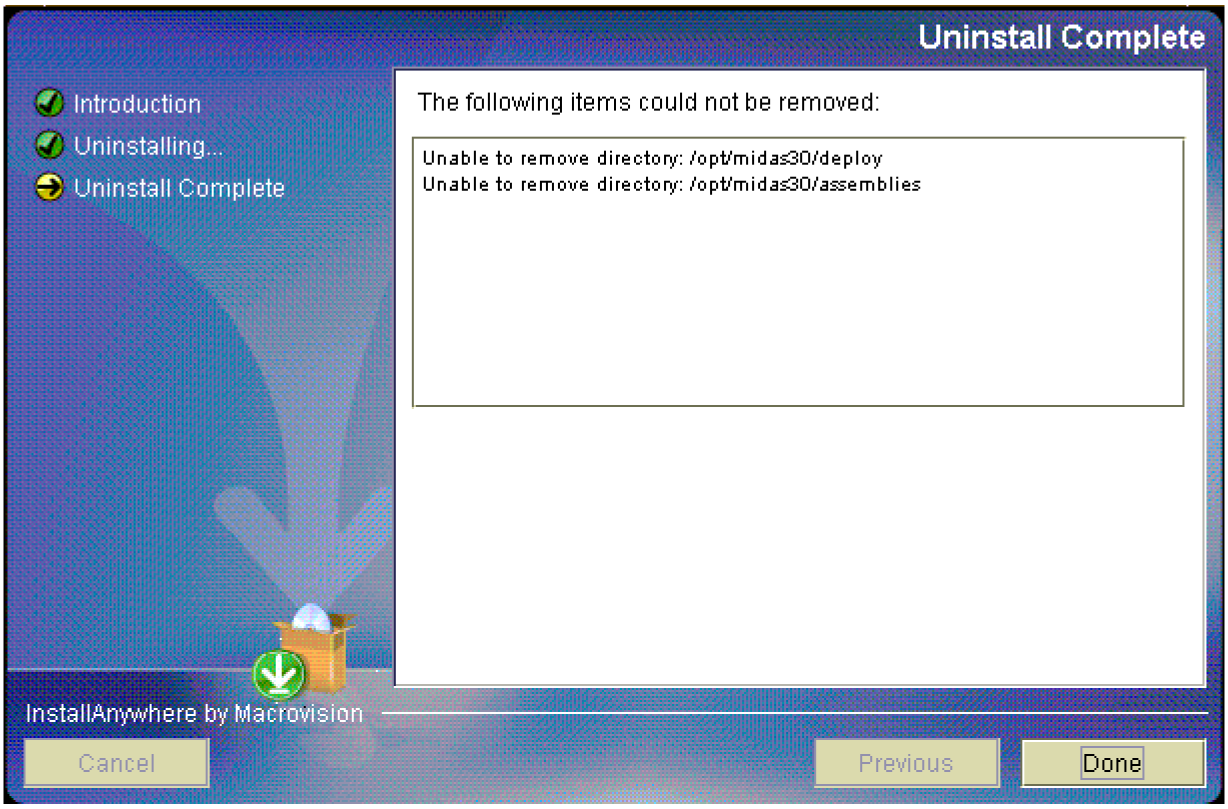
- [ ] MIDAS platform
- [ ] OVO Configurator
- [✓] Web Application Server
- [✓] Web Application

Description

This installs the application feature.

The set of displayed components depends on the installed OpenView software, operating system platform and the previously installed CVP components.

Click the ***Uninstall*** button to start the de-installation procedure. After completion of the de-installation a summary screen displays a list of the directories which could not be removed, for example, because they are still in use by other components or contain customer-specific files (documents or configuration files), the CVS repository, or static documentation from the CVP Web Application.



The following items could not be removed:

```
Unable to remove directory: /opt/midas30/deploy
Unable to remove directory: /opt/midas30/assemblies
```

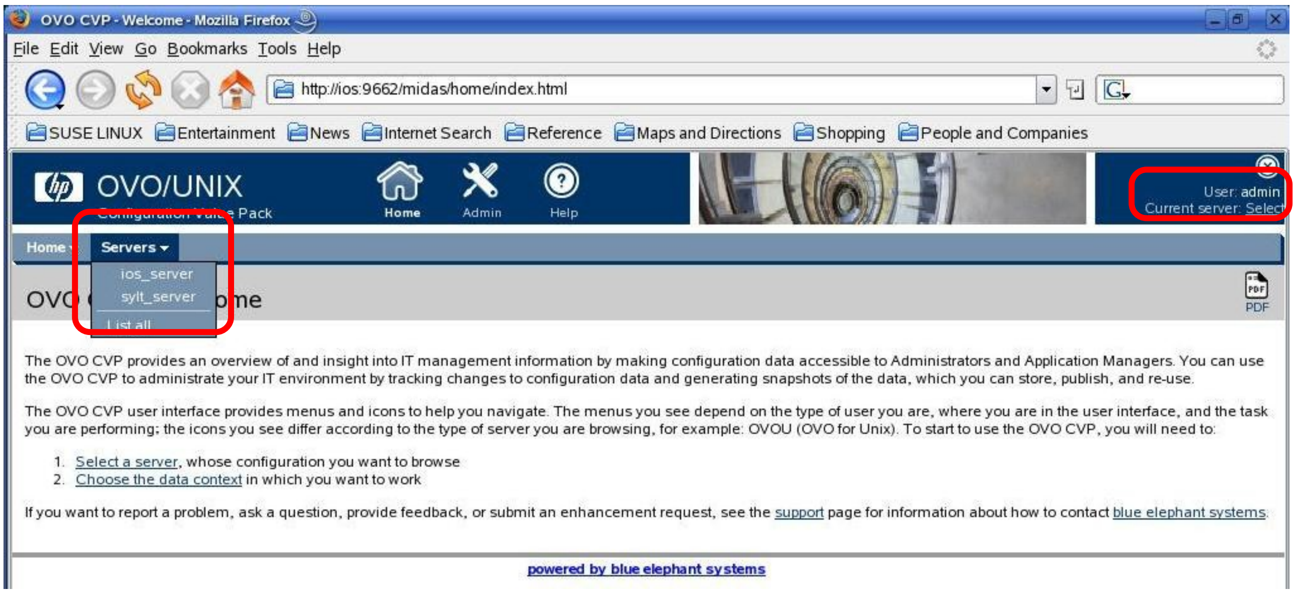
To completely remove the entire CVP installation including all custom data files, check the installation directory <MIDAS_HOME> and manually remove the remaining files.

6 Using CVP

After installing the CVP software and logging in the for the first time, you need to select the BackEnd server whose data configuration you want to browse and, in addition, set the data context for the selected back-end server, for example: *OVO /U* or *Admin*. For subsequent logins, you can avoid having to repeat the procedure of selecting a BackEnd and choosing the data context by bookmarking the login page in your browser as described in the following section.

6.1 Selecting a BackEnd

CVP allows you to browse the data in multiple BackEnds concurrently. However, even if there is only one back end configured in your installation, you still need to select it after logging on; you can select the back end either manually or by setting a browser bookmark.


OVO CVP - Welcome - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://ios:9662/midas/home/index.html
SUSE LINUX Entertainment News Internet Search Reference Maps and Directions Shopping People and Companies
OVO/UNIX
Home Admin Help
User: admin
Current server: Select
Home Servers
ios_server
sylt_server
List all
PDF

The OVO CVP provides an overview of and insight into IT management information by making configuration data accessible to Administrators and Application Managers. You can use the OVO CVP to administrate your IT environment by tracking changes to configuration data and generating snapshots of the data, which you can store, publish, and re-use.

The OVO CVP user interface provides menus and icons to help you navigate. The menus you see depend on the type of user you are, where you are in the user interface, and the task you are performing; the icons you see differ according to the type of server you are browsing, for example: OVOU (OVO for Unix). To start to use the OVO CVP, you will need to:

1. Select a server, whose configuration you want to browse
2. Choose the data context in which you want to work

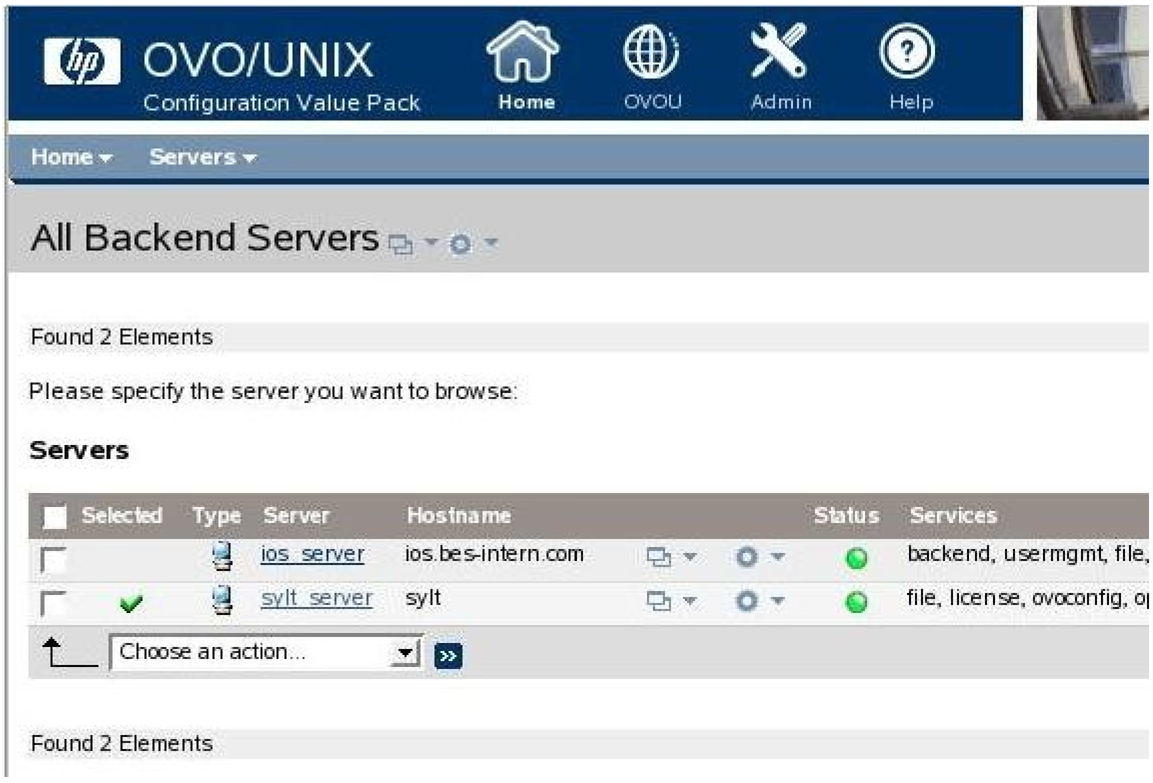
If you want to report a problem, ask a question, provide feedback, or submit an enhancement request, see the support page for information about how to contact blue elephant systems.

powered by blue elephant systems

To select a back-end server manually, use the Servers drop-down menu or the *Select* link on the right-hand side of the web-page banner.

Note that if there are separate installations of CVP WebApp and BackEnd systems, initially the Servers menu will display the the name of the system hosting the local WebApp only. This is because you log on to the WebApp, and the WebApp does not, at first, know about the existence of any other CVP BackEnd servers. If you know there are multiple servers available but cannot see them in the drop-down servers menu, click *List all* in the *Servers* drop-down menu or the *Select* link on the right-hand side of the web-page banner. If the servers are available in the server list, you can also select the desired BackEnd immediately by clicking the name that represents the BackEnd server whose data you want to browse.

The following figure shows that the server `sylt` is selected. Since `sylt` hosts an OVO management server, the OVOU icon appears in the web-page banner.



All Backend Servers

Found 2 Elements

Please specify the server you want to browse:

Servers

	Selected	Type	Server	Hostname			Status	Services
☐			ios_server	ios.bes-intern.com			●	backend, usermgmt, file,
☐	✓		sylt_server	sylt			●	file, license, ovoconfig, o

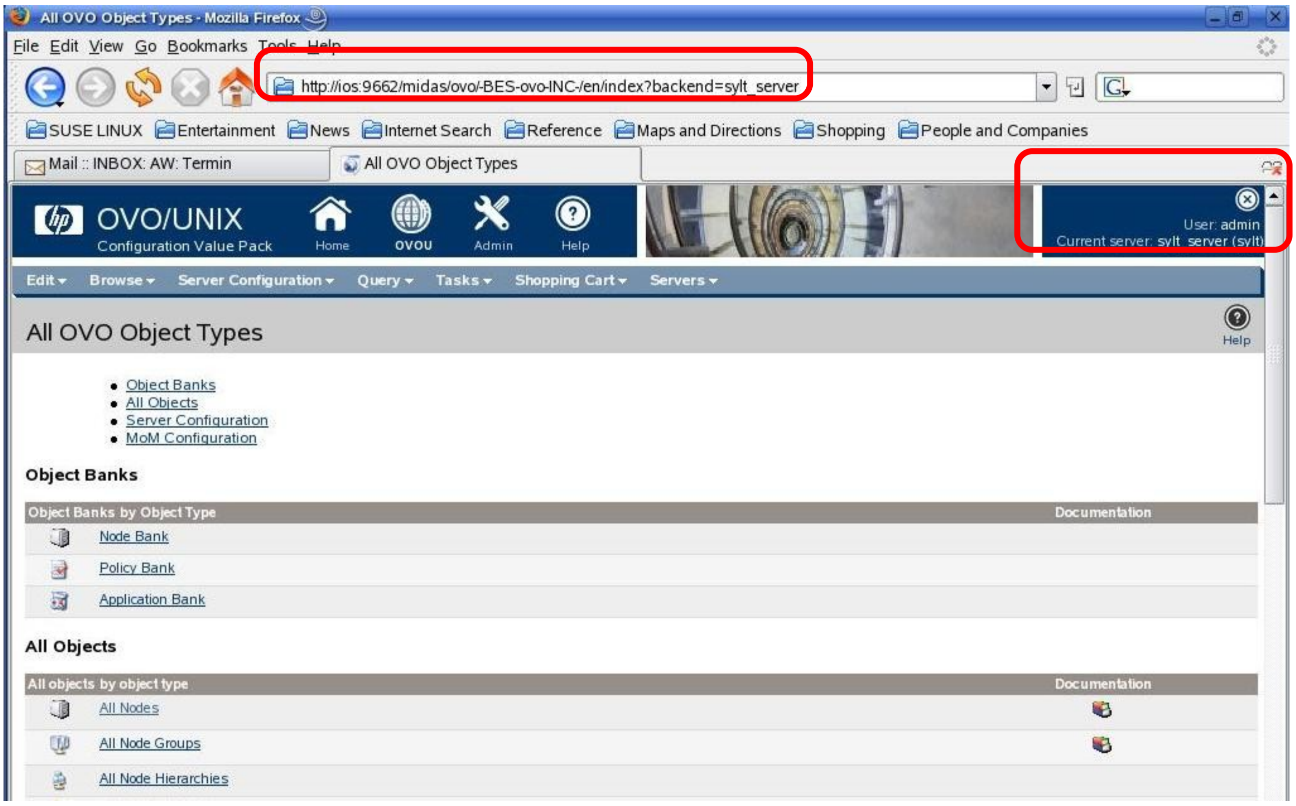
Choose an action...

Found 2 Elements

6.2 Selecting the OVO context

Click the ***OVO/U*** icon in the web-page banner to set the data context to OVO for UNIX; after setting the data context to OVO for UNIX, you can browse your OVO for UNIX management configuration. Depending on the setup of the system hosting the OVO management server and CVP, you might see other icons indicating the presence of other integrated software products.

The box in the top right-hand corner of the web page indicates the identity of the user currently logged in to CVP and, in addition, the name of the management server whose configuration data the named user is browsing. In the example illustrated in the following figure, the user `admin` is logged in to CVP on the WebApp `ios` and is browsing OVO configuration data on the OVO management server `sylt`. Note that the information about the identity of the BackEnd server is also available in the URL, for example: `.../index?backend=sylt_server`.


All OVO Object Types - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://ios:9662/midas/ovo/-BES-ovo-INC-/en/index?backend=sylt_server
SUSE LINUX Entertainment News Internet Search Reference Maps and Directions Shopping People and Companies
Mail :: INBOX: AW: Termin
All OVO Object Types
OVO/UNIX
Configuration Value Pack
Home
OVOU
Admin
Help
User: admin
Current server: sylt_server (sylt)
Edit Browse Server Configuration Query Tasks Shopping Cart Servers

All OVO Object Types

- Object Banks
- All Objects
- Server Configuration
- MoM Configuration

Object Banks

Object Banks by Object Type	Documentation
Node Bank	
Policy Bank	
Application Bank	

All Objects

All objects by object type	Documentation
All Nodes	
All Node Groups	
All Node Hierarchies	

If you are not sure how to find information that you need to browse or do not know what information is required to complete a task in CVP, use the on-line help system. Each CVP web page is linked to the CVP on-line help and provides quick access to context-sensitive help for the page displayed. If the information in the on-line help page displayed does not answer your question exactly or completely, use the list of related topics at the bottom of each on-line help page to browse information about similar topics.

> TIP: To avoid having to select a BackEnd server and choose the OVO/U data context each time you log in to CVP, set a bookmark to link directly to the page displayed after you log in.
>
> You can also create bookmarks for later reference at any other point while navigating through the OVO objects. This provides quick access to frequently used objects or views.

To log in to CVP and open a page displaying a selected BackEnd server and a particular data context (for example, OVO), you need to use a URL similar to the one displayed in the following example:

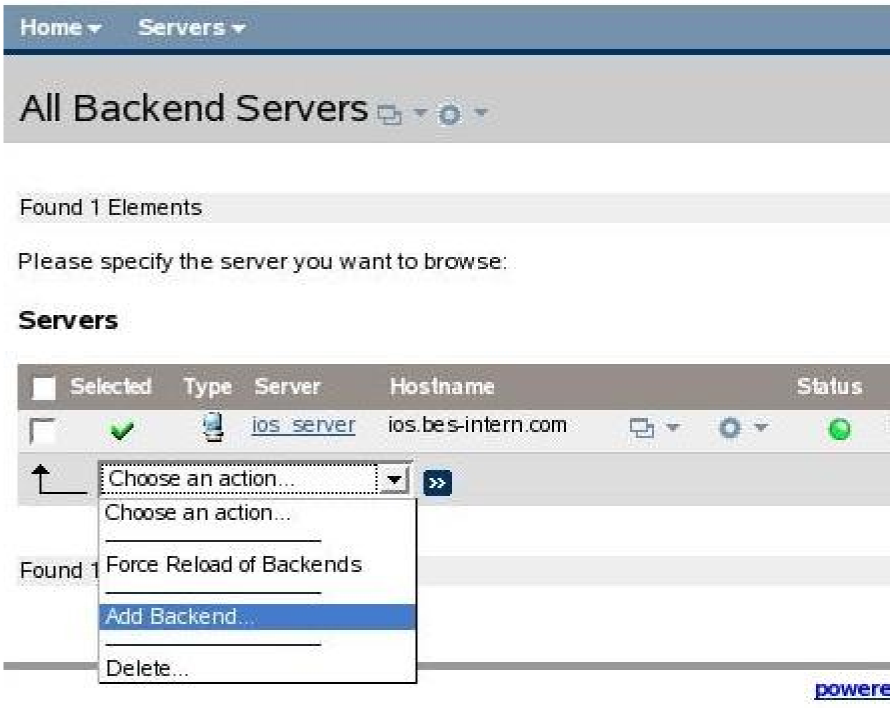
```
http://<WebApp>:<port>/midas/ovo/-BES-ovo-INC-/en/index?backend=<BackEnd ID>
```

For example, the following URL logs a user directly in to the BackEnd server `sylt` and sets the data context to OVOU. Note that, in the following example, the WebApp is running on a separate server called `ios`, so both servers (ios and sylt) would have to be available for the log-on to succeed. You can log in to any CVP page you choose; if you want to make the OVO NodeBank your start page, log in to CVP, browse to the Node Bank page, and set a bookmark for the page.

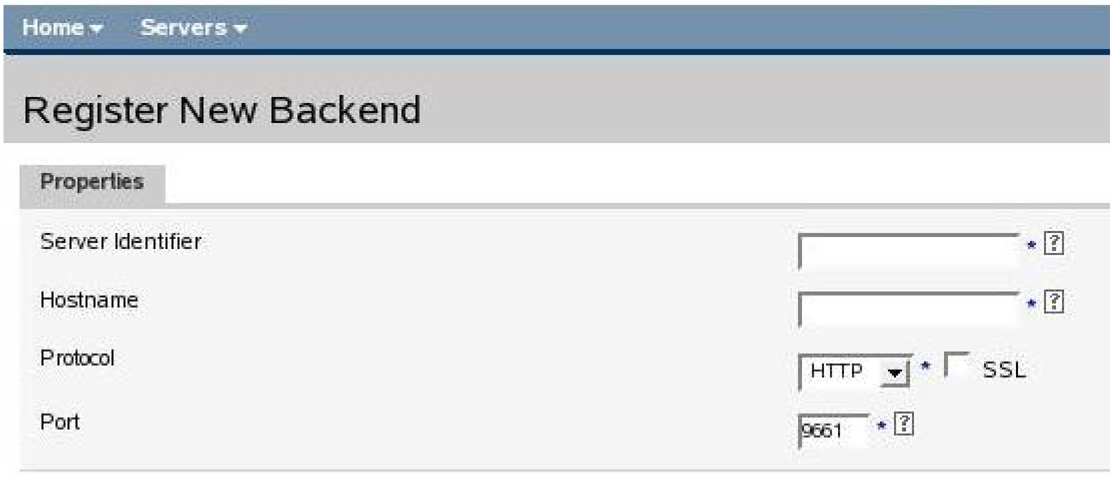
http://ios:9662/midas/ovo/-BES-ovo-INC-/en/index?backend=sylt_server

6.3 Registering a BackEnd

To register a BackEnd manually so that it appears in the list of BackEnd servers, use the *Add BackEnd* option in the Choose-an-action menu available in the *All BackEnd Servers* list, as illustrated in the following example:



CVP displays the following page to allow you to define the required parameters for the registration of the new BackEnd server:



If you do not know or cannot remember the settings required for the manual registration of the new BackEnd, log on to the system hosting the CVP BackEnd server (for example: via `telnet` or `ssh`) and run the following command:

```
# <MIDAS_HOME>/midas.sh backend
```

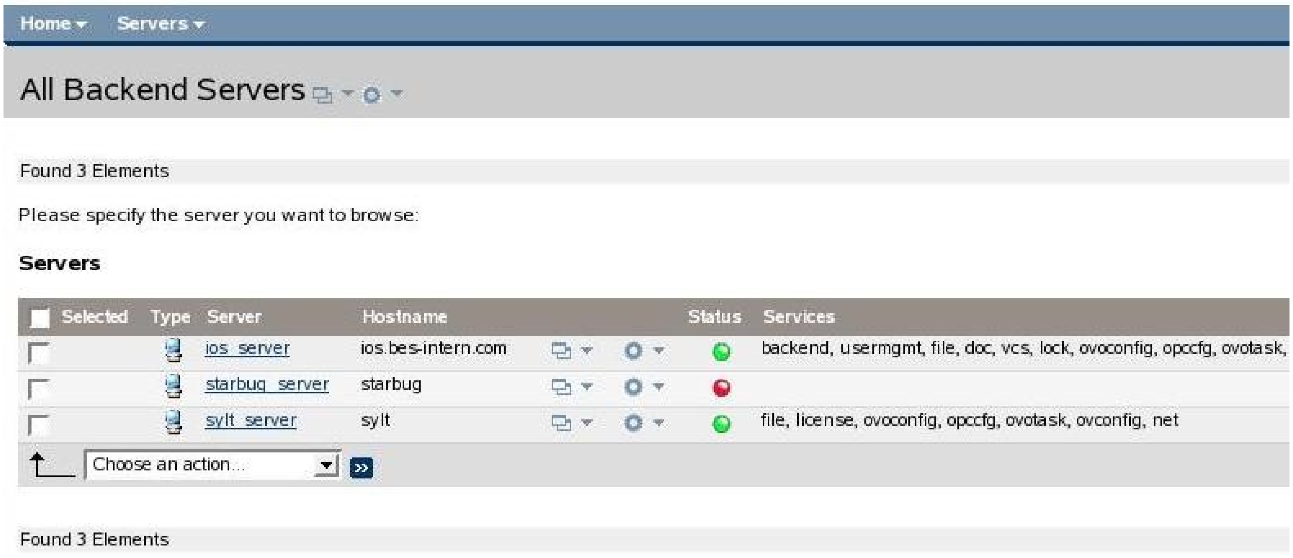
The output of the `midas.sh backend` command displays all the configuration information you need for the BackEnd you want to register manually, as illustrated in the following example:

```
[root@sylt:/opt/midas30] midas.sh backend
[...]
     [echo] Server sylt_server: OVO 8.2 HP-UX Itanium
     [echo]   Server Identifier: sylt_server
     [echo]   Hostname: sylt
     [echo]   Protocol: http
     [echo]   Port: 9661
     [echo]   Secure Communication: true
     [echo]   Platform: unix
     [echo]   Install Directory: /opt/midas30
[...]
```

In this example, the information you need for the manual registration of the new CVP BackEnd is:

● Server identifier:	sylt_server
● Hostname:	sylt
● Protocol used:	http (https, if *secure communication* is true)
● Port:	9661

If the registration of the new BackEnd succeeds, CVP displays a message confirming the success of the registration operation and the name of the newly registered BackEnd server appears in the list of BackEnd servers together with a green dot in the Status column indicating the system is running and available. If the BackEnd server status is green, you can select the BackEnd as the current server and browse any configuration data that is available. Note that a red dot in the Status column indicates that there are problems communicating with the listed BackEnd, as illustrated in the following example:



All Backend Servers

Found 3 Elements

Please specify the server you want to browse:

Servers

Selected	Type	Server	Hostname			Status	Services
		ios_server	ios.bes-intern.com			🟢	backend, usermgmt, file, doc, vcs, lock, ovoconfig, opccfg, ovotask,
		starbug_server	starbug			🔴	
		sylt_server	sylt			🟢	file, license, ovoconfig, opccfg, ovotask, ovconfig, net

Found 3 Elements

For more information about troubleshooting problems with BackEnd servers listed in the All BackEnd Server list, see 10 Troubleshooting.

7 External software

The following section explain the integration of additional software products into CVP. All are optional but required if the according CVP functionality is desired.

All have in common that the CVP software package only provides the integration interface but not the software itself (mostly for licensing reasons, e.g. CVS is open-source but licensed as GPL, therefore it cannot be included into the installer). To use them, please download the software from the recommended location and install/configure as described below.

7.1 Download locations

For the UNIX platforms, most open-source software can be downloaded as pre-compiled installable packages from some porting center or is available with a standard OS media kit.

Watch out for run-time dependencies, which have to be downloaded and installed as well. Some popular locations are:

OS	Download location	Installation Method
HP-UX	http://hpux.asknet.de	swinstall
Solaris	http://www.sunfreeware.com	pkgadd
Linux	Typically included in Linux distribution	rpm

7.2 Concurrent versioning system (CVS)

CVS (Concurrent versioning system) is a widely distributed, popular software to maintain a revision history of source data. Usually this applies to programming source code but can be used for any other data as well. Within the CVP context, it will be used to store OVO configuration data.

CVP already contains all related functionality, but the CVS software itself must be installed. By default, a local CVS repository will be created during installation of a CVP WebApp server.

It is recommended to install CVS **before** CVP and specify the location of the CVS executable correctly when installing the CVP WebApp. If CVS is not present when installing CVP but CVS is desired you must update the configuration CVP later.

The general home page of the CVS project is http://www.cvshome.org. Please review this page for documentation, examples and all other kinds of resources.

Alternatively, CVSNT can be used (see http://www.march-hare.com/cvspro). CVSNT behaves slightly different compared with *standard* CVS and some extra configuration steps may be needed (see below). Also, please review the CVSNT documentation.

7.2.1 Installing CVS on a UNIX WebApp

7.2.1.1 Using standard CVS

Download the CVS software from the location stated above in chapter Download locations and use the native installation method to install CVS on the target system. The download locations listed above provide a *standard* CVS implementation.

No extra configuration steps are needed.

7.2.1.2 Using CVSNT

After installing the CVSNT software and CVP, check the following:

- Make sure, the CVSNT lock server daemon is running on the CVP WebApp, whenever CVS is accessed.
- The CVP repository must be configured in `vcs.properties` as:

```
cvs.cvsRoot=:local:/opt/midas30/data/repository
cvs.cvsnt=true
```

- The CVS repository must be registered. If the *CVSNT* check-box has been selected in the CVP installer, the `wrapper.conf` contains a `--allow-root <CVS repository>` statement as shown in the following example:

```
wrapper.java.additional.7=-DEnv-CVS_EXE="%MIDAS_HOME%/bin/cvs
--allow-root /opt/midas30/data/repository"
wrapper.java.additional.7.stripquotes=TRUE
```

  Alternatively, the CVS repository can be registered by updating the CVSNT configuration.

These steps are not necessary with the regular CVS on UNIX systems.

7.2.2 Installing CVS on a Windows WebApp

There are several CVS implementations for Windows available, which may work with CVP, however CVP has been tested with CVSNT version 2.5.03.2382 only.

Download CVSNT from http://www.march-hare.com/cvspro and install it on the Windows WebApp system. It is recommended to install CVSNT **before** CVP and to specify the location of the `cvs.exe` program in the CVP installer. In this case, no additional steps are required, but nevertheless please verify:

- Make sure, that the *CVSNT lock server* is running as Windows service. The *CVSNT Service* is not required if only the local CVP WebApp accesses the CVS repository. See below for details.
- The CVP repository must be configured in `vcs.properties` as:

```
cvs.cvsRoot=:local:C:/MIDAS/data/repository
cvs.cvsnt=true
```

  Forward slashes “/” must be used, even on Windows systems.

- The location of the CVSNT executable `cvs.exe` and the CVP CVS repository must be registered. Check the properties in the CVP `wrapper.conf`:

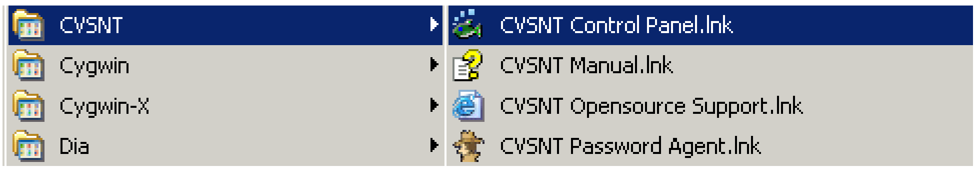
```
wrapper.java.additional.7=-DEnv-CVS_EXE="C:\Programs\CVSNT\cvs.exe
--allow-root C:/MIDAS/data/repository"
wrapper.java.additional.7.stripquotes=TRUE
```

  The complete property definition must in one single line (the first line in the example above is broken due to space constraints only).

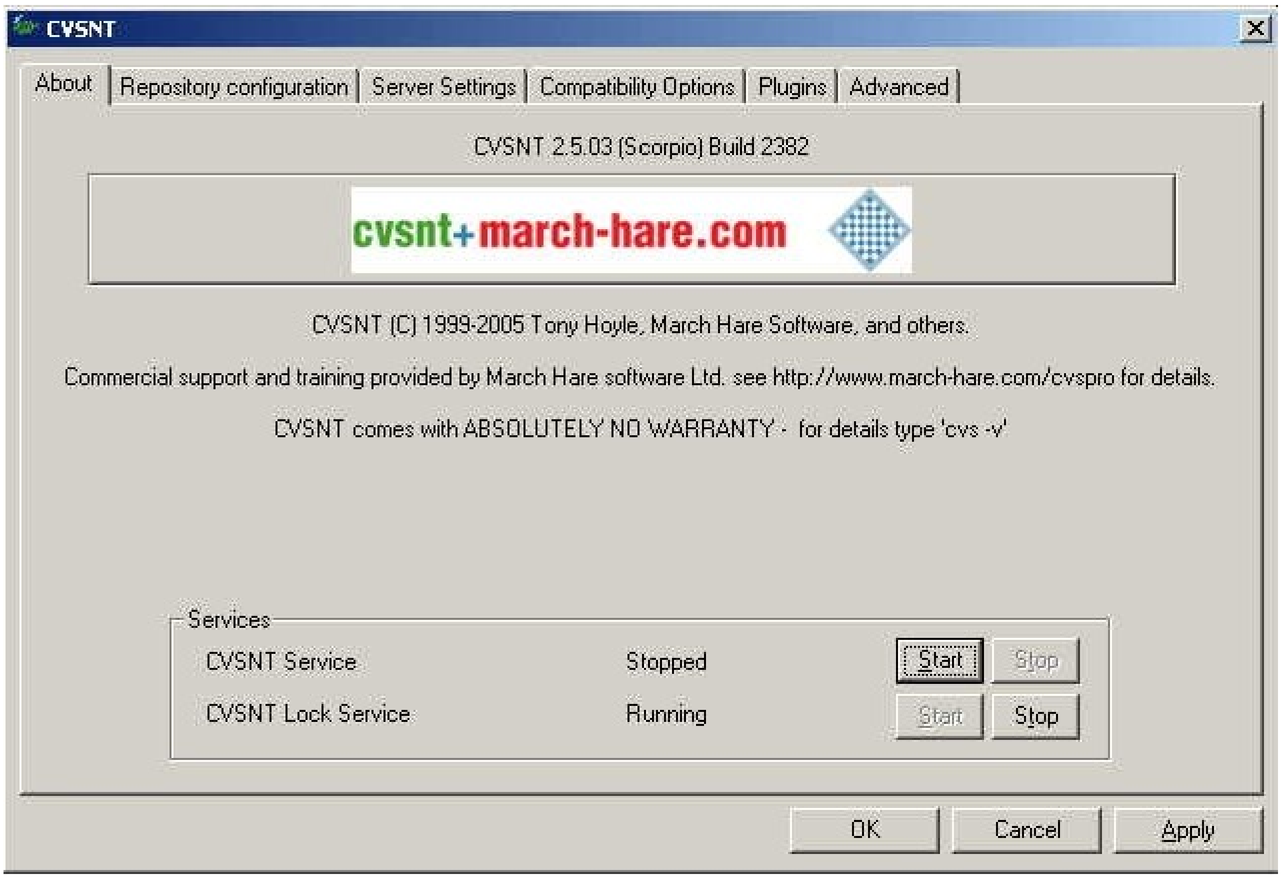
In addition, the same repository path as in `vcs.properties` must be specified after the `--allow-root` parameter. It must equal **literally** the repository path in `vcs.properties`. This applies to the slash characters as well as using upper and lowercase letters – both representations must be 100% identical.

If the CVP repository is referenced in the `wrapper.conf` property as described above, there is no need to register the repository in the CVSNT Control Panel. Just leave the **Repository configuration** tab of the CVSNT control panel empty.

To check and configure the CVSNT services, use the CVSNT Control Panel from the Start menu:



With version 2.5, the CVSNT Control Panel looks like this:



CVSNT 2.5.03 (Scorpio) Build 2382

CVSNT (C) 1999-2005 Tony Hoyle, March Hare Software, and others.

Commercial support and training provided by March Hare software Ltd. see http://www.march-hare.com/cvspro for details.

CVSNT comes with ABSOLUTELY NO WARRANTY - for details type 'cvs -v'

CVSNT Service	Stopped
CVSNT Lock Service	Running

For local access the only required CVSNT service is the `CVSNT Lock Service`. The `CVSNT Service` is not needed and can be stopped (unless you also want to connect to this repository remotely).

The `CVSNT Service` can also be disabled persistently in the Windows Service control panel.

7.3 Secure Shell (SSH)

For secure data transfer between CVP servers (e.g. WebApp and BackEnd systems) the use of SSH is supported.

Currently, on each involved system, the *Secure Shell (SSH)* software must be installed by the user. On systems acting as SSH server (opposite peer of the SSH client where the transfer gets initiated) the **sshd** must be running and configured. On both sides (SSH client and servers), the **ssh-keygen** binary is needed during CVP installation for key generation.

On the SSH client, the actual transfer happens using the *jsch* SSH implementation (see http://www.jcraft.com/jsch) which is built into the CVP server, i.e. the local SSH software on a SSH server is needed only during key generation.

Both sides must be configured in a way that the SSH client can connect the SSH server without user interaction. This is accomplished using public-key authentication. The necessary steps are described in chapter 9.9.2 Configuring SSH transfer.

Generally it is recommended to use the latest SSH version including all up-to-date security features. The only tested and supported SSH implementation with CVP is OpenSSH2 (see http://www.openssh.org).

7.3.1 Installing SSH on UNIX systems

Download the software from the according location and use the native installation method to install SSH on the target system.

7.3.2 Installing SSH on a Windows WebApp

SSH on Windows systems is currently not supported.

7.3.3 Testing the SSH set-up

Before starting to integrate SSH into CVP, perform some standalone tests first. After having installed the SSH software on the according systems:

1. Verify whether the midas user exists on both systems to be connected via SSH. The HOME directory of the user must be the <MIDAS_HOME> installation root.
2. Check whether the sshd process is running on the SSH server side
3. On the SSH client side, try to remotely execute the pwd command:

```
# su – midas -c “ssh midas@<ssh_server_node> pwd”
```

   If the set-up is generally correct you should be prompted for the password of the OS user midas on the SSH server system.
   After entering the password the path of the $HOME directory of the user midas on the SSH server system should be displayed as result of the pwd command. This path must equal the CVP installation root.

If this works properly, SSH can be integrated into CVP.

7.3.4 Configuring SSH in CVP

The main task of configuring SSH for CVP is to enable public-key-based connection. This involves mainly the exchange of the public keys.

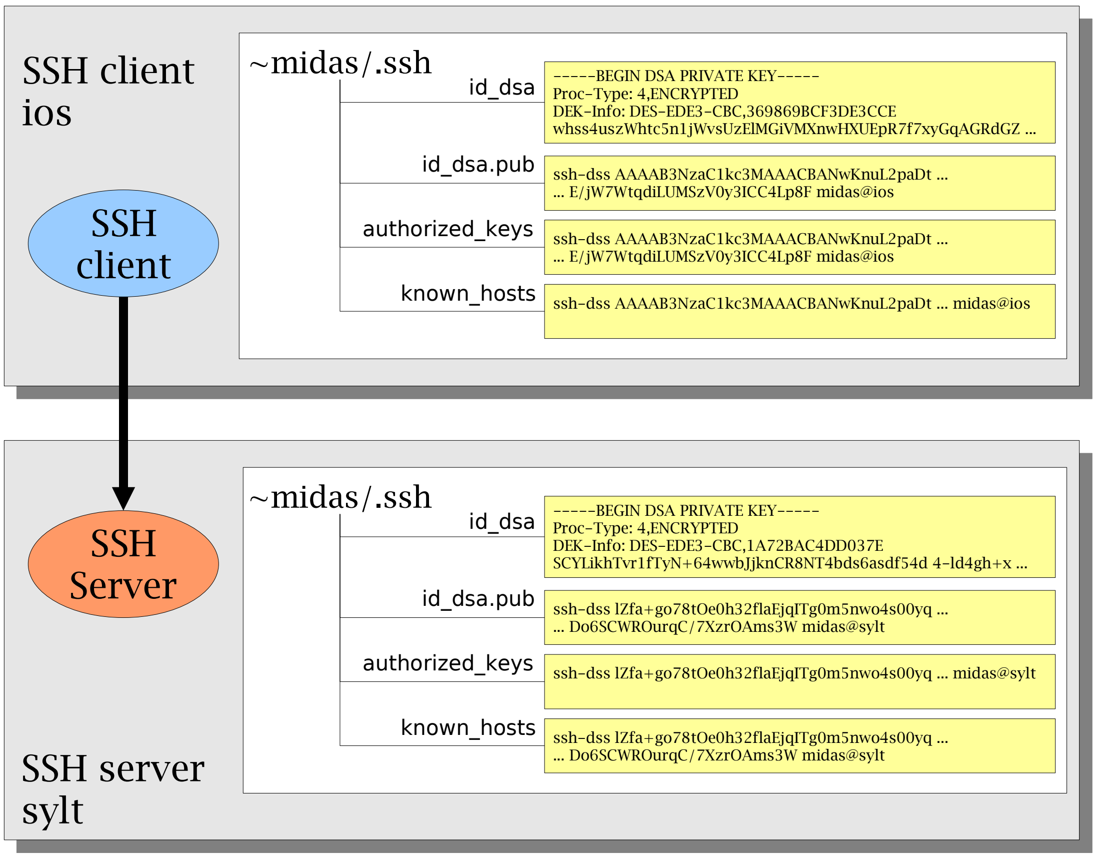
7.3.4.1 Initial set-up

During CVP installation default key pairs are generated automatically (if the installing user has enabled SSH transfer) which have to be exchanged to allow a non-interactive connection.

The generated key files are located in `<MIDAS_HOME>/.ssh` (`<MIDAS_HOME>` must be the `$HOME` directory of the local OS user `midas`) with the name `id_dsa` for the private key and `id_dsa.pub` for the public key. DSA is used as default encryption method.

The local public key is also automatically added to the file `authorized_keys` in `<MIDAS_HOME>/.ssh` as shown above. This allows local SSH connectivity.

This initial set-up will be generated during installation and is illustrated in the following picture (assuming OpenSSH):


SSH client
ios
SSH client
~midas/.ssh
id_dsa
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,369869BCF3DE3CCE
whss4uszWhtc5n1jWvsUzElMGiVMXnwHXUEpR7f7xyGqAGRdGZ ...
id_dsa.pub
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ...
... E/jW7WtqdiLUMSzV0y3ICC4Lp8F midas@ios
authorized_keys
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ...
... E/jW7WtqdiLUMSzV0y3ICC4Lp8F midas@ios
known_hosts
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ... midas@ios
SSH Server
SSH server
sylt
~midas/.ssh
id_dsa
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,1A72BAC4DD037E
SCYLikhTvr1fTyN+64wwbJjknCR8NT4bds6asdf54d 4-ld4gh+x ...
id_dsa.pub
ssh-dss lZfa+go78tOe0h32flaEjqITg0m5nwo4s00yq ...
... Do6SCWROurqC/7XzrOAms3W midas@sylt
authorized_keys
ssh-dss lZfa+go78tOe0h32flaEjqITg0m5nwo4s00yq ... midas@sylt
known_hosts
ssh-dss lZfa+go78tOe0h32flaEjqITg0m5nwo4s00yq ...
... Do6SCWROurqC/7XzrOAms3W midas@sylt

If this failed or needs to be re-done later, perform the following command:

```
# <MIDAS_HOME>/midas.sh ant -f run.xml intern.ssh_init
```

This can also be done to change the general SSH behavior within CVP. This is defined in the main configuration file related to SSH `<MIDAS_HOME>/conf/ant/config.xml` which contains the related entries, for example:

```
# cat <MIDAS_HOME>/conf/ant/config.xml
[...]
  <!-- ssh -->
  <property name="dir.openssh" value="/usr/bin"/>
  <property name="ssh.trust" value="true"/>
  <property name="ssh.type" value="dsa"/>
  <property name="ssh.key" value="id_${ssh.type}"/>
  <property name="ssh.user" value="midas"/>
[...]
```

The property `dir.openssh` determines the location where the SSH binaries are expected. This value is needed only during key generation.

The property `ssh.type` determines the SSH encryption algorithm. This is important when generating the SSH keys (normally during CVP installation) and later when referring to the private key file to be used during an actual transfer. The private key file used is `<MIDAS_HOME>/.ssh/id_<ssh.type>`. Default encryption method is `dsa`, alternatively it can be changed to `rsa` for RSA-encrypted keys. If the encryption method has been changed, a new key pair must be generated and the public key has to be added to the `authorized_keys` file on all SSH peer nodes (see below for details).

By default CVP trusts all unknown hosts automatically. This corresponds to automatically confirming the question whether to connect to a new host when using SSH on the command line, for example:

```
# ssh new-host
The authenticity of host 'new-host (192.168.100.1)' can't be
established.
RSA key fingerprint is 88:6c:c9:35:fa:02:06:3c:27:f4:12:6f:39:a0:6d:f5.
Are you sure you want to continue connecting (yes/no)? Yes
Warning: Permanently added 'new-host,192.168.100.1' (RSA) to the list
of known hosts.
```

Setting the property `ssh.trust` to true always answers *yes* to the question whether to connect to a previously unknown host. Also, the public key of this unknown host will be added to the `known_hosts` file. If this automatic trust is not desired set the property `ssh.trust` to `false`.

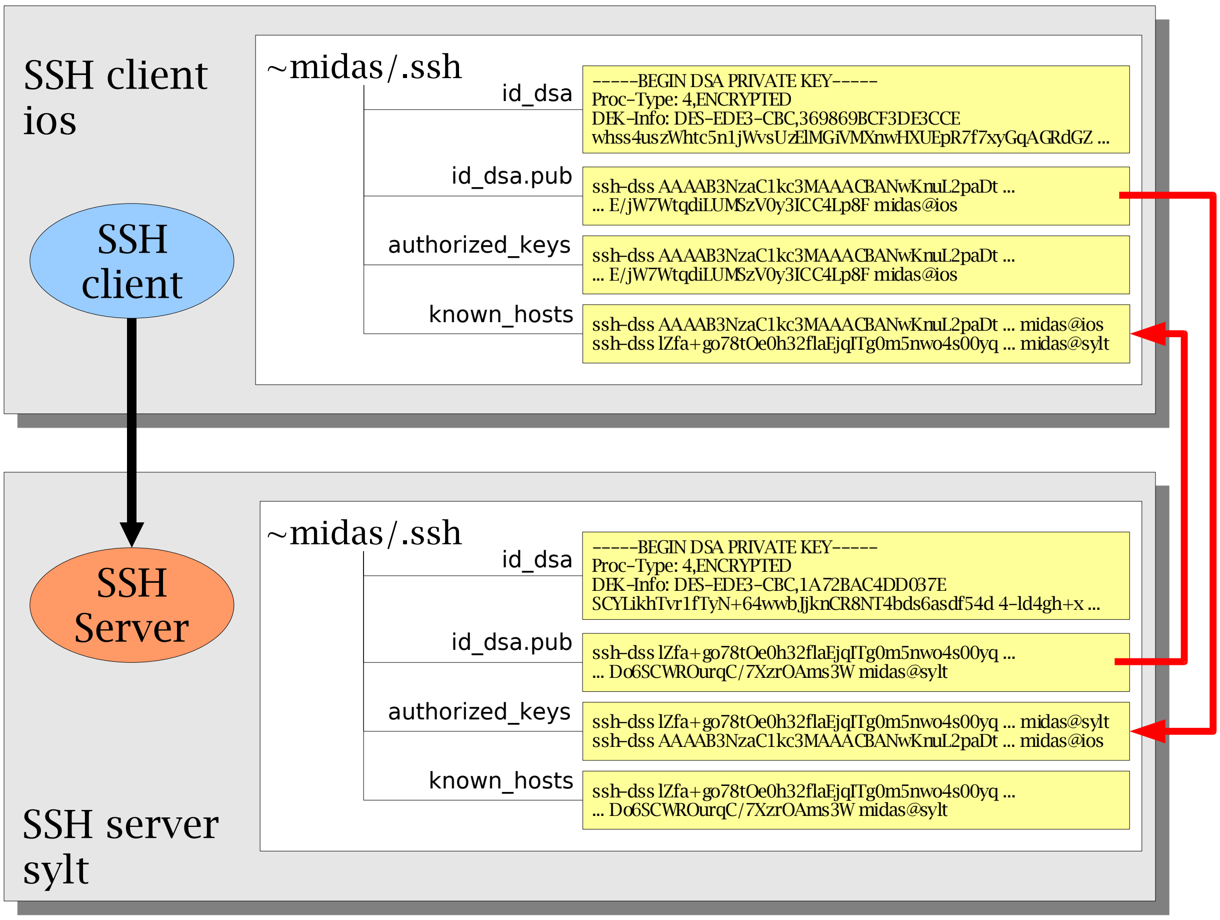
The property `ssh.user` controls as which user account the SSH connection will be established. This equals the following SSH command on shell level:

```
# ssh <ssh.user>@<target host>
```

Currently this user name is always `midas`.

7.3.4.2 Exchanging SSH keys

With OpenSSH, a set-up like the following (in this example, the system *ios* is SSH client and *sylt* is SSH server) has to be accomplished:


SSH client ios
SSH client
~midas/.ssh
id_dsa
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,369869BCF3DE3CCE
whss4uszWhtc5n1jWvsUzElMGiVMXnwHXUEpR7f7xyGqAGRdGZ ...
id_dsa.pub
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ...
... E/jW7WtqdiLUMSzV0y3ICC4Lp8F midas@ios
authorized_keys
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ...
... E/jW7WtqdiLUMSzV0y3ICC4Lp8F midas@ios
known_hosts
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ... midas@ios
ssh-dss lZfa+go78tOe0h32flaEjqlTg0m5nwo4s00yq ... midas@sylt
SSH Server
~midas/.ssh
id_dsa
-----BEGIN DSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,1A72BAC4DD037E
SCYLikhTvr1fTyN+64wwbJjknCR8NT4bds6asdf54d 4-ld4gh+x ...
id_dsa.pub
ssh-dss lZfa+go78tOe0h32flaEjqlTg0m5nwo4s00yq ...
... Do6SCWROurqC/7XzrOAms3W midas@sylt
authorized_keys
ssh-dss lZfa+go78tOe0h32flaEjqlTg0m5nwo4s00yq ... midas@sylt
ssh-dss AAAAB3NzaC1kc3MAAACBANwKnuL2paDt ... midas@ios
known_hosts
ssh-dss lZfa+go78tOe0h32flaEjqlTg0m5nwo4s00yq ...
... Do6SCWROurqC/7XzrOAms3W midas@sylt
SSH server sylt

Now transfer the public key `id_dsa.pub` from the SSH client system (ios) to the SSH server system (sylt) and append it to the file `<MIDAS_HOME>/.ssh/authorized_keys` there.

CAUTION: Make sure not to modify the text in any way! Particularly, watch out to not introduce any new-line or carriage-return characters.

Example (***ios*** is the SSH client and ***sylt*** the SSH server):

```
# root@ios> cat /opt/midas30/.ssh/id_dsa.pub
ssh-dss AAAAB3NzaC1kc3MAA ... p8FxDFqrvFe5chcsYeV== MIDAS server key midas@ios

# root@sylt> vi /opt/midas30/.ssh/authorized_keys
ssh-dss IDmh1Fy6Ge8Gnr ... DLVn35JlGQQhWCWD4o2MV== MIDAS server key midas@sylt
ssh-dss AAAAB3NzaC1kc3MAA ... p8FxDFqrvFe5chcsYeV== MIDAS server key midas@ios
```

Reversely, do the same and import the SSH server's public key into `<MIDAS_HOME>/.ssh/known_hosts` on the SSH client system. This step can be also accomplished by performing a manual SSH call and confirming the connection:

```
# su – midas -c “ssh midas@sylt pwd”
The authenticity of host 'sylt (192.168.100.200)' can't be established.
```

```
RSA key fingerprint is 60:a0:3c:ca:9a:90:a0:3a:70:fa:5b:2a:60:e0:9f:d4.
Are you sure you want to continue connecting (yes/no)? Yes
Warning: Permanently added 'sylt,192.168.100.200' (RSA) to the list of known
hosts.
[...]
```

Note, that the directory and all its file must belong to user `midas` and that the private key and `authorized_keys` file must have restricted permissions, for example as shown below:

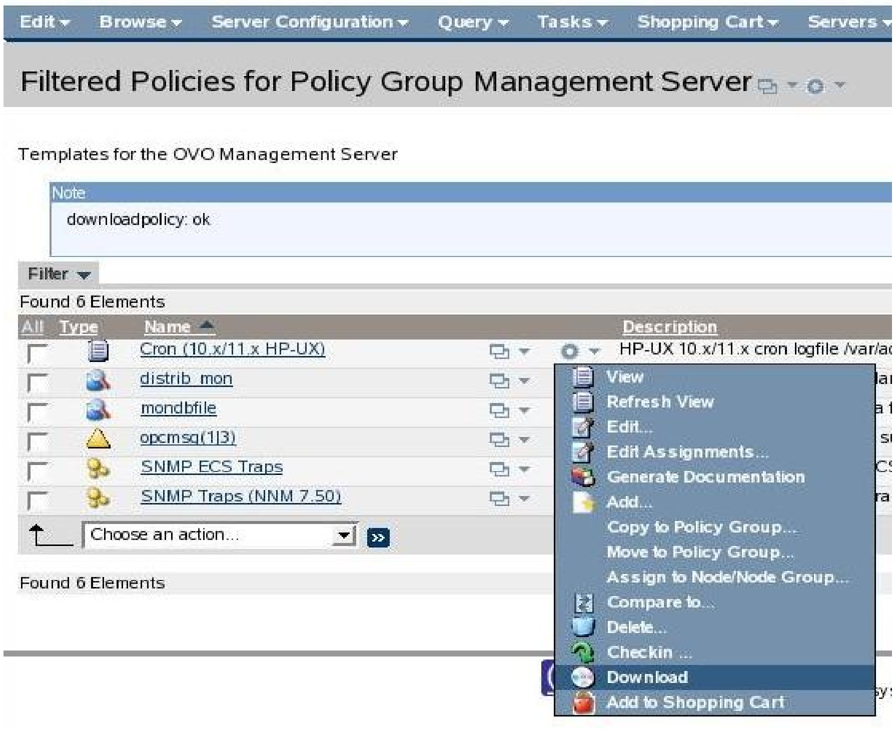
```
# ls -l /opt/midas30/.ssh:
[...]
drwxr-xr-x   3 midas root  224 2006-08-14 14:45 .
-rw-------   1 midas root  744 2006-08-14 14:44 id_dsa
-rw-r--r--   1 midas root  600 2006-08-14 14:44 id_dsa.pub
-rw-------   1 midas root 5045 2006-11-22 12:25 known_hosts
-rw-------   1 midas root 5045 2006-11-22 12:26 authorized_keys
```

Now, test the SSH connection again as described above (Testing the SSH set-up). It is important that you are not queried for a password or anything else. Otherwise the key exchange and file permissions weren't successfully set.

7.3.4.3 Testing SSH in CVP

To finally verify the entire CVP integration, perform a download and transfer via SSH from the BackEnd to the WebApp. To do this, perform the following steps.

1. Download an arbitrary element to the clipboard on the BackEnd (here the policy “Cron (10.x/11.x HP-UX)” is used as an example):



Filtered Policies for Policy Group Management Server

Templates for the OVO Management Server

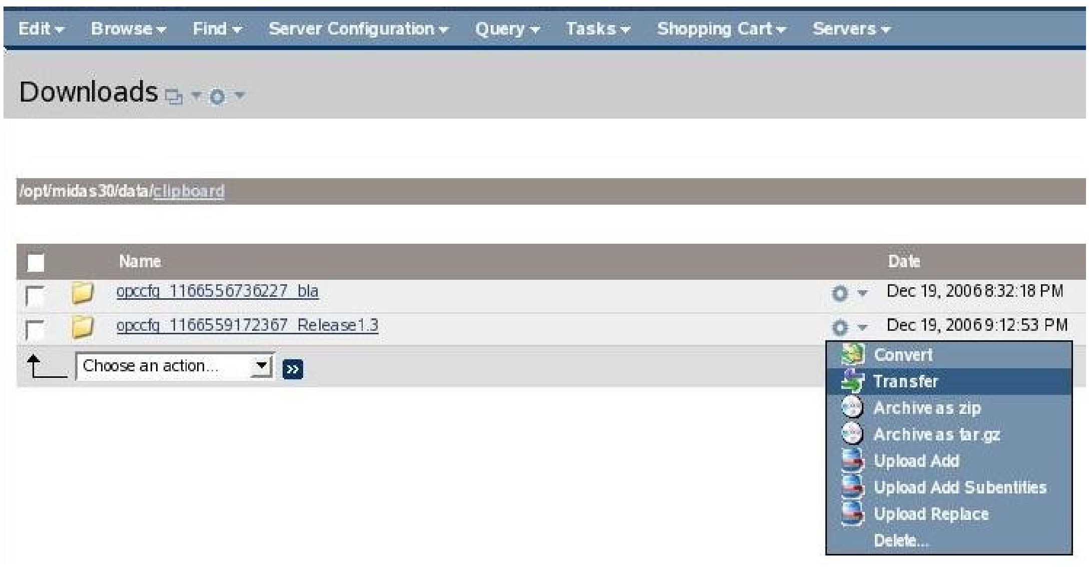
Note
downloadpolicy: ok

Found 6 Elements

All	Type	Name	Description
		Cron (10.x/11.x HP-UX)	HP-UX 10.x/11.x cron logfile /var/ac
		distrib_mon	
		mondbfile	
		opcmsg(1|3)	
		SNMP ECS Traps	
		SNMP Traps (NNM 7.50)	

Found 6 Elements

2. Navigate to the clipboard using the Browse -> Downloads menu and select the downloaded element and initiate the transfer:

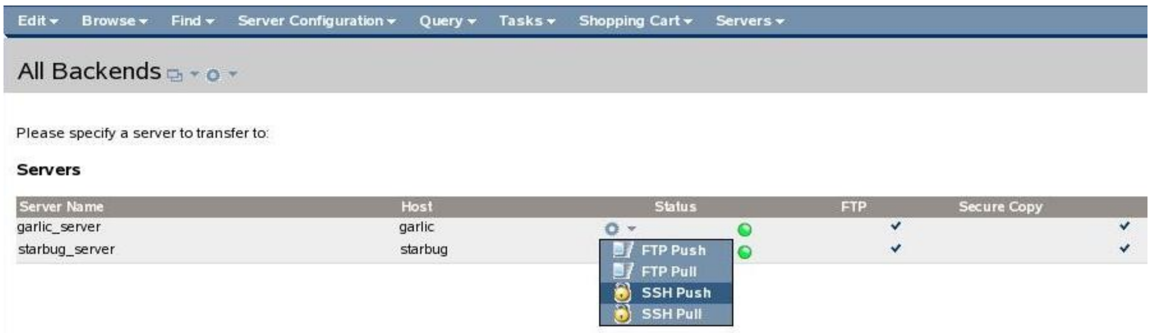


Downloads

/opt/midas30/data/clipboard

	Name	Date
	opccfg_1166556736227_bla	Dec 19, 2006 8:32:18 PM
	opccfg_1166559172367_Release1.3	Dec 19, 2006 9:12:53 PM

3. Select the WebApp as target (here garlic_server) and choose SSH push:



All Backends

Please specify a server to transfer to:

Servers

Server Name	Host	Status	FTP	Secure Copy
garlic_server	garlic	●	✓	✓
starbug_server	starbug	●	✓	✓

An *SSH Push* will initiate the SSH transfer from the BackEnd (ios_server, acting as SSH client) **to** the WebApp (garlic_server, acting as SSH server).

4. Select the WebApp server garlic_server, and again browse the clipboard. The transferred element should appear identically to as it has been present on the source system.

7.4 Authentication software

Authentication of CVP users happens on the CVP WebApp, to which the GUI connects to. Thus, all steps described in the following apply to WebApps only.

With the current product version, CVP supports authentication using LDAP or any authentication service which can be integrated into PAM.

NOTE: Authentication covers only the process of validating the user account with a password. It does not include any authorization control (user's capabilities). Authorization is implemented exclusively in CVP by defining CVP user roles.

Therefore, whenever setting up a new CVP user, make sure that the account exists in both CVP and the external authentication system. Furthermore, make sure that the CVP user is member of at least one CVP group which has at least one CVP user role assigned.

To use an external authentication software like LDAP or PAM in CVP additional software (for example, the LDAP client) may be required on the CVP WebApp. Install and configure this software as needed. More details are presented below.

7.4.1 PAM integration

To authenticate CVP users through PAM, no extra software is needed. CVP already includes the open-source module `jpam` (see http://jpam.sourceforge.net for details).

NOTE: PAM is available on UNIX systems only.

However, PAM is just an interface linking software providing authentication services (like LDAP, Kerberos, UNIX passwd) to consumer applications like CVP. Therefore, possibly software modules implementing the actual authentication service may be needed.

To configure PAM, perform the following steps:

1. Decide, which authentication method to use. If needed, install required software modules and configure them. Test the authentication service standalone, i.e. outside of the CVP context.
2. Configure all CVP user accounts in the authentication service.
3. Configure PAM to route CVP authentication requests to the desired authentication service. The PAM service name is **`midas`**.
4. Switch CVP to PAM authentication by setting the property `userreq-transformer.targetService` to the value `pam` as shown in the following example:

```
# vi <MIDAS_HOME>/conf/webapp.properties
userreq-transformer.targetService=pam
```

5. Deploy the `midas-wapam-sa.zip` service assembly:

```
# cp <MIDAS_HOME>/assemblies/midas-wapam-sa.zip
     <MIDAS_HOME>/deploy
```

6. Restart the WebApp:

```
# <MIDAS_HOME>/midas.sh restart
```

Configuring the actual authentication software depends on this software itself and the OS the CVP WebApp is running on. In the following example, the default authentication mechanism on a Linux system is used.

7.4.1.1 Example: UNIX passwd

By default, the default authentication mechanism on Linux is UNIX passwd, which in turn is always available (normally) and there is no need to install and configure anything.

Create a user account and set the password, if not done yet, for example the user `tge`:

```
# useradd tge

# passwd tge
Changing password for tge.
New Password: *******
Reenter New Password: *******
```

```
Password changed.
```

Other UNIX-related parameters like home directory or shell are not needed for CVP.

Configure PAM to route CVP authentication requests to UNIX passwd by creating the file `/etc/pam.d/midas` containing:

```
auth     include        common-auth
account  include        common-account
password include        common-password
session  include        common-session
```

The file name must equal the required PAM service name (for CVP this is `midas`). The contents of the file determines the authentication module to be used. For more details regarding PAM configuration please review the OS-specific PAM documentation.

On other UNIX implementations, the PAM service configuration may be slightly different. For example, to route CVP authentication requests with the service name `midas` on HP-UX, you must configure the following in `/etc/pam.conf`:

```
[...]
midas    auth    required       libpam_unix.so.1
midas    account required       libpam_unix.so.1
[...]
```

For further details like advanced PAM capabilities (for example using multiple and/or optional authentication services) refer to the OS-specific PAM documentation.

7.4.2 LDAP integration

CVP supports user authentication through LDAP. CVP includes the open-source component “Acegi Security System for Spring Project” (see http://acegisecurity.org for details).

NOTE: The built-in LDAP client can also refer to a Windows ADS server.

Currently only basic authentication or user accounts is supported. No additional LDAP features like group membership etc. are used.

To configure LDAP authentication in CVP, perform the following steps:

1. Configure all CVP user accounts on the LDAP server.
2. Configure the desired LDAP server in the CVP properties file `<MIDAS_HOME>/conf/ldap.properties` as shown in the following example:

```
# The LDAP URL
# Format: ldap://<host>:<port>/<base dn>
ldap.url=ldap://localhost:389/dc=blue-elephant-systems,dc=com

# Manager DN for login
ldap.managerDn=cn=Manager,dc=blue-elephant-systems,dc=com

# Manager password
ldap.managerPassword=secret
```

```
# The patterns for searching users (pipe-separated)
ldap.authenticationDnPatterns=sn={0},ou=People
```

Refer to the LDAP documentation for additional details about the individual parameters.

3. Switch CVP to LDAP authentication by setting the property `userreq-transformer.targetService` to the value `ldap` as shown in the following example:

```
# vi <MIDAS_HOME>/conf/webapp.properties
userreq-transformer.targetService=ldap
```

4. Deploy the `midas-waldap-sa.zip` service assembly:

```
# cp <MIDAS_HOME>/assemblies/midas-waldap-sa.zip
     <MIDAS_HOME>/deploy
```

5. Restart the WebApp:

```
# <MIDAS_HOME>/midas.sh restart
```

8 CVP Maintenance

8.1 Auditing

Auditing covers the tasks of keeping a log of who has done what and when. This is supported and enabled by default in CVP. Technically, log4j is used for this, therefore all configuration regarding auditing has to be done by modifying log4j configuration files. For details see chapter 9.3 Logging configuration.

For auditing, there are two log levels:

INFO	one line per operation representing a summary, what has been done by whom and when, however not containing all details
DEBUG	the full internal requests and responses exchanged for performing an operation.

An example of the according audit records (INFO level) looks like this:

```
INFO,2006-09-19 10:40:49,295,...,modifyresponse,...,
ios_server,ovoconfig,tge,modify,ovo:policy,null,opcmsg(1|3),opcmsg
```

Here, the user *tge* has modified the *opcmsg* policy named *opcmsg(1|3)* on the BackEnd *ios_server*.

With DEBUG level, there is the full data flow captured and logged. This contains all details about the objects being modified. The same operation as above now looks like this:

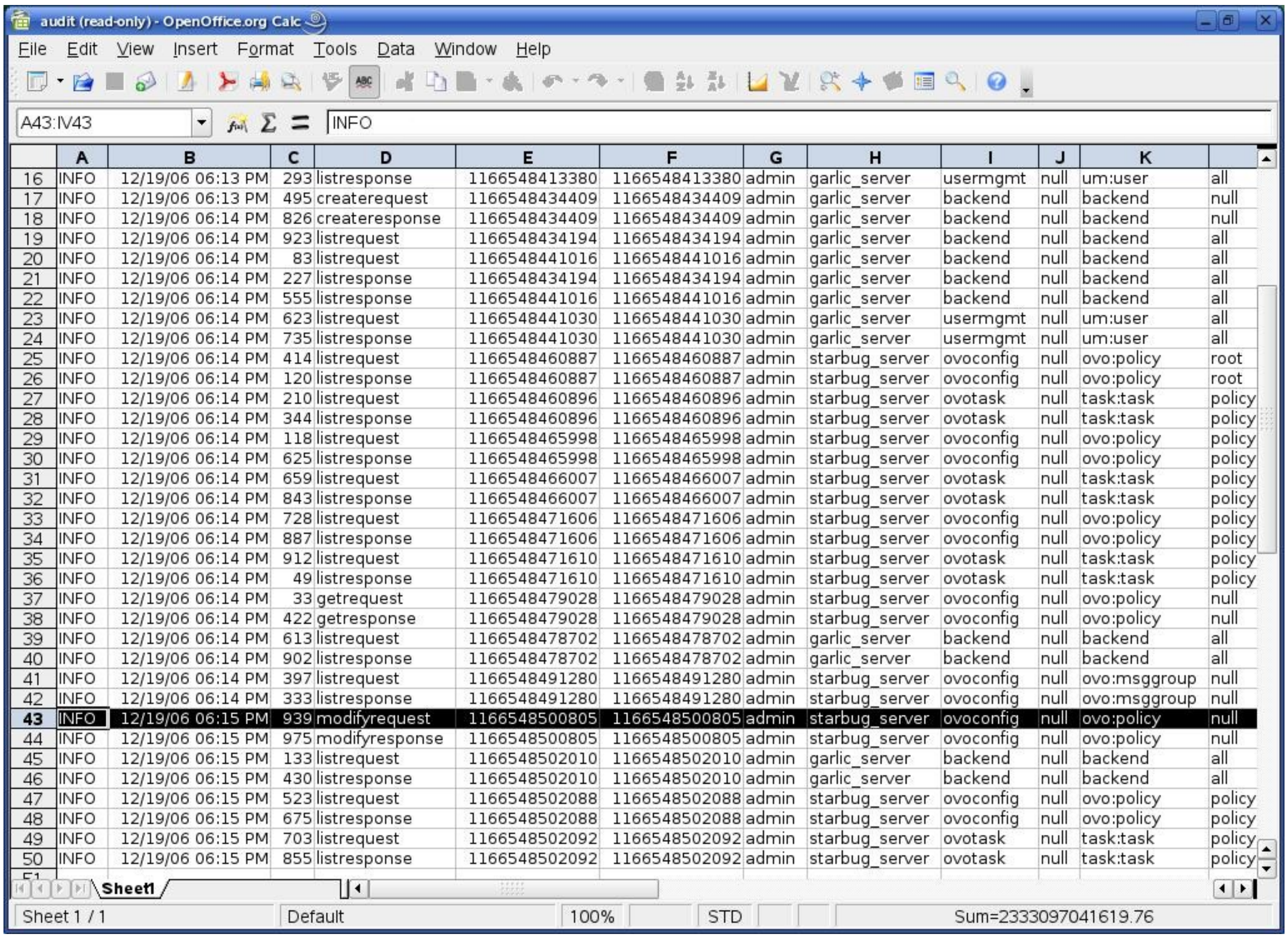
```
DEBUG,2006-09-19 10:40:49,294,...,
 <modifyresponse ...
  [...]
  <operation do:type="String">modify</operation>
  <objectclass do:type="String">ovo:policy</objectclass>
  <user do:type="String">tge</user>
  <content ...
   <ovo:policy ...
    <ovo:policytype do:type="String">opcmsg</ovo:policytype>
    <ovo:name do:type="String">opcmsg(1|3)</ovo:name>
[...]
```

The audit files are written in `<MIDAS_HOME>/logs/audit`. The active log file currently being written is `audit.log`. By default, this file will be rolled daily yielding files like `audit.log.2006-11-20`. This behavior can be controlled by editing `<MIDAS_HOME>/conf/log4j.xml`:

```
<appender name="audit" class="org.apache.log4j.DailyRollingFileAppender">
  <param name="File" value="logs/audit/audit.log"/>
  <param name="DatePattern" value="'.'yyyy-MM-dd"/>
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%p,%d,%m%n"/>
  </layout>
</appender>
```

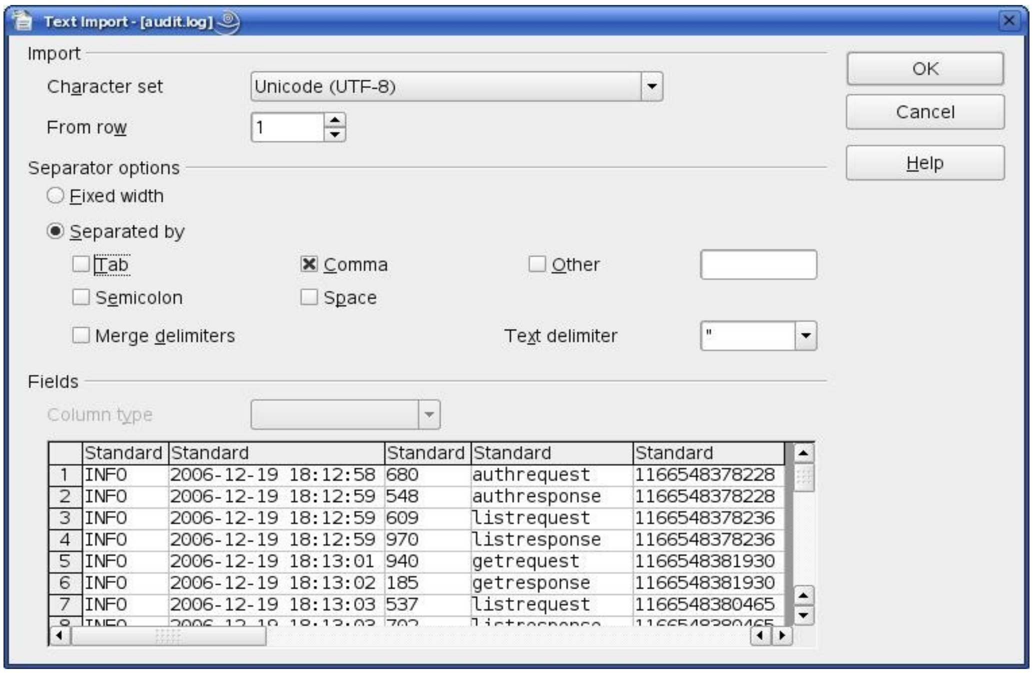
```
[...]

<logger name="com.bes.itm.comp.servicemix.DORequestAuditTransformer"
        additivity="false">
  <level value="DEBUG"/>
  <appender-ref ref="audit"/>
</logger>
```

CVP itself currently does not provide any tools to review the audit records. However, the audit log files can be loaded as CSV files into spreadsheet applications like MS Excel or OpenOffice for evaluation. This results in a spreadsheet as shown in the following example:


audit (read-only) - OpenOffice.org Calc
File Edit View Insert Format Tools Data Window Help
A43:IV43
INFO
Sheet1
Sheet 1 / 1
Default
100%
STD
Sum=2333097041619.76

	A	B	C	D	E	F	G	H	I	J	K	
16	INFO	12/19/06 06:13 PM	293	listresponse	1166548413380	1166548413380	admin	garlic_server	usermgmt	null	um:user	all
17	INFO	12/19/06 06:13 PM	495	createrequest	1166548434409	1166548434409	admin	garlic_server	backend	null	backend	null
18	INFO	12/19/06 06:14 PM	826	createresponse	1166548434409	1166548434409	admin	garlic_server	backend	null	backend	null
19	INFO	12/19/06 06:14 PM	923	listrequest	1166548434194	1166548434194	admin	garlic_server	backend	null	backend	all
20	INFO	12/19/06 06:14 PM	83	listrequest	1166548441016	1166548441016	admin	garlic_server	backend	null	backend	all
21	INFO	12/19/06 06:14 PM	227	listresponse	1166548434194	1166548434194	admin	garlic_server	backend	null	backend	all
22	INFO	12/19/06 06:14 PM	555	listresponse	1166548441016	1166548441016	admin	garlic_server	backend	null	backend	all
23	INFO	12/19/06 06:14 PM	623	listrequest	1166548441030	1166548441030	admin	garlic_server	usermgmt	null	um:user	all
24	INFO	12/19/06 06:14 PM	735	listresponse	1166548441030	1166548441030	admin	garlic_server	usermgmt	null	um:user	all
25	INFO	12/19/06 06:14 PM	414	listrequest	1166548460887	1166548460887	admin	starbug_server	ovoconfig	null	ovo:policy	root
26	INFO	12/19/06 06:14 PM	120	listresponse	1166548460887	1166548460887	admin	starbug_server	ovoconfig	null	ovo:policy	root
27	INFO	12/19/06 06:14 PM	210	listrequest	1166548460896	1166548460896	admin	starbug_server	ovotask	null	task:task	policy
28	INFO	12/19/06 06:14 PM	344	listresponse	1166548460896	1166548460896	admin	starbug_server	ovotask	null	task:task	policy
29	INFO	12/19/06 06:14 PM	118	listrequest	1166548465998	1166548465998	admin	starbug_server	ovoconfig	null	ovo:policy	policy
30	INFO	12/19/06 06:14 PM	625	listresponse	1166548465998	1166548465998	admin	starbug_server	ovoconfig	null	ovo:policy	policy
31	INFO	12/19/06 06:14 PM	659	listrequest	1166548466007	1166548466007	admin	starbug_server	ovotask	null	task:task	policy
32	INFO	12/19/06 06:14 PM	843	listresponse	1166548466007	1166548466007	admin	starbug_server	ovotask	null	task:task	policy
33	INFO	12/19/06 06:14 PM	728	listrequest	1166548471606	1166548471606	admin	starbug_server	ovoconfig	null	ovo:policy	policy
34	INFO	12/19/06 06:14 PM	887	listresponse	1166548471606	1166548471606	admin	starbug_server	ovoconfig	null	ovo:policy	policy
35	INFO	12/19/06 06:14 PM	912	listrequest	1166548471610	1166548471610	admin	starbug_server	ovotask	null	task:task	policy
36	INFO	12/19/06 06:14 PM	49	listresponse	1166548471610	1166548471610	admin	starbug_server	ovotask	null	task:task	policy
37	INFO	12/19/06 06:14 PM	33	getrequest	1166548479028	1166548479028	admin	starbug_server	ovoconfig	null	ovo:policy	null
38	INFO	12/19/06 06:14 PM	422	getresponse	1166548479028	1166548479028	admin	starbug_server	ovoconfig	null	ovo:policy	null
39	INFO	12/19/06 06:14 PM	613	listrequest	1166548478702	1166548478702	admin	garlic_server	backend	null	backend	all
40	INFO	12/19/06 06:14 PM	902	listresponse	1166548478702	1166548478702	admin	garlic_server	backend	null	backend	all
41	INFO	12/19/06 06:14 PM	397	listrequest	1166548491280	1166548491280	admin	starbug_server	ovoconfig	null	ovo:msggroup	null
42	INFO	12/19/06 06:14 PM	333	listresponse	1166548491280	1166548491280	admin	starbug_server	ovoconfig	null	ovo:msggroup	null
43	INFO	12/19/06 06:15 PM	939	modifyrequest	1166548500805	1166548500805	admin	starbug_server	ovoconfig	null	ovo:policy	null
44	INFO	12/19/06 06:15 PM	975	modifyresponse	1166548500805	1166548500805	admin	starbug_server	ovoconfig	null	ovo:policy	null
45	INFO	12/19/06 06:15 PM	133	listrequest	1166548502010	1166548502010	admin	garlic_server	backend	null	backend	all
46	INFO	12/19/06 06:15 PM	430	listresponse	1166548502010	1166548502010	admin	garlic_server	backend	null	backend	all
47	INFO	12/19/06 06:15 PM	523	listrequest	1166548502088	1166548502088	admin	starbug_server	ovoconfig	null	ovo:policy	policy
48	INFO	12/19/06 06:15 PM	675	listresponse	1166548502088	1166548502088	admin	starbug_server	ovoconfig	null	ovo:policy	policy
49	INFO	12/19/06 06:15 PM	703	listrequest	1166548502092	1166548502092	admin	starbug_server	ovotask	null	task:task	policy
50	INFO	12/19/06 06:15 PM	855	listresponse	1166548502092	1166548502092	admin	starbug_server	ovotask	null	task:task	policy

If needed, make sure that the comma character is used as separator:



Import

Character set: Unicode (UTF-8)

From row: 1

Separator options

( ) Fixed width

(•) Separated by

[ ] Tab [✓ Comma] [ ] Other

[ ] Semicolon [ ] Space

[ ] Merge delimiters Text delimiter: "

Fields

Column type

	Standard	Standard	Standard	Standard	Standard
1	INFO	2006-12-19 18:12:58	680	authrequest	1166548378228
2	INFO	2006-12-19 18:12:59	548	authresponse	1166548378228
3	INFO	2006-12-19 18:12:59	609	listrequest	1166548378236
4	INFO	2006-12-19 18:12:59	970	listresponse	1166548378236
5	INFO	2006-12-19 18:13:01	940	getrequest	1166548381930
6	INFO	2006-12-19 18:13:02	185	getresponse	1166548381930
7	INFO	2006-12-19 18:13:03	537	listrequest	1166548380465
8	INFO	2006-12-19 18:13:03	702	listresponse	1166548380465

8.2 General maintenance tasks

Most CVP maintenance tasks are performed using the `midas.sh` command.

NOTE: `midas.sh` and `midas.bat` are both installed, no matter what operating system CVP is installed on. The presented examples use midas.sh. On Windows systems, use `midas.bat` instead.

The `midas.sh` command supports a large set of options. To display an overview, just run it without any arguments:

```
[root@ios:/opt/midas30] midas.sh
midas.sh [operation] [parameter]...

ant         - run a ant task with the buildin ant
backend     - show details of local backend
backup      - creates backup of MIDAS configuration (*.zip file)
check       - performance sanity checks
checksum    - generate checksum
clean       - remove logfiles and work files
config      - show detailed configuration of components
import      - import file into clipboard
init        - initialize XML DB (deletes old configuration!)
install     - install licenses
licenses    - list installed licenses
generate    - generate sitemap for documentation for documentation
              generation
patch       - apply a fixpack
post        - post request to server
```

```
restart    - restart start the server
restore    - restore a MIDAS configuration backup from a zip file
servicemix - show servicemix deployments
start      - start the server
status     - show status of server
stop       - stop the server
support    - collect support information
unpatch    - remove a fixpack
version    - print version information
xmldb      - XML DB administration
```

To display more information about each sub-command, use the ***help*** option. In the following, more details about each sub-command is given.

8.2.1 Displaying server information

The ***backend*** sub-command shows all important configuration settings of the local CVP server (both WebApp or BackEnd).

This information is useful for registering a CVP BackEnd server in the Web Application.

Example:

```
[root@ios:/opt/midas30] midas.sh backend
[...]
     [echo] Server ios_server: OVO 8.2 HP-UX Itanium
     [echo]   Server Identifier: ios_server
     [echo]   Hostname: ios
     [echo]   Protocol: http
     [echo]   Port: 9661
     [echo]   Secure Communication: true
     [echo]   Platform: unix
     [echo]   Install Directory: /opt/midas30
     [echo]   Services:
     [echo]   backend
     [echo]     usermgmt
     [echo]     file
     [echo]       FTP Port: 10021
     [echo]       FTP Enabled: true
     [echo]       SSH Enabled: false
     [echo]     doc
     [echo]       Paper Format: A4
     [echo]     vcs
     [echo]     lock
     [echo]     ovoconfig
     [echo]       OVO Version: A.08.20.050
     [echo]       OVO Codeset: iso-8859-15
     [echo]     opccfg
     [echo]       OVO Version: A.08.20.050
     [echo]     ovotask
     [echo]       OVO Version: A.08.20.050
```

In this, example, the CVP server has the following key attributes:

Server Identifier:	ios_server
Host name:	ios
Protocol used (http/https):	http (https, if *secure communication* is true)
Server Port (TCP/IP):	9661

For displaying extended configuration information use the ***config*** sub-command.

8.2.2 Creating and restoring backups

The ***backup*** and ***restore*** sub-commands can be used to backup and restore the complete CVP configuration on a local server. The data is copied into a *.zip file, which is stored in the `<MIDAS_HOME>` directory.

Particularly this includes:

1. the XML DB containing CVP users, groups and roles
2. the default CVS repository in `<MIDAS_HOME>/data/repository`
3. all configuration files in `<MIDAS_HOME>/conf`

The following example illustrates the creation of a backup:

```
[root@ios:/opt/midas30] midas.sh backup
[...]
backup:
    [mkdir] Created dir: /opt/midas30/work/backup_200612201058
     [echo] backing up MIDAS configuration to /opt/midas30/work/backup_200612201058

intern.backup_xmldb:
     [echo] saving XML DB
    [mkdir] Created dir: /opt/midas30/work/backup_200612201058/xmldb
[xdb:backup] Creating backup of collection: xmldb:exist://ios.bes-
intern.com:9662/exist/xmlrpc/db
[xdb:backup] Backup directory: /opt/midas30/work/backup_200612201058/xmldb
[xdb:backup] writing
/opt/midas30/work/backup_200612201058/xmldb/db/backends/backends.xml
[xdb:backup] writing /opt/midas30/work/backup_200612201058/xmldb/db/rss/news.rss
[xdb:backup] writing /opt/midas30/work/backup_200612201058/xmldb/db/usermgmt/roles.xml
[xdb:backup] writing
/opt/midas30/work/backup_200612201058/xmldb/db/usermgmt/usergroups.xml
[xdb:backup] writing /opt/midas30/work/backup_200612201058/xmldb/db/usermgmt/users.xml
[xdb:backup] writing /opt/midas30/work/backup_200612201058/xmldb/db/system/users.xml

intern.backup_cvs:
     [echo] saving CS repository /opt/midas30/data/repository
    [mkdir] Created dir: /opt/midas30/work/backup_200612201058/cvs
     [copy] Copying 27 files to /opt/midas30/work/backup_200612201058/cvs
     [copy] Copied 2 empty directories to 1 empty directory under
/opt/midas30/work/backup_200612201058/cvs
```

```
intern.backup_conf:
    [mkdir] Created dir: /opt/midas30/work/backup_200612201058/conf
     [copy] Copying 185 files to /opt/midas30/work/backup_200612201058/conf
      [zip] Building zip: /opt/midas30/backup_20061220_1058.zip
   [delete] Deleting directory /opt/midas30/work/backup_200612201058
     [echo] backup archived in /opt/midas30/backup_20061220_1058.zip

BUILD SUCCESSFUL
Total time: 8 seconds
```

To restore a backup use the restore sub-command with the path of a backup ZIP file created with an earlier backup as shown in the following example:

```
[root@ios:/opt/midas30] midas.sh restore /opt/midas30/backup_20061220_1058.zip
[...]
restore:
    [mkdir] Created dir: /opt/midas30/work/restore_200612201104
     [echo] restoring backup backup_20061220_1058.zip
    [unzip] Expanding: /opt/midas30/backup_20061220_1058.zip into
/opt/midas30/work/restore_200612201104

intern.restore_xmldb:
     [echo] restoring backends

[xdb:restore] Restoring from /opt/midas30/work/restore_200612201104/xmldb/db/backends
[xdb:restore] restoring
file:/opt/midas30/work/restore_200612201104/xmldb/db/backends/__contents__.xml
[xdb:restore] Restoring backends.xml
     [echo] restoring user model
[xdb:restore] Database driver already registered.
[xdb:restore] Restoring from /opt/midas30/work/restore_200612201104/xmldb/db/usermgmt
[xdb:restore] restoring
file:/opt/midas30/work/restore_200612201104/xmldb/db/usermgmt/__contents__.xml
[xdb:restore] Restoring roles.xml
[xdb:restore] Restoring usergroups.xml
[xdb:restore] Restoring users.xml

intern.restore_cvs:
     [echo] restoring CVS repository to /opt/midas30/data/repository
     [copy] Copying 27 files to /opt/midas30/data/repository

intern.restore_conf:
     [echo] restoring configuration to /opt/midas30/conf
     [copy] Copying 27 files to /opt/midas30/conf
     [copy] Copied 3 empty directories to 1 empty directory under /opt/midas30/conf
   [delete] Deleting directory /opt/midas30/work/restore_200612201104
     [echo] restore successful

BUILD SUCCESSFUL
Total time: 6 seconds
```

During both activities the CVP server must be running (otherwise the XML DB cannot be accessed). After restoring a backup, it is recommended to restart the CVP server, though.

8.2.3 Displaying configuration

The ***config*** sub-command shows extensive information about the server settings and configuration of all installed adapters and components. It can be used to find out the settings specified during installation such as ports, hostnames etc. of all deployed CVP components:

Example:

```
[root@ios:/opt/midas30] midas.sh config
[...]
config.server:
     [echo] Server configuration
     [echo]   Backend name: ios_server
     [echo]   Installation directory: /opt/midas30
     [echo]   Version:      3.0.0
     [echo]   JMX port:     9660
     [echo]   HTTP port:    9662

config.server.file:
     [echo]   FTP configuration
     [echo]   FTP host: ios
     [echo]   FTP port: 10021
     [echo]   FTP enabled: true
     [echo]   License holder: blue elephant systems GmbH

config.server.ovoconfig:
     [echo] OVO/U adaptor configuration
     [echo]   Oracle JDBC URL: jdbc:oracle:thin:@ios:1521:openview
     [echo]   License holder: blue elephant systems GmbH

[...]
```

For basic configuration information use the ***backend*** sub-command.

8.2.4 Installing licenses

The ***install*** sub-command can be used to install CVP license files. This operation needs to be performed on each CVP server.

Simply copy the correct license file to the CVP installation folder or any other location (e.g. the `/tmp` directory) and run, e.g.:

```
# <MIDAS_HOME>/midas.sh install /tmp/licensefile.zip
```

After that, restart the CVP server:

```
# <MIDAS_HOME>/midas.sh restart
```

8.2.5 Displaying licenses

This sub-command lists all installed CVP licenses on the local system.

Example (shows a system with a full CVP installation and instant-on licenses):

```
# midas.sh licenses
[...]
licenses:
     [echo] installed MIDAS licenses
  [license] License 1161854473571:
  [license] Issued to:
  [license]   Name:       blue elephant systems GmbH (BES)
  [license] Signed by:
  [license]   Name:       blue elephant systems GmbH (BES)
  [license]   At:         2006/10/26 11:21:13
  [license] Component vendor:
  [license]   Name:       blue elephant systems GmbH (BES)
  [license] Component:
  [license]   Name:       Management Instrumentation Documentation and
                          Automation System Backend Adaptor (MIDAS_BACKEND)
  [license]   Version:    3.0.0
  [license]   Expiration: 2007/01/05 17:02:27
  [license]   NodeLock:   any
  [license]   Count:      1
  [license]   Level:      Instant-On
  [license] Status: Initialized
  [license]   in use:     0

  [license] License 1161854476811:
  [license] Issued to:

[...]
```

The licenses listed are those of

4. the CVP BackEnd server
5. the CVP web application
6. the CVP version control adapter
7. the CVP documentation adapter

depending on the role of the CVP server (WebApp or BackEnd) and the deployed components.

8.2.6 Install and remove patches

The ***patch*** sub-command has to be used to install future CVP fixpacks (patches), which will be delivered as *.zip packages.

To install such a fixpack file copy the ZIP file either into the CVP installation directory `<MIDAS_HOME>` or any other location (e.g. `/tmp`). Then, install the fixpack by executing the `midas.sh patch` command as shown in the following example:

```
# <MIDAS_HOME>/midas.sh patch /tmp/update.zip

...
```

The ***unpatch*** sub-command removes a previously installed fixpack. You must specify the ID of the fixpack to be de-installed as shown in the following example:

```
# <MIDAS_HOME>/midas.sh unpatch 3.0.0-midas-fp-0001
```

8.2.7 Starting, stopping and re-starting

The ***stop*** sub-command stops the complete CVP server on the local system, including both the BackEnd and WebApp depending on what is installed.

This command waits until the stop operation has completed.

Example:

```
# <MIDAS_HOME>/midas.sh stop
Stopping CVP Server...
Waiting for CVP Server to exit...
Waiting for CVP Server to exit...
Waiting for CVP Server to exit...
Stopped CVP Server.
#
```

The ***start*** sub-command starts CVP on the local system, whether the WebApp or some OVO BackEnd is installed (or both).

IMPORTANT: The start command returns immediately, however the CVP are still starting up. Depending on the speed of the local system, the start-up may take a while.

Example:

```
# <MIDAS_HOME>/midas.sh start
Starting CVP Server...
#
```

To check the start-up progress, view the file wrapper.log until the contents looks similar to the following and no more activities appear:

```
# tail -f <MIDAS_HOME>/logs/wrapper.log
[...]
INFO | jvm 1 | 2006/11/06 17:05:05 | User Mgmt Dispatcher ready.
INFO | jvm 1 | 2006/11/06 17:05:05 | Virtual Backend Request Transformer ready
INFO | jvm 1 | 2006/11/06 17:05:05 | User Model Request Transformer ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | User Management Filter ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Lock Releaser ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Authentication Transformer ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Lock Dispatcher ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Lock Transformer ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Auto Checkin Transformer ready.
INFO | jvm 1 | 2006/11/06 17:05:06 | Virtual Backend Request Transformer ready
```

The ***restart*** sub-command performs both operations in one step.

8.2.8 Displaying server status

The sub-command ***status*** displays the status of the CVP processes including connection status, license status and whether the server assembly is correct or not.

Example:

```
# <MIDAS_HOME>/midas.sh status
[...]
     [echo]   Server: ios_server
     [echo]   Service: ovotask
     [echo]     Name: User Model Server
     [echo]     Status: connected to XML DB
     [echo]     Error count: 0
     [echo]     Status: Component has a license
     [echo]     License held: 1161854473571
     [echo]
     [echo]     Name: File Server
     [echo]     Status: connected
     [echo]     Error count: 0
     [echo]     Status: Component has a license
     [echo]     License held: 1161854473571
     [echo]
     [echo]     Name: Task Server
     [echo]     Status: connected
     [echo]     Error count: 0
     [echo]     Status: Component has a license
     [echo]     License held: 1161854473571
     [echo]
     [echo]     Name: VCS Server
     [echo]     Status: Connected to CVS repository
[...]
```

If something is not correct, an according error is displayed.

In addition you can use the `ovstatus` command if the `ovstart/ovstop` integration has been selected during installation:

```
# /opt/OV/bin/ovstatus -c midas
 Name              PID  State     Last Message(s)
 midas               -  unknown   (Does not communicate with ovspmd.)
```

The statement *unknown* is normal since there is no tight integration with ovspmd.

8.2.9 Displaying product version

This command shows the current CVP version.

Example:

```
# <MIDAS_HOME>/midas.sh version
[...]
version:
     [echo] CVP version = 3.0.0
     [echo] CVP build number = 1
     [echo] CVP installation directory = /opt/midas30
```

8.3 Advanced tasks

The following tasks can be used to deal with data stored in CVP.

8.3.1 Importing OVO download data

The ***import*** command is to be used to import an OVO configuration download directory into the CVP clipboard directory. From there it can be processed further using regular GUI functionality.

Example:

```
# midas.sh import /tmp/my_download

...
```

8.3.2 Generating document blocks

Whenever, any changes were made to the documentation generation configuration file, run this command. Otherwise the changes have no effect. It re-generates a sitemap that is used during the documentation generation process.

The documentation generation configuration file referred to above is:

```
<MIDAS_HOME>/webapps/midas/work/webapp/content/doc/gendoc.xml
```

8.3.3 Cleaning up runtime data

This command cleans up runtime data on the CVP server, especially the work directory and log files.

IMPORTANT: Only execute this command when the CVP server has been stopped and when instructed to do so by product support.

Example:

```
[root@ios:/opt/midas30] midas.sh clean
[...]
clean:
   [delete] Deleting 570 files from /opt/midas30
   [delete] Deleted 162 directories from /opt/midas30
```

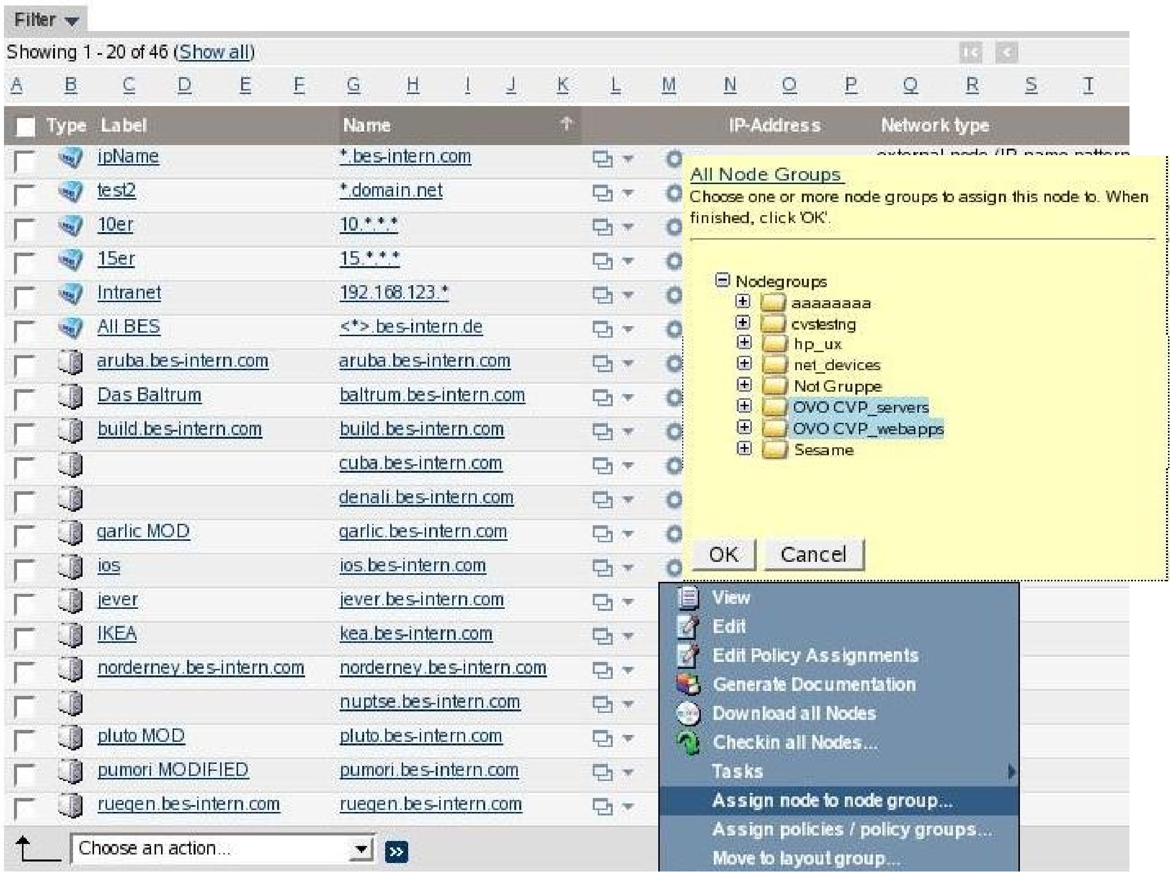
8.4 OVO Integration

8.4.1 Self-Monitoring

CVP comes with a set of policies, applications as well as node groups, profiles etc. intended for self-monitoring.

To use them in an OVO environment, perform the following steps:

6. Assign all CVP WebApp and BackEnd servers into the corresponding NodeGroups (in the following example, *ios* is both WebApp and BackEnd):



Filter

Showing 1 - 20 of 46 (Show all)

A B C D E F G H I J K L M N O P Q R S T

Type	Label	Name	IP-Address	Network type
	ipName	*.bes-intern.com		external node (IP name pattern
	test2	*.domain.net		
	10er	10.*.*.*		
	15er	15.*.*.*		
	Intranet	192.168.123.*		
	All BES	<*>.bes-intern.de		
	aruba.bes-intern.com	aruba.bes-intern.com		
	Das Baltrum	baltrum.bes-intern.com		
	build.bes-intern.com	build.bes-intern.com		
		cuba.bes-intern.com		
		denali.bes-intern.com		
	garlic MOD	garlic.bes-intern.com		
	ios	ios.bes-intern.com		
	jever	jever.bes-intern.com		
	IKEA	kea.bes-intern.com		
	norderney.bes-intern.com	norderney.bes-intern.com		
		nuptse.bes-intern.com		
	pluto MOD	pluto.bes-intern.com		
	pumori MODIFIED	pumori.bes-intern.com		
	ruegen.bes-intern.com	ruegen.bes-intern.com		

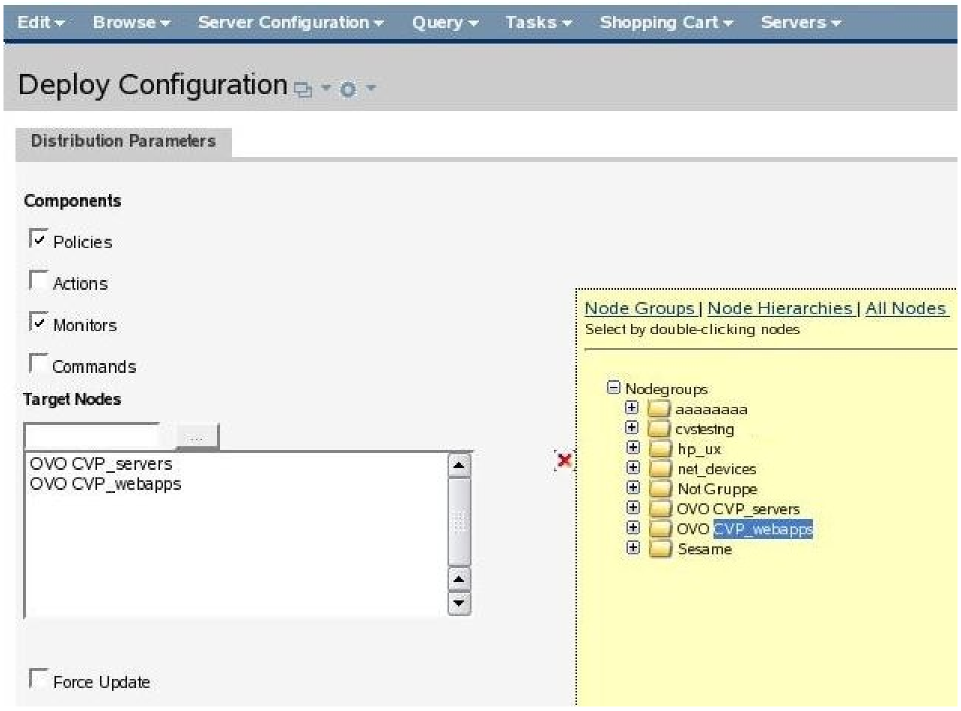
Choose an action...

All Node Groups
Choose one or more node groups to assign this node to. When finished, click 'OK'.

Nodegroups
- aaaaaaaa
- cvstestng
- hp_ux
- net_devices
- Not Gruppe
- OVO CVP_servers
- OVO CVP_webapps
- Sesame

OK Cancel

7. This will automatically assign the applicable policies. Deploy them (including *monitors* since there are also some scripts!):


Edit Browse Server Configuration Query Tasks Shopping Cart Servers

Deploy Configuration

Distribution Parameters

Components

[✓] Policies

[ ] Actions

[✓] Monitors

[ ] Commands

Target Nodes

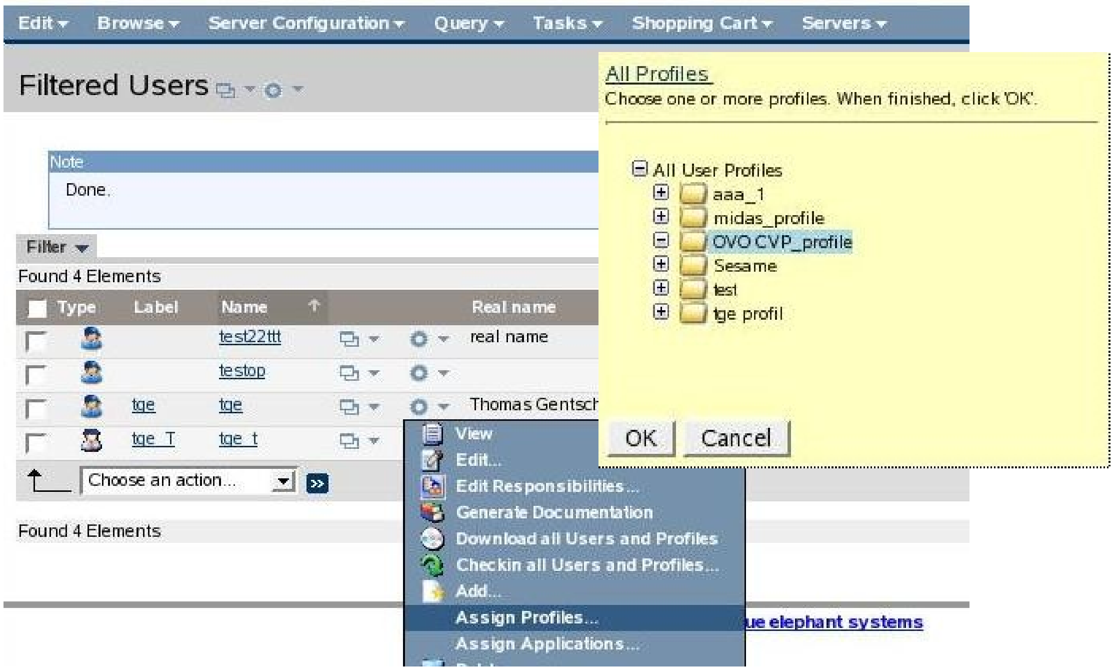
OVO CVP_servers
OVO CVP_webapps

[ ] Force Update

Node Groups | Node Hierarchies | All Nodes
Select by double-clicking nodes

Nodegroups
- aaaaaaaa
- cvstestng
- hp_ux
- net_devices
- Not Gruppe
- OVO CVP_servers
- OVO CVP_webapps
- Sesame

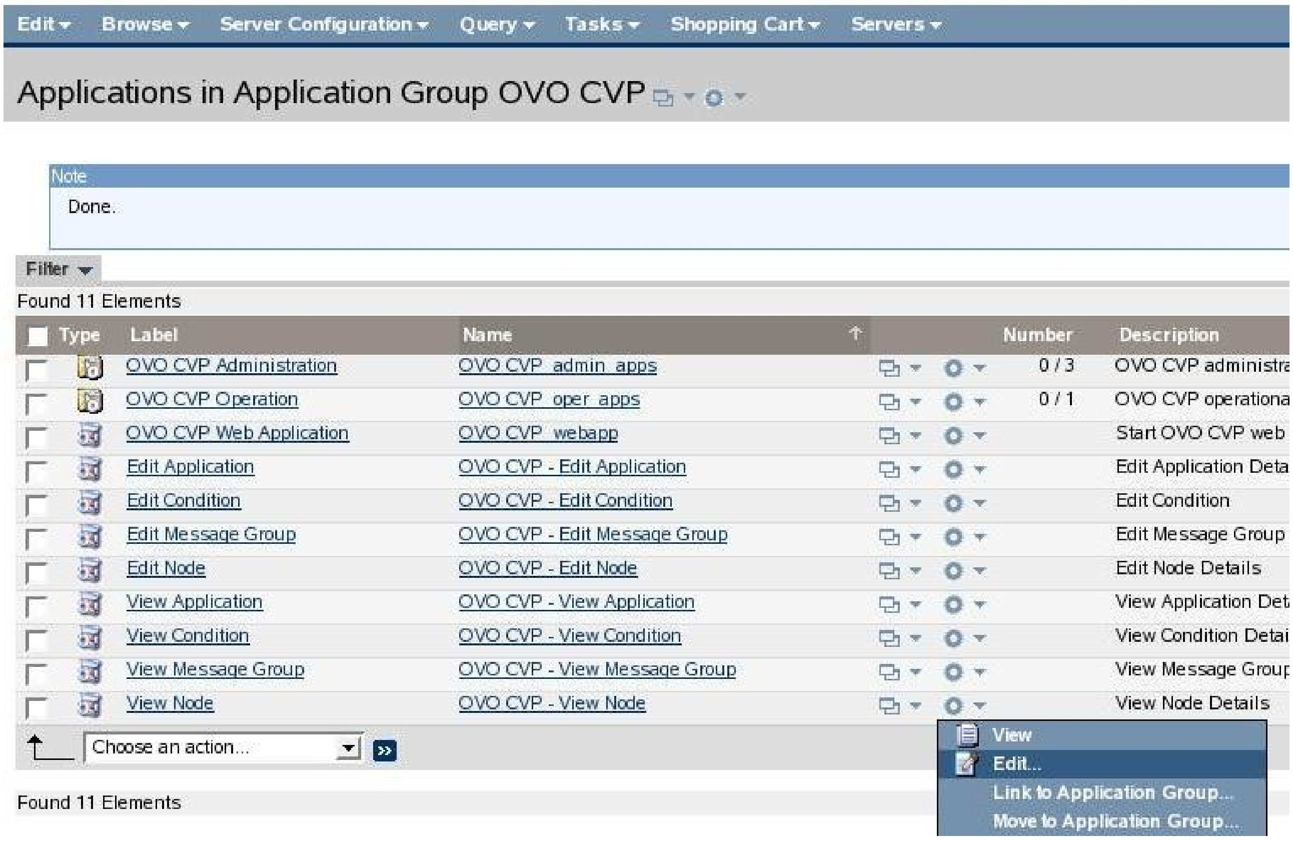
8. To make the alarm messages available to OVO operators, assign the profile *CVP_profile* to these users:



No other steps are necessary.

All objects can be customized as needed, however, normally there should be no need to to so except for the integrated Applications.

There are some OVO Application elements which refer to the WebApp host name and port, e.g. to launch the CVP GUI from the OVO desktop. Initially these will be set to the value entered during installation (in case of a full installation this will be the local WebApp). To change these URLs, just update the according OVO Application elements:



Applications in Application Group OVO CVP

Note

Done.

Found 11 Elements

Type	Label	Name	Number	Description
	OVO CVP Administration	OVO CVP admin apps	0 / 3	OVO CVP administra
	OVO CVP Operation	OVO CVP oper apps	0 / 1	OVO CVP operationa
	OVO CVP Web Application	OVO CVP webapp		Start OVO CVP web
	Edit Application	OVO CVP - Edit Application		Edit Application Deta
	Edit Condition	OVO CVP - Edit Condition		Edit Condition
	Edit Message Group	OVO CVP - Edit Message Group		Edit Message Group
	Edit Node	OVO CVP - Edit Node		Edit Node Details
	View Application	OVO CVP - View Application		View Application Det
	View Condition	OVO CVP - View Condition		View Condition Detai
	View Message Group	OVO CVP - View Message Group		View Message Group
	View Node	OVO CVP - View Node		View Node Details

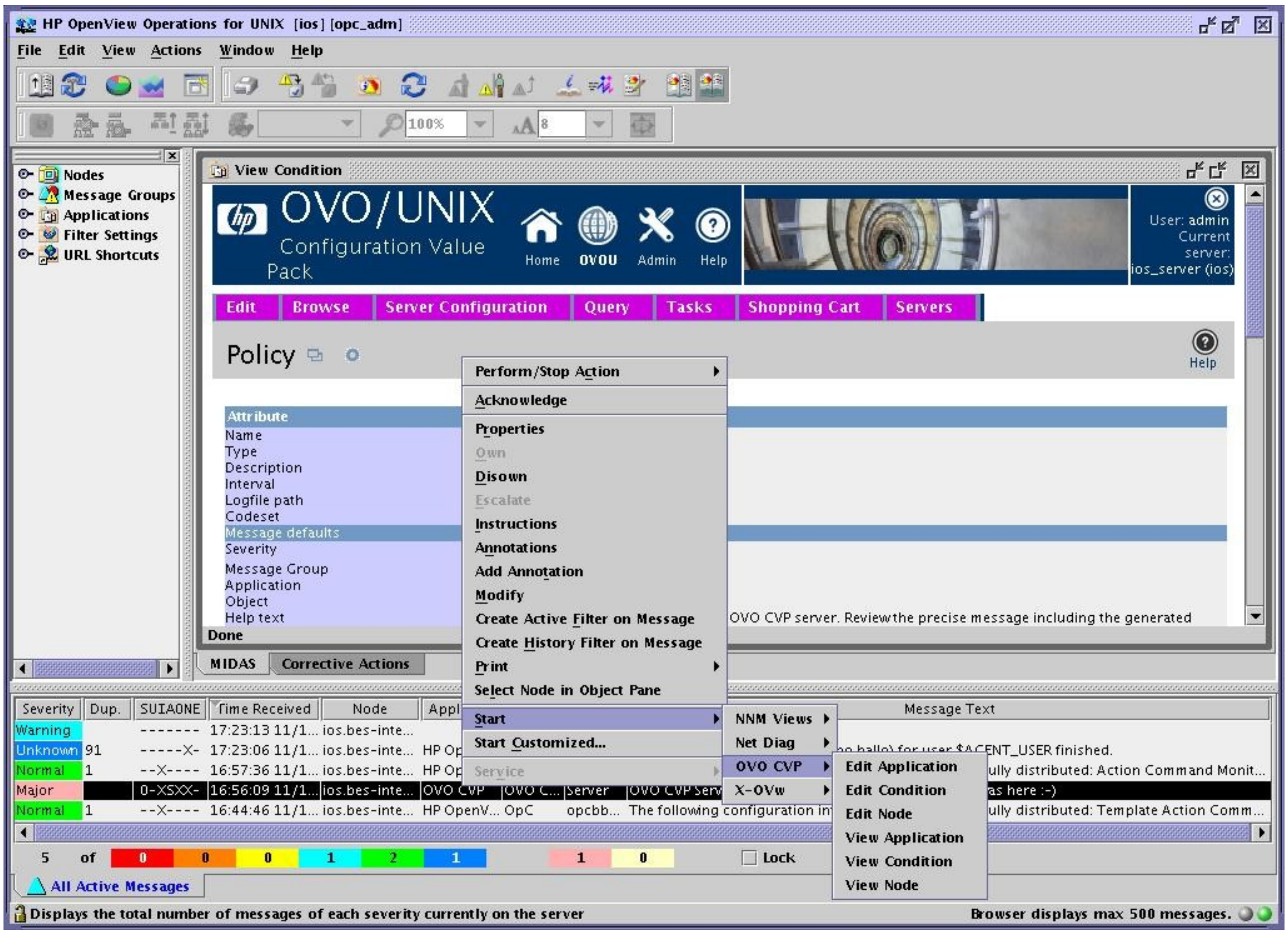
Found 11 Elements

Re-load the OVO GUI sessions to activate the changes.

8.4.2 OVO Java GUI

In addition to the classic OVO SMART-PlugIn capabilities like starting, stopping the server status or launching a GUI, CVP provides a set of OVO Applications, which can be used most efficient in the OVO Java GUI. They all launch the CVP GUI as HTML page inside the Java GUI and are able to directly jump to the desired context, e.g. a policy condition, which has caused a message or the node where the message originated.

For example, view the policy condition which has caused the message:



OVO CVP server. Review the precise message including the generated

```
Severity  Dup.  SUIAONE  Time Received     Node           Appl               Message Text
Warning         -------  17:23:13 11/1...  ios.bes-inte...
Unknown   91    -----X-  17:23:06 11/1...  ios.bes-inte...  HP Op   ...no hallo) for user $AGENT_USER finished.
Normal    1     --X----  16:57:36 11/1...  ios.bes-inte...  HP Op   ...ully distributed: Action Command Monit...
Major           0-XSXX-  16:56:09 11/1...  ios.bes-inte...  OVO CVP  OVO C...  Server  OVO CVP Serv...  ...as here :-)
Normal    1     --X----  16:44:46 11/1...  ios.bes-inte...  HP OpenV... OpC  opcbb...  The following configuration in... ...ully distributed: Template Action Comm...

5 of  0  0  0  1  2  1    1  0
```

There is no set-up needed for this, just make sure that the OVO users have the CVP_profile profile assigned.

9 CVP Configuration

This chapter contains information for tuning the CVP environment. Normally, setup attempts to discover values for all relevant parameters during the installation, and there should be no need to change any configuration. However, if you make any changes to the environment after installation, you will have to make sure that the changes do not have an adverse impact on CVP. For example, if you change the password for the OVO administrator, opc_adm, you will have to inform CVP , too.

9.1 Configuring OVO and Oracle access

9.1.1 Oracle connectivity

The Oracle connection information is stored in the `ovoconfig.properties` file as illustrated in the following example:

```
# cat server/conf/ovoconfig.properties
[...]
ovodb.url=jdbc:oracle:thin:@ios:1521:openview
ovoapi.password=411BCA8089C84376
ovoapi.codeset=iso-8859-15
ovodb.user=opc_op
ovodb.password=FA16D5346CA25CA0
ovoconfig.readOnly=false
ovoconfig.codeset=iso-8859-15
licenses.holderName=blue elephant systems GmbH
```

Verify that the `ovodb.url` property is correctly configured, for example, by making sure that the hostname, port number, and database name are all correctly specified in the JDBC connection string.

Change this setting if the Oracle listener is configured to listen for connections on another port, if Oracle is running on another server, or you make any other customization. If any of the properties explained here have been changed, you must restart the CVP server.

The `ovodb.url` connection string is structured as following:

```
jdbc:oracle:<thin|oci>:@<service|host:port:servicename>
```

This connection string consists of the connection type `thin` or `oci` and the actual target after the @ character.
To access a database via SQL*Net, use the *host:port:servicename* target definition. In this case, *host* and *port* have to be set to the values where the Oracle listener process for the DB instance *servicename* can be reached. Default port is 1521 and with a default OVO set-up, the *servicename* is “openview”. Alternatively, specify a *service* as target, which must equal the DB service name as registered in the `tnsnames.ora` file. With a default OVO set-up the *service* name is “ov_net”.
For secure Oracle connections use connection type `oci`, otherwise `thin`.

The `ovodb.user` determines the Oracle user account to be used to log on to the DB. Usually this `opc_op` or `opc_report`. The user `opc_report` has read-only capabilities by default. This is sufficient in almost all cases except:

- Deleting OVO instruction interfaces
- Deleting OVO notification services

These two operations are implemented by directly writing the Oracle DB and will fail with an `ovodb.user` without writing capabilities.

For more information regarding Oracle JDBC as used by CVP please refer to http://www.oracle.com/technology/tech/java/sqlj_jdbc/htdocs/jdbc_faq.htm.

9.1.2 Users and passwords

CVP accesses OVO via the OVO Configuration API and directly (read-only) the OVO Oracle database instance. For both purposes, a dedicated user account and password is needed which have to be maintained in CVP.

Both will be queried during installation and stored in the CVP configuration as shown above. Thus, normally there is no need to change anything unless either value changes in OVO or Oracle itself – then also CVP has to be updated.

9.1.2.1 Encrypting and configuring passwords

For security reasons, all passwords stored in CVP are encrypted. To encrypt a password (e.g. of the OVO user opc_adm or the Oracle user opc_op), use the tool password.sh:

```
# <MIDAS_HOME>/bin/password.sh [-a|-d] [-u] [-c] [<passwd>]
```

You can use this tool with a pop-up GUI window or via the command line.

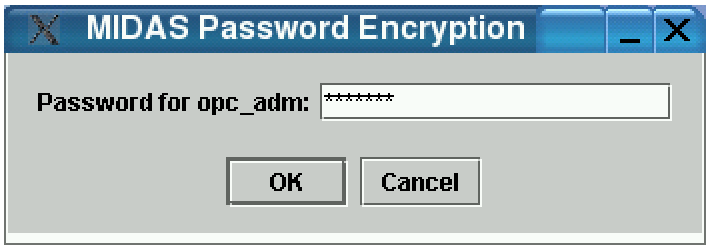
The recommended mode is to start it with the option -u (--update) – then the tool will automatically update the corresponding configuration file. In that case, it is mandatory to specify either -a (--admin) or -d (--database) to select which password (OVO Administrator opc_adm or Oracle DB user opc_op) is to be encrypted and updated.

In GUI mode the tool opens a small dialog and asks for the password (which it does not display on screen) and outputs the encrypted string.

To use the password tool in graphical-mode, run the following command:

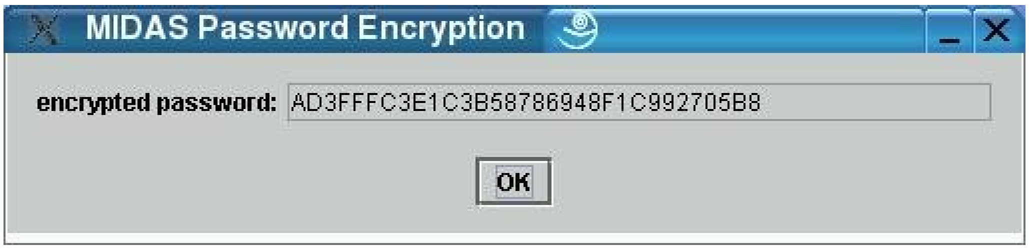
```
# <MIDAS_HOME>/bin/password.sh [-a|-d] [-u]
```

When the password-encryption window prompts you for the existing password, type the plain-text password in to the password box, as illustrated in the following figure:


MIDAS Password Encryption

Password for opc_adm: *******

If the -u (--update) option is *not* specified, the password-encryption tool displays the encrypted version of the password in a new window, as illustrated in the following figure:


MIDAS Password Encryption

encrypted password: AD3FFFC3E1C3B58786948F1C992705B8

You must manually copy the displayed encrypted string and paste it into the appropriate place in the ovoconfig.properties file. In this example, the encrypted password is for the OVO administrator opc_adm, so you must paste the copied string into the property ovoapi.password, as illustrated in the following example:

```
# vi <MIDAS_HOME>/conf/ovoconfig.properties
[...]
ovoapi.password=AD3FFFC3E1C3B58786948F1C992705B8
[...]
```

If you want to run the password-encryption on the command line, use the -c option, as illustrated in the following example:

```
# <MIDAS_HOME>/bin/password.sh [-a|-d] -c [-p <plain passwd>]
```

In command-line mode, the password-encryption tool prints the encrypted password to `stdout`. You must copy the encrypted password and paste it into the appropriate property in the `ovoconfig.properties` file.

NOTE: It is common to forget to update the CVP configuration after changing the OVO password. If the CVP server does not start but Oracle and OVO are running fine, check the password settings first. For more information about typical error codes for problems relating to incorrect passwords, see 9.2 Wrong or unknown OVO passwords.

9.2 Wrong or unknown OVO passwords

If CVP does not start or work correctly, a common problem is that the OVO-access parameters (particularly passwords) are incorrect. For example, when installing CVP, one of the required password was either unknown or entered incorrectly. Alternatively, a password was changed in OVO or Oracle at some point after the installation completed but the changed password was not updated in CVP.

The information in the following sections describe how to solve problems relating to missing or incorrect passwords.

9.2.1 Oracle database access

9.2.1.1 Testing an Oracle password

If the required password is unknown, either ask someone who knows it or if you have some ideas what it could be, first test it using a standalone commands.

Determine the related Oracle parameters:

```
# cat /etc/opt/OV/share/conf/ovdbconf
DB_VENDOR Oracle
DB_NAME openview
DB_RELEASE 10.1.0
DB_TIME_STAMP "Tue Aug 1 14:50:20 METDST 2006"
DB_USER ovdb
ORACLE_SID openview
ORACLE_HOME /opt/oracle/product/10.1.0
ORACLE_BASE /opt/oracle
DBA_USER oracle
DATA_DIR /opt/u01/oradata/openview
CREATE_DIR /opt/oracle/admin/openview/create
INDEX_DIR /opt/u01/oradata/openview
ADMIN_DIR /opt/oracle
OS_AUTHENT_PREFIX
CHARACTER_SET WE8ISO8859P15
BASE_DATA_TS_SIZE 25
BASE_INDEX_TS_SIZE 5
DATA_TS_SIZE 25
INDEX_TS_SIZE
```

```
TEMP_TS_SIZE 2
DATA_TS_EXTENT_SIZE 2
DATA_TS_MAX_SIZE 500
INDEX_TS_EXTENT_SIZE
ECHO_CMD echo
PROMPT TRUE
DBA_PROGRAM sqlplus
OV_USER ovdb
DBA_LOGFILE /var/opt/OV/share/log/sqlplus_log
ORACLE_BASE_REV 10
ORACLE_SECOND_REV 1
NLS_LANG american_america.WE8ISO8859P15
ITO_DATADIR /opt/u01/oradata/openview
ITO_INDEXDIR /opt/u01/oradata/openview
SQLNET_ALIAS ov_net
```

Switch to the user `oracle` and try to connect to the database using the user-password combination you want to test. If the OVO database instance runs under a different user account, you will have to check the `ovdbconf` file for the appropriate values for the user name and password. The example above shows an excerpt from a `ovdbconf` file.

```
# su – oracle -c sqlplus
SQL*Plus: Release 9.2.0.1.0 - Production on Tue Mar 29 16:42:16 2005
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

Enter user-name: opc_op
Enter password:
[...]
Connected.
SQL> exit
```

If the `sqlplus` command displays the response `Connected`, the user name and password you tested are correct; if not, you will have to try again.

CAUTION: DO NOT USE this command to login like this (or similar):

Enter user-name: opc_op as sysdba

Here you can enter any password even a wrong one and you can still login, so this is no real check.

This should work with all Oracle version 8.x through 10.x.

If you found out the password this way, update the CVP configuration as described above (Encrypting and configuring passwords).

9.2.1.2 Updating the Oracle password

If it is not possible to find out the Oracle password, you will have to change it.

CAUTION: You should change the Oracle password only if absolutely necessary. For more information about changing the password, see the opcdbpwd man page.

To change the Oracle password for the OpenView database, use the OVO command opcdbpwd as illustrated in the following example:

```
# /opt/OV/bin/OpC/opcdbpwd -set
```

You must use the opcdbpwd command to change the Oracle password for the OpenView database (rather than the Oracle SQL command alter). The opcdbpwd command also updates OVO's internal security file opcdbpwd.sec with the new authentication, which is essential for OVO to continue to work properly after the password change. If you use the opcdbpwd command to change the Oracle password, make sure you also update the CVP configuration as described in the section Encrypting and configuring passwords.

If you have changed the Oracle opc_op password, update the CVP configuration as described above (Encrypting and configuring passwords).

The Oracle opc_report password cannot be changed using opcdbpwd, instead use the appropriate sqlplus commands as shown in the following example:

```
SQL> alter user opc_report identified by <new password>;
SQL> commit;
```

9.2.2 OVO Admin access

The easiest way to verify the opc_adm password is to try to log in using the regular OVO GUI. Alternatively, you can test the password of the OVO Administrator user opc_adm using the commands shown in the following examples:

```
# vi /tmp/conn_test
H,1.0,Test User API (rc)
S,Connect,connects to database,OPC_ERR_OK,TERM
S,Disconnect,disconnects from database,OPC_ERR_OK,TERM

# /opt/OV/contrib/OpC/opccfgtest -f /tmp/conn_test -u opc_adm -p <pwd>
[...]
15:14:26.247 Test User API (rc) :    number of passed steps     : 2
[...]
```

If the password you test is correct, the OVO connection succeeds and the command displays the number of steps *passed.* If the connection test fails, the command displays a message indicating a fatal error, as illustrated in the following example.

```
# /opt/OV/contrib/OpC/opccfgtest -f /tmp/conn_test -u opc_adm -p <wrong
passwd>
[...]
15:14:34.590   call opc_connect
15:14:34.590    ---> -50            ... Fatal
15:14:34.622                       .... Exception Terminate occurs
```

If the password of the OVO Administrator user opc_adm is lost entirely, you will have to contact product support.

9.2.3 How to identify a password error

In case one of the passwords required by CVP is incorrect, the CVP server will not start. If an error occurs relating to missing or incorrect passwords, check the contents of the `<MIDAS_HOME>/logs/wrapper.log` file. If the password of the OVO Administrator opc_adm is wrong, the `wrapper.log` file reports one of the following errors:

```
INFO   | jvm 1    | 2005/08/26 17:22:43 |
org.apache.excalibur.containerkit.lifecycle.LifecycleException:
Component named "ovoapi" failed to pass through the Initialization
stage. (Reason: com.bes.ovo.api.common.impl.OVException: No login
(unknown message=-50)).
```

or:

```
INFO   | jvm 1    | 2005/10/06 18:45:01 |
org.apache.excalibur.containerkit.lifecycle.LifecycleException:
Component named "ovoapi" failed to pass through the Initialization
stage. (Reason: com.bes.ovo.comp.OVOException: Unable to connect to OVO
as administrator).
```

If the password of the Oracle DB user opc_op is wrong, you will find one of the following errors in the `wrapper.log` file:

```
ERROR  | jvm 1    | 2005/08/29 09:35:02 | java.sql.SQLException: ORA-
01017: invalid username/password; logon denied
INFO   | jvm 1    | 2005/08/29 09:35:02 |  at
oracle.jdbc.dbaccess.DBError.throwSqlException(DBError.java:169)
```

If no password was supplied for the OpenView database during the CVP installation, you will find the following entry in the `wrapper.log` file:

```
INFO   | jvm 1    | 2005/09/22 11:40:56 | java.sql.SQLException: Null
user or password not supported in THIN driver
INFO   | jvm 1    | 2005/09/22 11:40:56 |  at
oracle.jdbc.dbaccess.DBError.throwSqlException(DBError.java:169)
```

9.3 Logging configuration

Logging stands for all activities where CVP writes some status information into a dedicated log file. Normally these are errors only, but for troubleshooting scenarios, also tracing data can be obtained this way.

9.3.1 Log4j logging

Both logging and tracing are covered by the same logging mechanism ***log4j*** (for details see http://logging.apache.org).

General logging for the CVP **Server** (both WebApp and BackEnd) is configured in the following file (also containing more explanations about the syntax, meaning of values etc.):

```
# cat <MIDAS_HOME>/conf/log4j.xml

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
```

```
<!--
Log4J Configuration Quick Reference:
====================================
Priority order is DEBUG < INFO < WARN < ERROR < FATAL
PatternLayout conversion characters:
%c   Category of the logging event
%C   Fully qualified class name of the caller
[...]
Examples:  "%r [%t] %-5p %c %x - %m\n"
"%-6r [%15.15t] %-5p %30.30c %x - %m\n"
-->
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/"
debug="false">
  <appender name="console" class="org.apache.log4j.ConsoleAppender">
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%p - %C{1}.%M(%L) |
%m%n"/>
    </layout>
  </appender>
  <!-- MIDAS adaptor default log file -->
  <appender name="midas" class="org.apache.log4j.RollingFileAppender">
    <param name="File" value="logs/midas.log"/>
    <param name="MaxFileSize" value="10MB" />
    <param name="MaxBackupIndex" value="10" />
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%p - %d | %C{1}.%M(%L) |
%m%n"/>
    </layout>
  </appender>
[...]
  <logger name="com.bes.itm.comp.servicemix" additivity="false">
    <level value="DEBUG"/>
    <appender-ref ref="midas"/>
  </logger>
[...]
  <!-- default -->
  <root>
    <level value="INFO"/>
    <appender-ref ref="midas"/>
  </root>
</log4j:configuration>
```

An <appender> tag describes the actual log target (e.g. name of log file, entry format, rolling behavior, ...). A <logger> tag defines which appender to use with which log level for a CVP component.

If you want to configure the log level, change the appropriate <logger> tag; to change the log file behavior itself, change the <appender> tag.

You can choose between any of the following log levels:

Log level	Description
DEBUG	Most detailed level of logged information.
INFO	Detailed logging level to report minor and major error conditions, warnings, as well as correct system behavior.
WARN	Detailed logging of all unusual warning and error conditions.
ERROR	Level to report only error conditions. Warnings and correct system behavior is not logged.
FATAL	Only critical system failures are logged.

CAUTION: The log level to NONE disables error logging completely. It is strongly recommended to set the log level at least to WARN; ideally, you should set the logging level to INFO.

9.3.2 WebApp logging

To configure additional logging of the CVP Web Application, use the `logkit.xconf` file

```
# cat <MIDAS_HOME>/webapps/midas/work/webapp/WEB-INF/logkit.xconf
[...]
  <targets>
  [...]
    <!--
      This log file gets only messages with log level ERROR and below.
    -->
    <priority-filter id="error" log-level="ERROR">
      <cocoon>
        <filename>${context-root}/WEB-INF/logs/error.log</filename>
        <format type="cocoon">          %7.7{priority} %{time}   [%{category}]
(%{uri}) %{thread}/%{class:short}: %{message}\n%{throwable}
        </format>
        <append>false</append>
      </cocoon>
    </priority-filter>

    <cocoon id="debug">
      <filename>${context-root}/WEB-INF/logs/debug.log</filename>
      <format type="cocoon">          %7.7{priority} %{time}   [%{category}]
(%{uri}) %{thread}/%{class:short}: %{message}\n%{throwable}
      </format>
      <append>false</append>
    </cocoon>
[...]
```

The WebApp actually uses Apache logkit (http://excalibur.apache.org/logger.html), but the settings are very similar as shown in the example above.

9.4 System Startup/Shutdown

You can configure the automatic start-up and shutdown of the CVP server processes during installation of CVP. The options you can choose for automatic startup are as follows:

- Integrate with OpenView startup (ovstart/ovstop)
- Integrate with the operating-system boot process
- Manually start the CVP server

9.4.1 ovstart/ovstop integration

To change the behavior of the `ovstart/ovstop` integration:

1. Update the LRF file `/etc/opt/OV/share/lrf/midas_ovo.lrf`
2. Run the `ovaddobj` command with the updated LRF file:

```
# ovaddobj /etc/opt/OV/share/lrf/midas_ovo.lrf
```

To remove the CVP integration with `ovstart/ovstop`, use the `ovdelobj` command:

```
# ovdelobj /etc/opt/OV/share/lrf/midas_ovo.lrf
```

For further details, see the OV product documentation.

9.4.2 OS boot integration on UNIX systems

If selected during CVP installation, the CVP server can be started/stopped along with the regular OS boot sequence. Depending on the platform there is a script:

- `/etc/init.d/midas_server` (Linux or Solaris)
- `/sbin/init.d/midas_server` (HP-UX)

to start the CVP server. All startup scripts support the standard parameters `start_msg`, `stop_msg`, `start` and `stop`.

Additionally, run-level links are created referring to the `midas_server` script. Depending on the platform these are in:

- `/etc/init.d/rc3.d` (Linux)
- `/etc/rc3.d` (Solaris)
- `/sbin/rc3.d` (HP-UX)

for the start-up link and in

- `/etc/init.d/rc2.d` (Linux)
- `/etc/rc2.d` (Solaris)
- `/sbin/rc2.d` (HP-UX)

for the shutdown links. Default numbers are 25 for startup links and 75 for shutdown links.

With the run-level links as above the CVP server is started in run-level 3 and stopped in run-level 2. The CVP server has to start after the Oracle database and preferably after the HP OVO server.

9.4.3 Startup / Shutdown on Windows systems

On Windows systems the CVP server is configured as Windows service in start mode ***automatically*** if automatic system startup has been selected in the installer and **manual** otherwise.

The CVP server can be started and stopped manually via the Windows service panel or from the command line with

```
# net start midas_server
# net stop midas_server
```

If a personal firewall is in place it may ask for permission to start `wrapper.exe` and `java.exe`.

9.5 Changing location of dynamic directories

It is possible to change the location of the log and runtime data directories, e.g. to /var/opt. Doing so ensures that a full file system does not lead to a server failure. At the same time this setup is in line with the OpenView OVO/U directory architecture.

In the following example we assume that these changes are done:

```
<MIDAS_HOME>/logs → /var/opt/midas/logs
<MIDAS_HOME>/data → /var/opt/midas/data
<MIDAS_HOME>/work → /var/opt/midas/work
```

NOTE: Stop MIDAS before applying any of these changes!

9.5.1 Changing the log directory

The log directory contains all log files created by all CVP components. To change the location, perform the following steps:

- `# mkdir /var/opt/midas/logs`
- Edit `<MIDAS_HOME>/conf/log4j.xml`
  Search for `<param name="File" value="`
  You have to edit multiple entries! Change each entry to the new file location, like:
  ```
  <param name="File" value="/var/opt/midas/logs/debug.log"/>
  ```

9.5.2 Changing the work directory

The work directory contains intermediate files created by CVP, usually temporarily only. To change the location, perform the following steps:

- `# mkdir /var/opt/midas/work`

- Edit `<MIDAS_HOME>/conf/servicemix.xml`
  Search for `<sm:container id="jbi"`
  Change the file location of the work directory:
  ```
  <sm:container id="jbi"
  rootDir="/var/opt/midas/work"
  ```

- Edit `<MIDAS_HOME>/conf/ant/config.xml`
  Change the properties of “`dir.work`”
  ```
  <property name="dir.work" value="/var/opt/midas/work"/>
  ```

- Edit all `<MIDAS_HOME>/conf/<adaptor>.properties` files
  TIP: use `# grep tempResource *.properties` to find these files easily
  Edit each file to `<component>.tempResource=file:/var/opt/midas/work`

9.5.3 Changing the data directory

The data directory contains data files stored by CVP persistently, for example the CVS repository of the clipboard exists here. Therefore it is necessary to move the contained data (unless not needed). Some parts (like the XML DB) absolutely have to be moved.

To change the location, perform the following steps:

- `# mkdir /var/opt/midas/data`

- move all data from `<MIDAS_HOME>/data to /var/opt/midas/data`

- Edit the CVS repository path in `<MIDAS_HOME>/conf/vcs.properties`
  Change this line to:
  ```
  cvs.cvsRoot=:local:/var/opt/midas/data/repository
  ```

- Edit `conf/ant/config.xml`
  and change these values for directory properties like
  ```
  <property name="dir.clipboard" value="/var/opt/midas/data/clipboard"/>
  <property name="dir.archive" value="/var/opt/midas/data/archive"/>
  <property name="dir.cvsrepo" value="/var/opt/midas/data/repository"/>
  ```

- Edit `<MIDAS_HOME>/conf/user.properties`
  change this line to:
  ```
  FtpServer.user.file.home=/var/opt/midas/data
  ```

- To change the location of the XML db located in `<MIDAS_HOME>/data/xmldb`
  edit `<MIDAS_HOME>/webapps/exist/work/webapp/WEB-INF/conf.xml`
  to:
  ```
  <db-connection database="native" files="/var/opt/midas/data/xmldb"
  pageSize="4096" cacheSize="96M" free_mem_min="5"
  collectionCacheSize="128">
  ```

- Edit `<MIDAS_HOME>/conf/tx.xml` change
  `logFileDir=”/var/opt/midas/data/txlog”`

- CVP can now be started again

9.6 *Running* CVP *in a HA cluster*

The basic concepts of using CVP in a HA cluster are described in chapter 5.7 Installing in a HA cluster.

The following specific configuration steps are possibly required.

If the CVP server running as HA package is communicating with other CVP servers (e.g. separated WebApp and BackEnd):

- Add on the peer CVP system in the file
  `<MIDAS_HOME>/conf/ip.properties`
  the virtual host name of the HA package running the CVP server and in addition all IP addresses of the possible physical cluster nodes.
  This file contains all host names and IP addresses which are permitted to access the local FTP server. If FTP is not being used, this is not needed.
- Make sure that a firewall possibly existing between both permits traffic for all possible IP addresses (virtual and physical)

9.7 Advanced communication topics

9.7.1 Changing default ports

The communication relationships and default ports used by CVP are explained in chapter 4.3 Communication and Ports and chapter . During installation most of these values can be specified. To update ports later, review the following steps.

NOTE: The ports used in the following CVP configuration files are the default ports. If different values have been specified during installation, apply them here as needed.

In all cases, the CVP has to be re-started.

9.7.1.1 HTTP(S) server ports

Each CVP server (both WebApp and BackEnd) opens HTTP(s) ports for internal communication between the CVP servers. This is normally configured during installation but can be changed later by updating:

```
# vi <MIDAS_HOME>/conf/backend_local.xml
[...]
<backend xmlns:do="http://www.openadaptor.org/" do:type="backend">
  <name do:type="String" xml:space="preserve">ios_server</name>
  <hostname do:type="String" xml:space="preserve">ios</hostname>
  <basedir do:type="String" xml:space="preserve">/opt/midas30</basedir>
  <description do:type="String" xml:space="preserve">ovo 8 full
  </description>
  <platform do:type="String" xml:space="preserve">unix</platform>
  <protocol do:type="String" xml:space="preserve">http</protocol>
  <secure do:type="Boolean">false</secure>
  <http do:type="String" xml:space="preserve">9661</http>
```

```
[...]
```

Modify the value of the <http> tag to change the port number.

By changing the <secure> tag, the server port can be switched between HTTP and HTTPS. There is no way to use both at the same time. Also, using HTTPS automatically enforces client certificates. For more details regarding HTTPS see chapter 9.7.2 Using HTTPS.

9.7.1.2 HTTP(S) WebApp ports

The CVP WebApp opens both an HTTP and HTTPS port which is the access point for GUI sessions.

This can be configured in the `jetty.properties` file as shown in the following example:

```
# vi <MIDAS_HOME>/conf/jetty.properties
[...]
# layer configuration file for jetty adaptor
jetty.connectors[0].port=9662
jetty.connectors[1].port=9663
```

The port referenced by `jetty.connectors[0].port` is the HTTP port, the other is the HTTPS port.

NOTE: With the current version, both are mandatory.

9.7.1.3 FTP and SSH communication

These topics are covered in chapters 9.9.1 Configuring FTP transfer and 7.3 Secure Shell (SSH).

9.7.1.4 Internal communication

This topic covers mainly the communication between the `wrapper` process and the actual JRE. The wrapper is a controller process integrated in the start/stop sequence which starts, stops, restarts and monitors the JRE process. The JRE performs all the actual CVP functions. For more details see 4.5 Processes and http://wrapper.tanukisoftware.org.

The JRE is started by the wrapper process as child process. The wrapper connects to the JRE using a local TCP socket.

The port will not be accessed over the network. To avoid port conflicts with other applications the ports can be configured by checking and updating the file `<MIDAS_HOME>/conf/wrapper.conf` as shown below.

A single explicit port can be enforced by setting the `...port` property or alternatively, a port range can be chosen with the `min` and `max` properties:

```
# vi <MIDAS_HOME>/conf/wrapper.conf
[...]
# Local communication between the wrapper process and the JRE.
# Server port opened by the JRE.
# wrapper.port=15012
# wrapper.port.min=32000
# wrapper.port.max=32999

# Local communication between the wrapper process and the JRE.
# Client port used by the wrapper process.
# wrapper.jvm.port=12345
# wrapper.jvm.port.min=31000
# wrapper.jvm.port.max=31999
[...]
```

If neither is specified, both port numbers are by default in the range as specified above (31xxx and 32xxx), the first unallocated port will be picked. Normally this should be sufficient and this should be changed only if needed.

9.7.1.5 JMX

For troubleshooting purposes each CVP server opens a port which can be used to connect with a JMX console. This port is not needed for anything else, but again, to avoid port conflicts with other applications the port range can be configured. To do this, check and update:

```
# vi <MIDAS_HOME>/conf/jmx.xml
[...]
  <sm:rmiRegistry id="rmiRegistry" port="9660" />
  <!-- JMX server -->
  <sm:jmxServer id="jmxServer" locateExistingServerIfPossible="true" />
  <!-- JMX Remote connector -->
  <sm:jmxConnector objectName="connector:name=rmi"
                   serviceUrl="service:jmx:rmi:///jndi/rmi://ios:9660/jmxrmi"
                   threaded="true"
                   daemon="true"
                   depends-on="rmiRegistry">
[...]
```

To actually use a JMX console on the local CVP server it has to be installed there (not part of CVP). This can be done also remotely – in this case make sure that the JMX port is reachable from the system where the JMX console is running (configure firewall rules if needed).

9.7.2 Using HTTPS

9.7.2.1 Overview

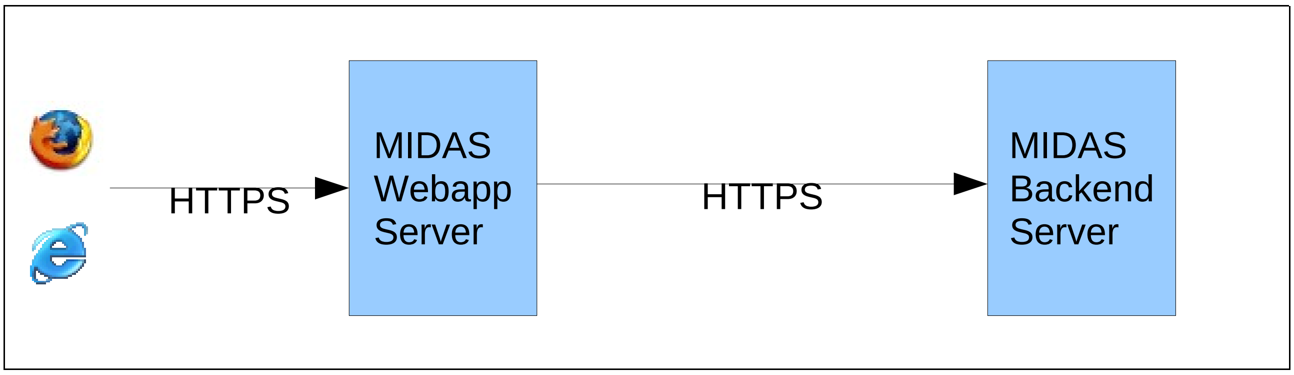
HTTPS communication provides the following two security-related features in addition to basic HTTP:

- authentication – the communication peers can be sure to really talk to the partner they think they do
- encryption – the HTTPS data flow is SSL encrypted

Within CVP there are the following two applicable HTTP/HTTPS relationships:

- GUI/WebApp – the WebApp acts as HTTP/HTTPS server and the GUI as client

- WebApp/BackEnd – the BackEnd acts as HTTP/HTTPS server and the WebApp as client



In complex environments there may be multiple BackEnds, WebApps and GUIs. The logical relations are the same though.

There are the following aspects of HTTPS communication within by CVP:

- Authentication
  - Server authentication – the HTTPS server has to present some credential the client can verify. This way the client can be sure to talk to a real server. This is mandatory when using HTTPS.
  - Client authentication – in addition to server authentication, also the client has to authenticate itself to the server. This way also the server knows that it talks to the client it thinks it does.
- Authorization – this is actually not part of HTTPS but can be used as well, particularly to restrict the set of clients who are allowed to perform operations on the server.

Client authentication and authorization are optional, but particularly in the relationship WebApp - BackEnd both are strongly recommended to make sure that only authorized clients perform operations on a BackEnd. It is also conceivable and useful for the GUI – WebApp communication to allow only a certain set of users access (those who have a client certificate which is registered at the WebApp).

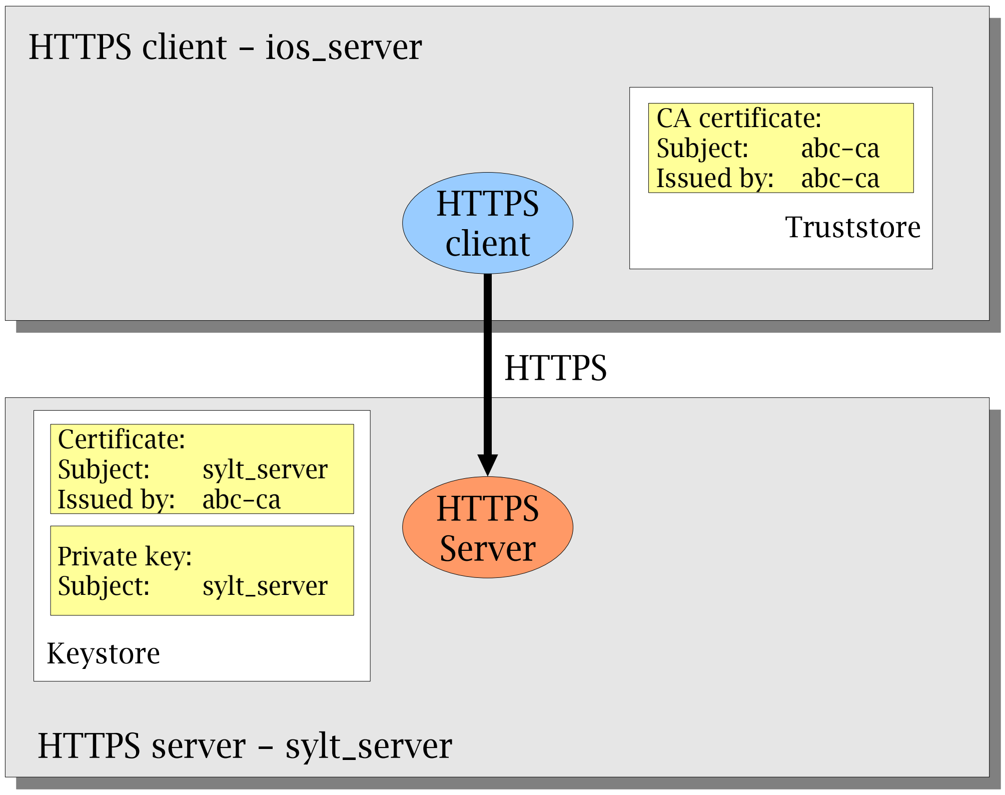
9.7.2.2 HTTPS and Java

HTTPS communication requires some special consideration (for HTTP no extra steps are needed) in the area of keys and certificates.

The HTTPS servers (i.e. the BackEnd from the WebApp perspective, and the WebApp from the GUI perspective) must be able to present a server certificate, which the HTTPS client is able to verify. CVP uses the standard Java mechanism of *keystores* and *truststores*. A keystore contains

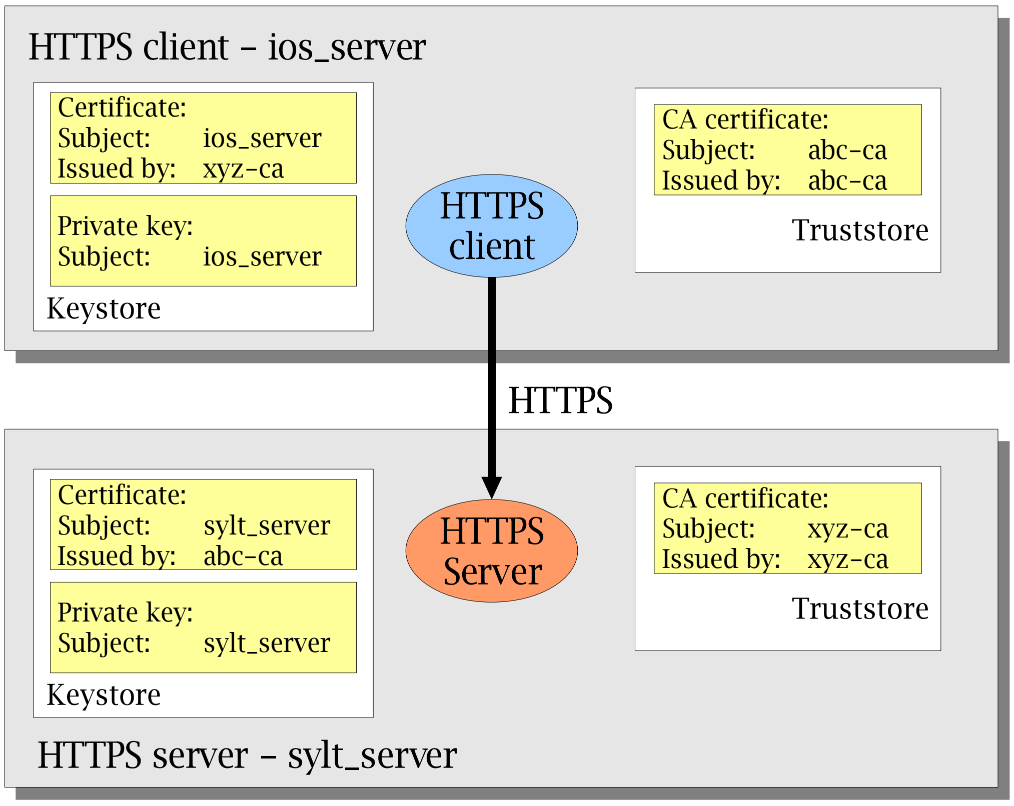
- private keys used to encrypt and sign data
- certificates presented to communication peers

A truststore contains certificates which are considered as trusted (issued by *certificate authorities, CAs*). These CA certificates are used to verify peer certificates. For server authentication the following picture illustrates the set-up:



In this example the WebApp **ios_server** (acting as HTTPS client) connects to the BackEnd **sylt_server** (HTTPS server). The HTTPS server has to present its server certificate (contained in its keystore), which is signed by the CA **abc-ca** (whoever this is, we will look into this later). The HTTPS client must have a copy of the abc-ca in its truststore certificate to be able to verify the server certificate. The abc-ca certificate is *self-signed*, i.e. it is assumed to have been issued by a trust-worthy party (a CA) and no further certificates are needed to verify it (which is possible though).

For client authentication (no authorization yet) the following set-up is needed:



Again, the WebApp ios_server (acting as HTTPS client) connects to the BackEnd sylt_server (HTTPS server). The HTTPS server authenticates with its server certificate (sylt_server, issued by CA abc-ca).
In addition, the HTTPS client has an own certificate issued to **ios_server** by the CA **xyz-ca**. The HTTPS server must have a copy of the xyz-ca in its truststore certificate to be able to verify the client certificate. The CA certificates abc-ca and xyz-ca can be possibly the same.

The HTTPS server can be configured to request client authentication globally (either for all clients or none). For CVP, client authentication is enabled by default between WebApp and BackEnd but disabled between GUI and WebApp.

Key- and truststores are normally protected by a password which can be specified when creating. To deal with key- and truststores the `keytool` command is recommended since it is part of the JRE which in turn is bundled with CVP. All following examples assume `keytool`.

To import a certificate into a keystore or truststore, the following command has to be used:

```
# <MIDAS_HOME>/jre/bin/keytool -import -file <cert file>
```

The keytool command can also be used to display the contents of key- and truststores:

```
# <MIDAS_HOME>/jre/bin/keytool -list -keystore <cert file>
```

For more details about the `keytool` command, keystores, truststores etc. refer to the man page of keytool(1):

```
# man keytool
[...]
```

or http://java.sun.com.

Well-known CA certificates are normally built-in to web browsers and Java JREs. To display the contents of the standard truststore in a JRE:

```
# <MIDAS_HOME>/jre/bin/keytool -list -keystore
     <MIDAS_HOME>/jre/lib/security/cacerts -storepass changeit
```

If the default password *changeit* has been changed supply the correct password. The command should list something like (note the entry type *trustedCertEntry*):

```
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
Certificate fingerprint (MD5):
  78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D
[...]
```

Custom CA certificates can be registered by importing them to this truststore or by using a custom truststore. Within CVP, the latter will be done.

9.7.2.3 Configuring HTTPS in CVP

HTTPS communication can be used within CVP in the following combinations:

- between GUI and WebApp
  - server authentication only
  - client authentication (optional, by default disabled)
- between WebApp and BackEnd
  - server authentication, client authentication

By default, CVP generates self-signed certificates on all CVP servers which can be used for out-of-the-box HTTPS communication. If this is an acceptable security level, nothing has to be done in addition. Otherwise certificates may need to be generated and imported key- and truststores.

Certificates within CVP are standard PKCS12 certificate file in DER format, which can be generated using any desired mechanism, e.g. `keytool`, openssl, the SSH key generator command or the OVO SecCore functionality. Again, `keytool` is recommended.

The key- and truststores used by a CVP server are in

```
# ls -l <MIDAS_HOME>/conf/*store*
-r--r--r--  1 root root 1093 2006-11-22 16:24 keystore_endpoint.jks
-r--r--r--  1 root root 3243 2006-11-15 09:32 keystore.jks
-r--r--r--  1 root root  686 2006-11-22 16:24 truststore_endpoint.jks
```

```
-r--r--r--  1 root root  662 2006-11-15 09:32 truststore.jks
```

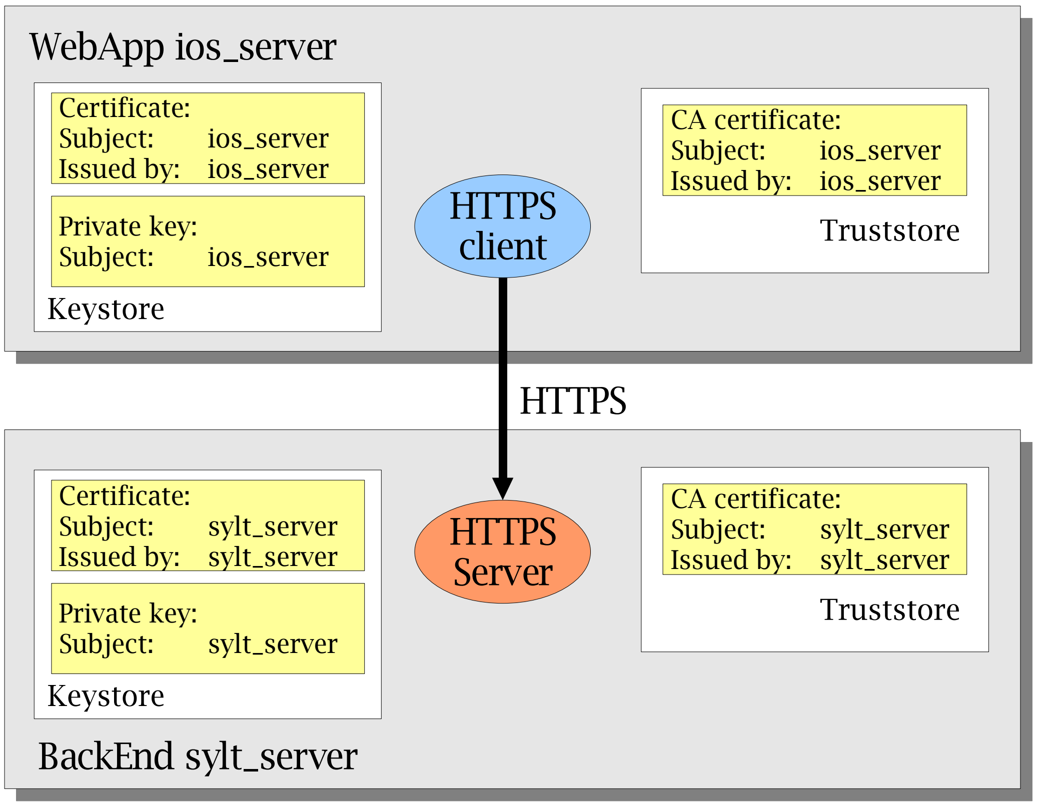
The files named `*_endpoint.jks` are related to the CVP HTTPS communication. All following references to *keystore* and *truststore* are related to these two files.

The other two files are required by infrastructure components (FTP Server, Jetty Web container and ServiceMix). These must not be modified.

Authorization within CVP is controlled entirely by the CVP user model. Therefore there is nothing to be configured elsewhere. If an HTTPS client is able to establish an HTTPS connection (and the certificate exchange has been successful), it is also considered as authorized to perform all tasks.

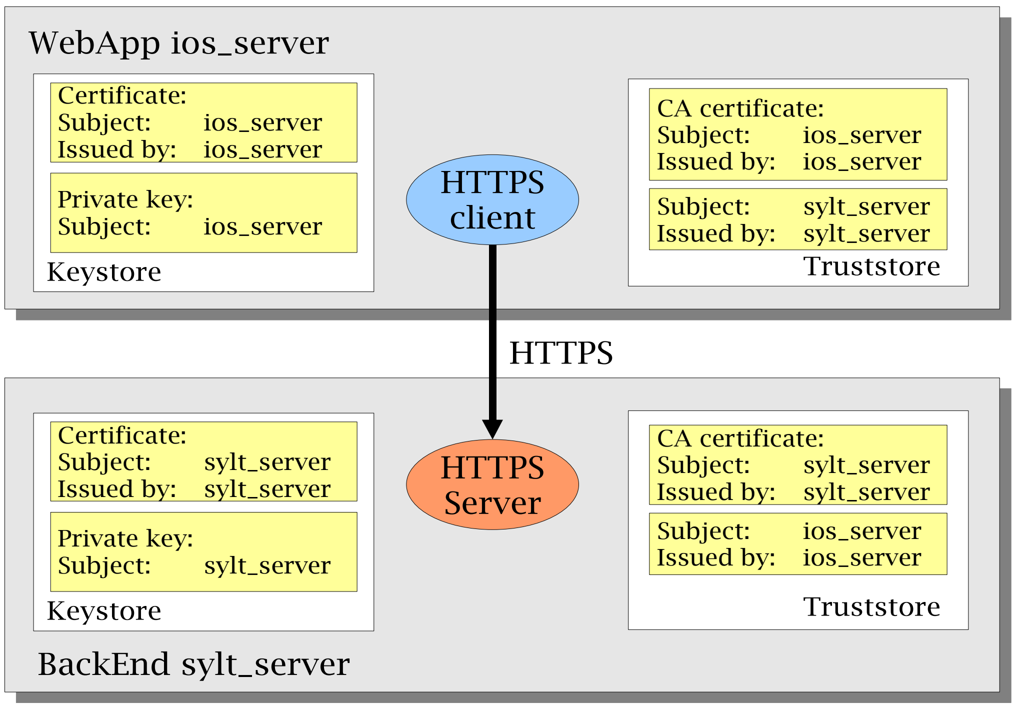
9.7.2.4 HTTPS between WebApp and BackEnd

By default the following scenario exists after a completed CVP installation with a separate WebApp and BackEnd:



This means, that only local HTTPS communication will work correctly, since the only trusted CA certificates are the self-signed ones used by the local CVP server itself. This is important though, because this happens when using local command line tools like `midas.sh`.

To make HTTPS communication functional between these two CVP servers, the following set-up must be accomplished:



Now the WebApp ios_server accepts certificates which are signed by either the CA ios_server (local traffic) or sylt_server (HTTPS communication with the BackEnd sylt_server). Vice versa, the client-side certificate presented by the WebApp ios_server will be trusted by the BackEnd sylt_server, because the CA certificate (who had signed the ios_server certificate) is present.

To accomplish this, perform the following steps:

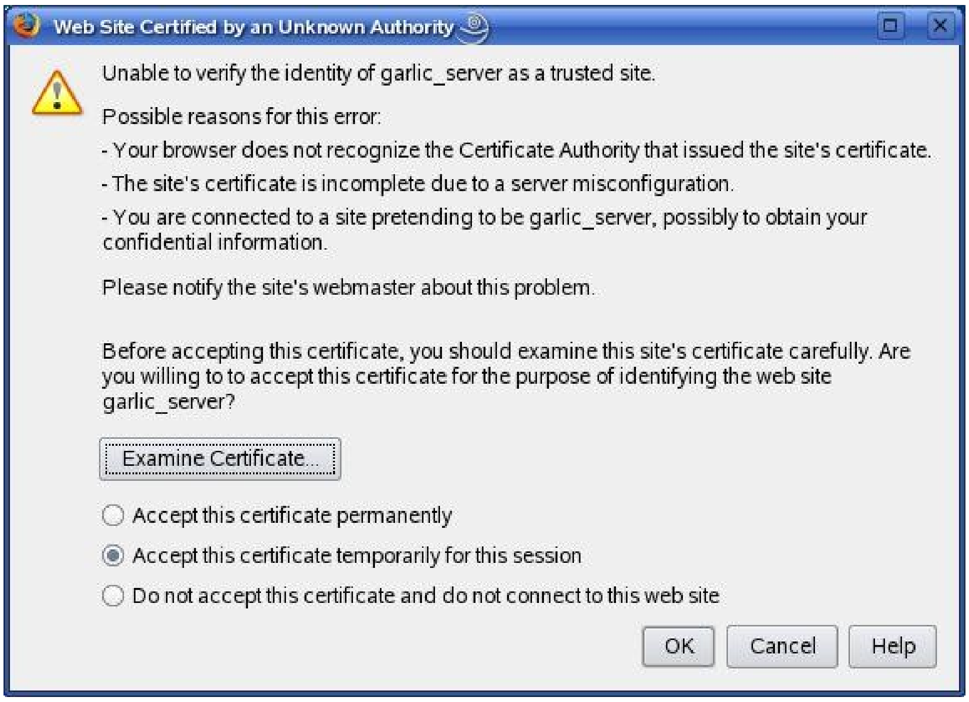
1. Export the CA certificate of ios_server from the truststore on ios:
   ```
   ios # keytool -export -keystore
     <MIDAS_HOME>/conf/truststore_endpoint.jks -alias ios_server -file
     /tmp/ios.cert
   ```
2. Transfer the file `/tmp/ios.cert` to sylt_server (via scp, ftp, external media, ...), make sure not to modify the file
3. Import the ios_server CA certificate on sylt_server into the truststore:
   ```
   sylt # keytool -import -keystore
        <MIDAS_HOME>/conf/truststore_endpoint.jks -file /tmp/ios.cert
   ```
4. Re-start both CVP servers

For the steps 1 and 3 you will be queried for a password – this is by default *password*.

9.7.2.5 HTTPS between web browser and WebApp

For HTTPS traffic between GUI and WebApp, the same rules apply as between WebApp and BackEnd.

For out-of-the-box server-side authentication, nothing has to be configured, the user has just to accept the WebApp certificate in the browser (here the web browser complains, that it cannot validate the default self-signed certificate, because it does not have the signing CA certificate):



Web Site Certified by an Unknown Authority

Unable to verify the identity of garlic_server as a trusted site.

Possible reasons for this error:

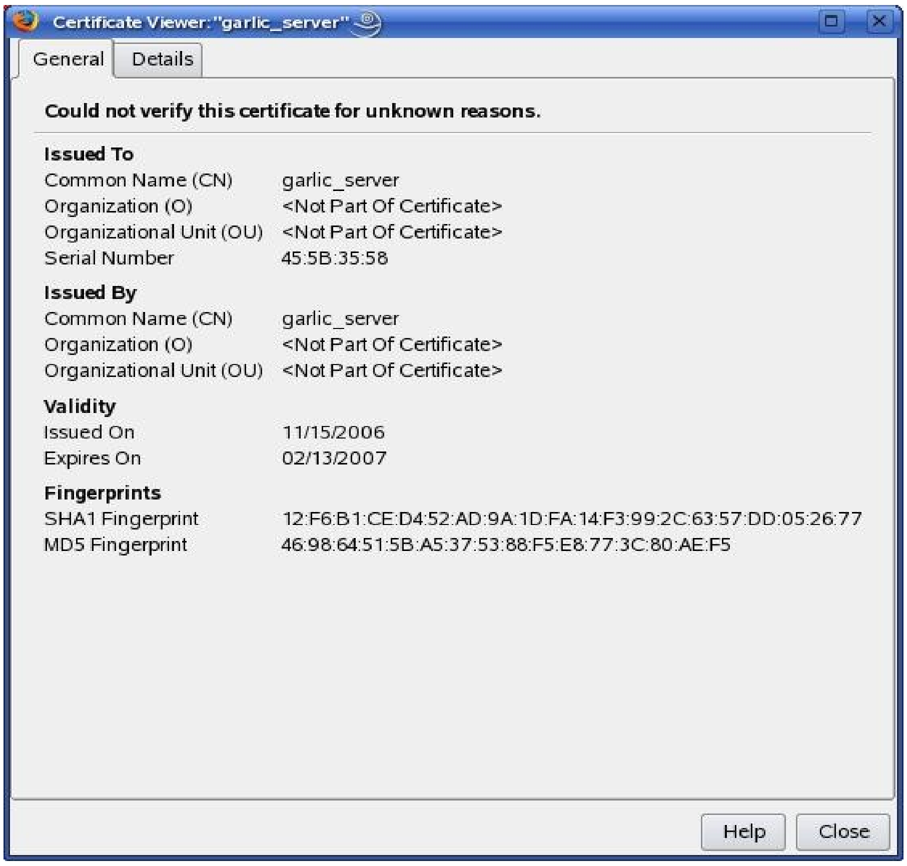
- Your browser does not recognize the Certificate Authority that issued the site's certificate.
- The site's certificate is incomplete due to a server misconfiguration.
- You are connected to a site pretending to be garlic_server, possibly to obtain your confidential information.

Please notify the site's webmaster about this problem.

Before accepting this certificate, you should examine this site's certificate carefully. Are you willing to to accept this certificate for the purpose of identifying the web site garlic_server?

( ) Accept this certificate permanently
[✓ Accept this certificate temporarily for this session]
( ) Do not accept this certificate and do not connect to this web site

To inspect the certificate, click the according button:

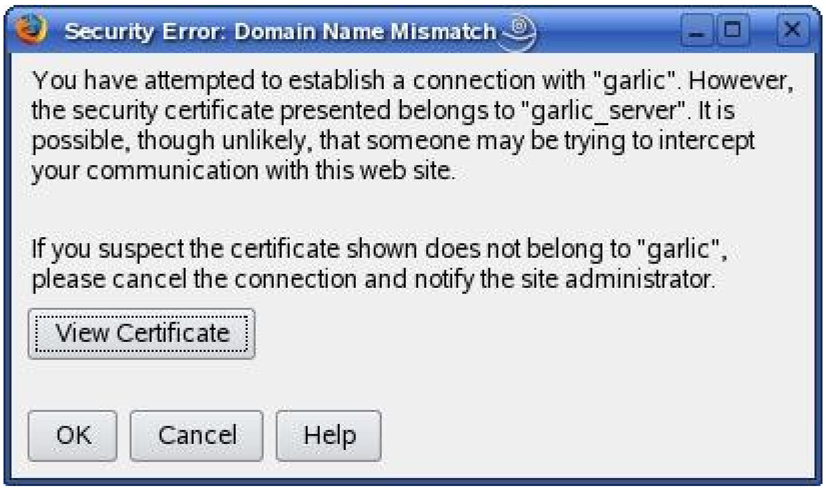


Certificate Viewer:"garlic_server"

Could not verify this certificate for unknown reasons.

Issued To	
Common Name (CN)	garlic_server
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	45:5B:35:58
Issued By	
Common Name (CN)	garlic_server
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>
Validity	
Issued On	11/15/2006
Expires On	02/13/2007
Fingerprints	
SHA1 Fingerprint	12:F6:B1:CE:D4:52:AD:9A:1D:FA:14:F3:99:2C:63:57:DD:05:26:77
MD5 Fingerprint	46:98:64:51:5B:A5:37:53:88:F5:E8:77:3C:80:AE:F5

The following warnings may appear – just click OK to proceed (the reason is, that usually HTTPS server certificates are issued to the ***common name (CN)*** which equals the server host name. This has not been implemented in CVP):



You have attempted to establish a connection with "garlic". However, the security certificate presented belongs to "garlic_server". It is possible, though unlikely, that someone may be trying to intercept your communication with this web site.

If you suspect the certificate shown does not belong to "garlic", please cancel the connection and notify the site administrator.

NOTE: If the names appearing here are **NOT** the WebApp host name and the CVP server ID defined during installation of the WebApp, possibly a real security threat exists.
Otherwise the combination of both reflects the correct situation.

You can choose whether to accept this certificate permanently or repeat this process again next time. If accepting the certificate permanently, it will be placed in the web browsers truststore.

This step can be avoided entirely, if the WebApp certificate is signed by a CA registered in the web browser. This can be done by installing a certificate on the WebApp, which is issued by a CA (either well-known or custom), whose certificate is already present in the web browsers.

To do this:

- Put CA-signed certificate in WebApp keystore
- Load CA certificate into browser (if not in there yet)

After exchanging the certificates, the CVP WebApp server has to be restarted.

In addition, you may configure client-side certificates for the GUI – WebApp relationship:

- Create CA-signed certificate and load this into the web browser as client certificate
- Put the CA certificate of the CA, who has signed the client certificate, into WebApp truststore (if not yet in)
- Configure the WebApp to enforce client-side certificates (see below)
- When connecting the WebApp with the web browser, you may need to select the correct client certificate (if there is more than one)

To configure client certificates in the WebApp, set the property `needClientAuth` as:

```
# vi <MIDAS_HOME>/conf/jetty.properties

jetty.connectors[0].port=9662
jetty.connectors[1].port=9663
jetty.connectors[1].needClientAuth=true
```

9.7.2.6 Using custom certificates

If custom keys and certificates are to be used (e.g. the customer has an own CA or keys/certificates used elsewhere), just the default keys and certificates need to be replaced. Important is to accomplish the situation as explained in chapter 9.7.2.4 HTTPS between WebApp and BackEnd.

To install a custom CA certificate, it must be imported to the web browser or the JRE on the CVP server. If the CVP server certificate has not been signed by a well-known CA, it is necessary to register that CA's certificate as trusted on the HTTPS client. This is not needed, if the server certificate has been issued by any CA which is listed when calling the following command on the CVP server:

```
# <MIDAS_HOME>/jre/bin/keytool -list -keystore
     <MIDAS_HOME>/jre/lib/security/cacerts -storepass changeit
[...]
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
verisignclass3ca, Jun 29, 1998, trustedCertEntry,
Certificate fingerprint (MD5):
  78:2A:02:DF:DB:2E:14:D5:A7:5F:0A:DF:B6:8E:9C:5D
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5):
  3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D
[...]
```

If your server certificate has been issued by any of these CAs it is trusted automatically - then nothing needs to be done here. Otherwise import your CA certificate into the CVP truststore by executing:

```
# <MIDAS_HOME>/jre/bin/keytool -import -keystore
     <MIDAS_HOME>/conf/truststore_endpoint.jks -storepass <password>
     -file <your CA cert file in DER format>
```

Then the truststore looks like this:

```
Alias name: mykey
Creation date: Dec 4, 2003
Entry type: trustedCertEntry

Owner: EMAILADDRESS=tge@blue-elephant-systems.com, CN=tge, OU=R&D, O=blue
elephant systems GmbH, L=Stuttgart, ST=Baden-Wuerttemberg, C=DE
Issuer: EMAILADDRESS=tge@blue-elephant-systems.com, CN=tge, OU=R&D, O=blue
elephant systems GmbH, L=Stuttgart, ST=Baden-Wuerttemberg, C=DE
Serial number: 0
Valid from: Wed Dec 03 14:26:54 CET 2003 until: Fri Dec 02 14:26:54 CET 2005
Certificate fingerprints:
         MD5:  14:D1:9B:08:7B:4D:61:B3:D4:29:B8:26:E9:A2:B5:CE
         SHA1: 9D:01:8D:15:D6:C7:8C:93:21:FF:39:6A:3E:74:BC:08:36:9F:8E:61
```

Here the -v option has been used to print more details. Note again the entry type *trustedCertEntry* , the certificate identity and the CA identity - for CA certificate they are normally identically (here both are 'tge').

To use custom client or server certificates (in this example for the CVP server ios_server), perform the following steps:

1. Create a keystore containing the custom certificate:

   ```
   # keytool -genkey -keystore /tmp/my_keystore
             -storepass <password> -alias ios_server
   ```

   The alias name must be the CVP server ID.

2. Answer the questions about your identity. Then extract a certificate request which has to be sent to a CA:

   ```
   # keytool -certreq -alias ios_server -keystore /tmp/my_keystore
             -storepass <password> -file ios-certreq.pem
   ```

3. The CA signs the certificate request and returns the certificate (needed in DER format). Import this into the keystore:

   ```
   # keytool -import -alias ios_server -keystore
             <MIDAS_HOME>/conf/keystore_endpoint.jks
             -storepass <password> -file ios-cert.der
   ```

   The actual value of the alias is not important for client certificates, but it must be the same as specified when generating the initial key pair. Then the contents of the keystore looks like this:

```
Alias name:  ios_server
Creation date: Dec 8, 2003
Entry type: keyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=ios_server, OU=TM4WM, O=blue elephant systems GmbH, ST=Baden-
Wuerttemberg, C=DE
Issuer: EMAILADDRESS=tge@blue-elephant-systems.com, CN=tge, OU=R&D, O=blue
elephant systems GmbH, L=Stuttgart, ST=Baden-Wuerttemberg, C=DE
Serial number: 5
Valid from: Mon Dec 08 18:42:13 CET 2003 until: Tue Dec 07 18:42:13 CET 2004
Certificate fingerprints:
         MD5:  04:FD:A6:95:F4:B1:19:BC:70:E8:5E:48:D6:3D:CA:17
         SHA1: 59:61:C7:DB:E9:34:17:9F:5B:9D:E2:14:B0:B0:6E:40:B0:C7:8B:A6
Certificate[2]:
Owner: EMAILADDRESS=tge@blue-elephant-systems.com, CN=tge, OU=R&D, O=blue
elephant systems GmbH, L=Stuttgart, ST=Baden-Wuerttemberg, C=DE
Issuer: EMAILADDRESS=tge@blue-elephant-systems.com, CN=tge, OU=R&D, O=blue
elephant systems GmbH, L=Stuttgart, ST=Baden-Wuerttemberg, C=DE
Serial number: 0
Valid from: Wed Dec 03 14:26:54 CET 2003 until: Fri Dec 02 14:26:54 CET 2005
Certificate fingerprints:
         MD5:  14:D1:9B:08:7B:4D:61:B3:D4:29:B8:26:E9:A2:B5:CE
         SHA1: 9D:01:8D:15:D6:C7:8C:93:21:FF:39:6A:3E:74:BC:08:36:9F:8E:61
```

   Here we see the certificate issued to the identity of *ios_server* by the CA *tge*. The entry type *keyEntry* states that this is not a CA certificate but a regular private key/certificate pair.

4. Finally, the CA's certificate has to be registered on the HTTPS server as trusted certificate.

9.7.3 Using proxies

Using a HTTP proxy between Web browser and CVP WebApp is currently not supported.

9.7.4 Using CVP in firewall environments

CVP supports the scenario, that different CVP server components (BackEnd and/or WebApp systems) are running in different networks, possibly separated by firewalls. Then, it is important to know the CVP communication flow to configure the according firewall rules.

All CVP communication is TCP/IP based.

With a CVP standard installation only one port is absolutely required:

- For communication between the CVP Web Application to the CVP BackEnd server using HTTP(S) (default port 9661)

Optionally, for volume data traffic using FTP or SSH:

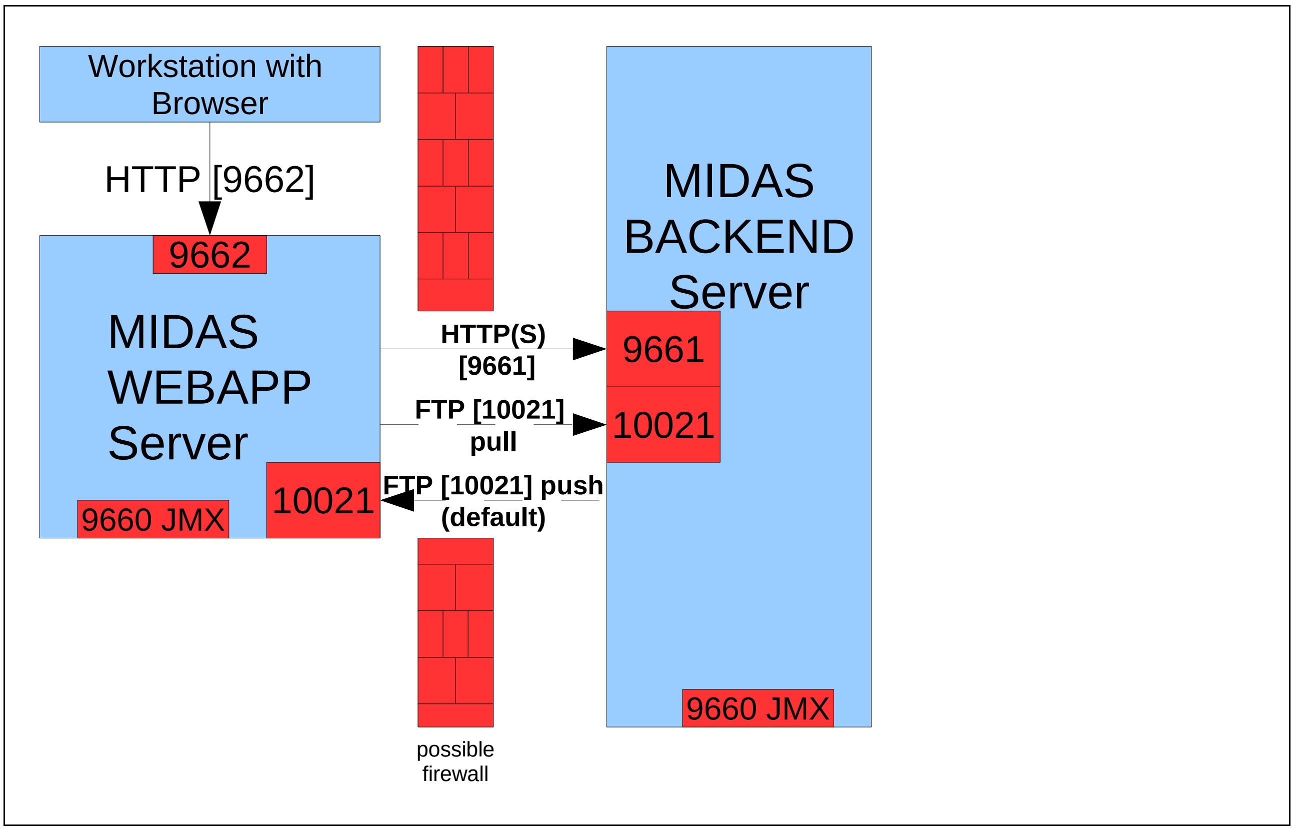
- Default port 10021 for any file transfers via FTP
- TCP/22 for SSH transfers

may be used (or none or both).

A JMX console can be connected to every CVP server (by default on port 9660), but this is entirely optional. This can be done remotely too (then, the port must be allowed in a firewall).

This yields the following scenarios:

Example: Webapp server and one OVO Backend in different networks

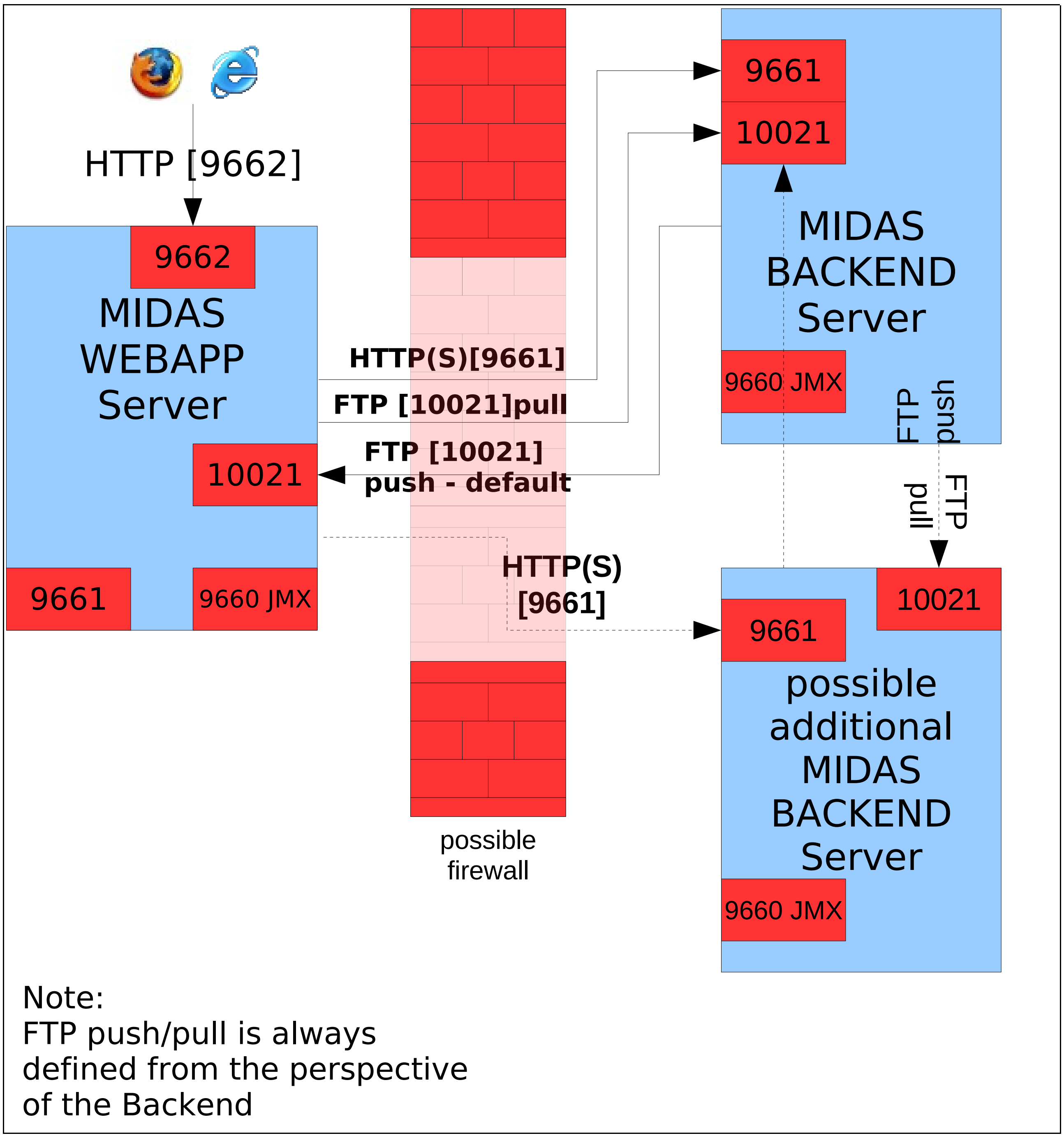


Source Host	Source Port	Target Host	Target Port	Description
web application	*	backend server	9661	HTTP(S) communication between webapp to any

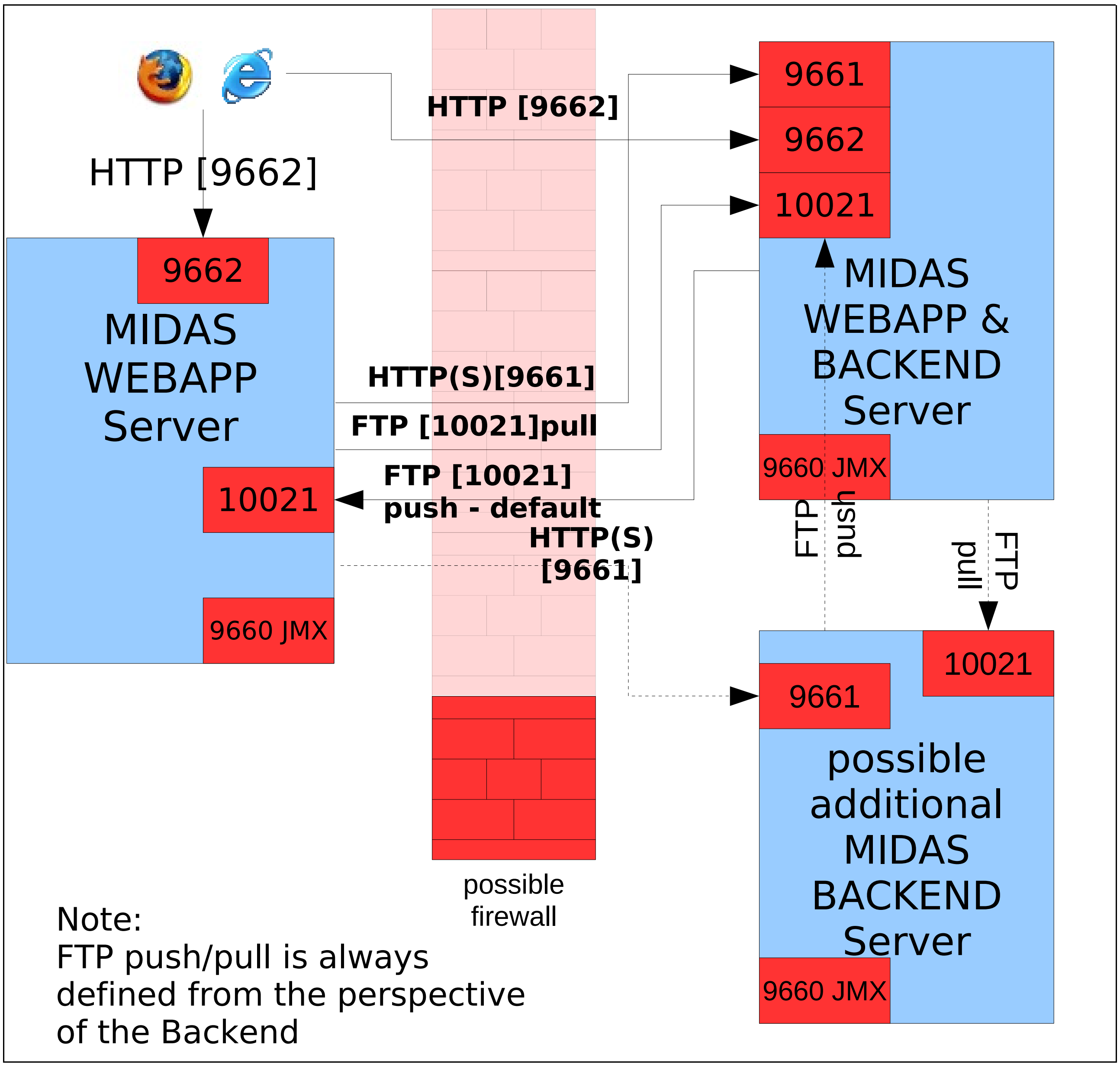
Source Host	Source Port	Target Host	Target Port	Description
				backend server (OVO, NNM, task etc.)
web application	*	backend server	10021	FTP access to download or log files from the web application (push mode)
Backend server	*	ftp server	10021	FTP access for file transfers between backends (pull mode)

These are the default ports. If different values have been specified during installation, apply these as appropriate.

Example: One WebApp with two BackEnd servers:



Example: Separate WebApp, one server with a full CVP installation (WebApp & BackEnd) and one separate BackEnd server:



9.8 CVS configuration

There are only two things to configure regarding CVS within CVP:

- the location of the CVS binary
- the CVS repository

All CVS-related set-up has to be done on the WebApp system. The actual CVS software must be installed and CVP must be pointed to the CVS executable. See chapter 7.2 Concurrent versioning system (CVS)for details.

During installation, an empty CVS repository will be created on the WebApp system. This is any other CVS repository (local or remote) can be used, just make sure that CVP is able to access it.

9.8.1 Using an alternative CVS repository

By default, CVP uses a CVS repository created during installation on the WebApp system. Alternatively, a different local or a remote CVS repository can be used. To do so, update the entry in `<MIDAS_HOME>/conf/vcs.properties`:

```
# cat /opt/midas30/conf/vcs.properties
cvs.cvsRoot=:local:/opt/midas30/data/repository
```

Refer to the CVS documentation about the syntax and possible values. CVP also supports SSH access with a special connection protocol ssh, i.e. a resulting CVSROOT string looks then like `:ssh:potato:/repo` for a repository path `/repo` on host `potato`.

NOTE: CVP currently supports only the connection methods `local, pserver and ssh`.

In addition, depending on the connection method, a user name and password for accessing the remote repository may be required. For `ssh` and `pserver` methods, these are for example the user *tge* with the password *secret123*:

```
# cat /opt/midas30/conf/vcs.properties
cvs.user=tge
cvs.password=secret123
```

Currently the password is required in plain text.

Currently, CVP is no able to deal with more than one repository at the same time.

9.9 File transfer configuration

As described in chapter 4.3.1 FTP vs. SSH communication volume data can be transferred between CVP servers using FTP or SSH. Both mechanisms can be selected and configured initially during installation, but if something has to be changed later, review the following steps.

9.9.1 Configuring FTP transfer

To verify the FTP server setting on a CVP server, review the file:

```
# cat <MIDAS_HOME>/conf/ftpd.xml

<?xml version="1.0"?>
<!-- XML file based configuration -->
<config>
  [...]
  <socket-factory>
    <class>org.apache.ftpserver.socketfactory.FtpSocketFactory</class>
    <port>10021</port>
  [...]
  <data-connection>
    <pasv-port>0</pasv-port>
    <port-enable>true</port-enable>
    <port-ip-check>false</port-ip-check>
  [...]
  <ip-restrictor>
    [...]
    <file>/opt/midas30/conf/ip.properties</file>
  </ip-restrictor>
  [...]
  <user-manager>
    [...]
    <admin>admin</admin>
    <prop-file>/opt/midas30/conf/user.properties</prop-file>
    <prop-password-encrypt>true</prop-password-encrypt>
  </user-manager>
  [...]
```

The XML tag `<port>` specifies the port (default 10021) on which the FTP server will listen. Make sure that this port can be reached from other CVP servers, for which FTP transfer is desired.

9.9.1.1 Enabling/Disabling the FTP service

If the FTP server component is not desired at all (or it has been disabled previously and needs to be activated again), the property `ftp.enabled` can be set in the following configuration file:

```
# vi <MIDAS_HOME>/conf/file.properties
licenses.holderName=blue elephant systems GmbH
ftp.enabled=true
```

9.9.1.2 Changing the FTP server port

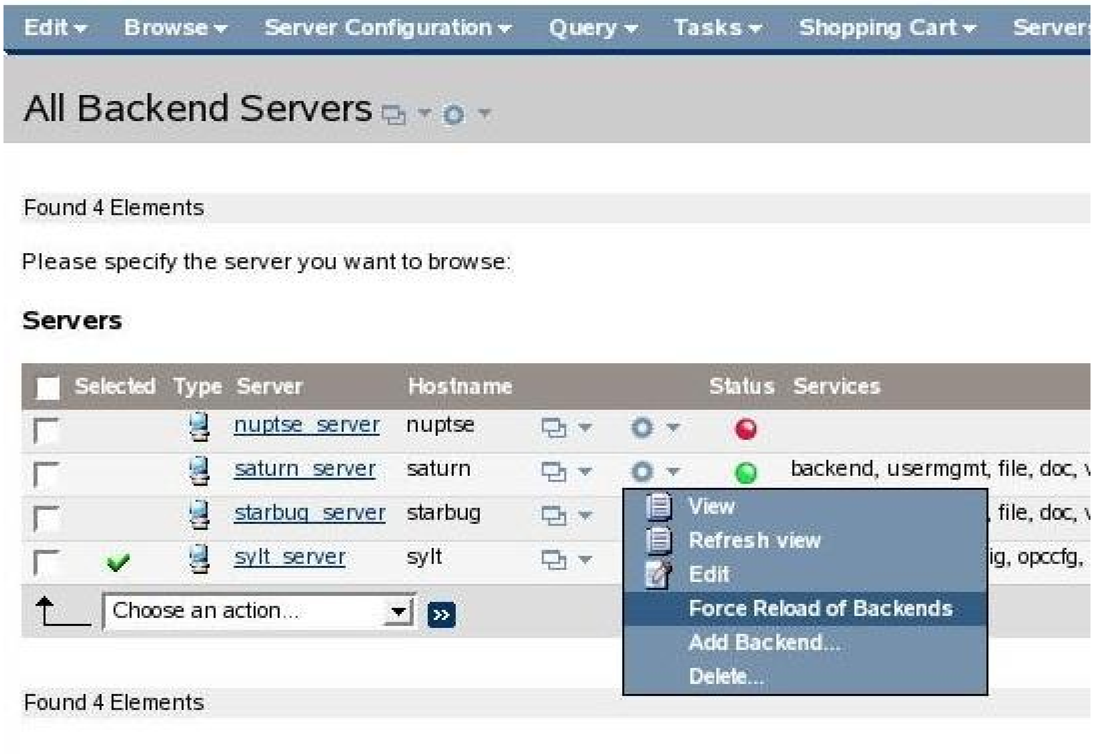
To change the FTP server port, update the file above and re-start the CVP server. In addition, the FTP clients must be updated accordingly – see below.

9.9.1.3 Changing FTP ports for peer servers

If a CVP server uses a non-standard FTP port, all other CVP servers possibly acting as FTP clients must be aware of this. However, there is no need to configure anything here.

The FTP server port is part of the BackEnd resolution process, which is performed automatically by a CVP server upon start-up or any related configuration change such as the registration of a new BackEnd.

This process can be triggered in the GUI:



All Backend Servers

Found 4 Elements

Please specify the server you want to browse:

Servers

Selected	Type	Server	Hostname	Status	Services
		nuptse_server	nuptse	●	
		saturn_server	saturn	●	backend, usermgmt, file, doc, v
		starbug_server	starbug		, file, doc, v
✓		sylt_server	sylt		ig, opccfg,

Found 4 Elements

This operation sends a request to the selected CVP server and queries all configuration values (incl. the FTP server configuration).

9.9.1.4 Controlling FTP access

On FTP servers, the set of CVP servers which may access the local FTP service may be restricted. By default, only the local system is configured, and, for standalone BackEnds, the initial WebApp specified during installation.

To verify and change this, edit the file `ip.properties`:

```
# vi <MIDAS_HOME>/conf/ip.properties
127.0.0.1
localhost
ios
```

The host names and IP addresses listed here are the ones which are allowed to access this FTP server (in this example **ios** is the local host name). Make sure, all names are resolvable with name service and all desired CVP servers acting as FTP client are registered.

Also IP address patterns like `192.168.*` are can be configured to specify the set of permitted FTP client hosts.

9.9.2 Configuring SSH transfer

To use SSH transfer between CVP servers, an non-interactive SSH relationship must be established, i.e. which does not require any user data like passwords.

Currently, both CVP servers must have SSH software installed. For details regarding SSH installation and configuration see chapter 7.3 Secure Shell (SSH).

9.10 Tuning Java parameters

9.10.1 Virtual memory

For large numbers of users or objects, and if the system the CVP server is running on has plenty of RAM installed, performance can be improved significantly by allowing the JRE to obtain more virtual memory.

This can be controlled by increasing the following standard Java JRE parameters:

```
# vi <MIDAS_HOME>/conf/wrapper.conf
[...]
# Initial Java Heap Size (in MB)
wrapper.java.initmemory=64

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=384
[...]
```

Do no decrease the value below the initial setting – this will also decrease performance and CVP may not function properly anymore.

9.10.2 Start-up time

On slow systems, the start-up of the CVP server may take quite a while. To prevent the controlling `wrapper` process from erroneously re-starting the JRE, the following parameter may be increased:

```
# vi <MIDAS_HOME>/conf/wrapper.conf
[...]
# Number of seconds to allow for the JVM to be launched and contact the
# wrapper before the wrapper should assume that the JVM is hung and
# terminate the JVM process. 0 means never time out.
# Defaults to 30 seconds.
wrapper.startup.timeout=300

# Number of seconds to allow between the wrapper pinging the JVM and
# the response. 0 means never time out. Defaults to 30 seconds.
wrapper.ping.timeout=100

# Number of seconds to allow for the JVM to shutdown before the wrapper
# should assume that the JVM is hung and terminate the JVM process.
# 0 means never time out. Defaults to 30 seconds.
wrapper.shutdown.timeout=300
[...]
```

10 Troubleshooting

The following sections contain information, tools and procedures helpful for troubleshooting CVP.

10.1 General procedures

In general, whenever experiencing problems, think about the following (some of that appears obvious but sometimes it may be helpful to remember the obvious things):

- Describe the problem as precise as possible
  - Error reports “Cannot edit a policy” are not very helpful
  - Provide screenshots, shell output data, log files, the support.zip file
- Think about differences
  - to other instances (“It works for all items, except this”). What is special about the non-working item?
  - “Yesterday it used to work”. So, what has changed since then?
- Try to determine whether the problem is in OVO or in CVP
  - Can you perform the same operation using native OVO tools (GUI, commands)?
    - opcragt, opcnode, opchbp, ...
  - Are there any related entries in the OVO error log files
    - OVO <= 7.x: /var/opt/OV/log/OpC/mgmt_sv/opcerrror
    - OVO >= 8.x: /var/opt/OV/log/System.txt
  - Use OVO tracing. Most CVP operations are performed by calling the OVO API. This can be traced using the regular OVO tracing facility. The trace area of most interest is **API** (OVO 7-style tracing) or **opc.api** (OVO 8 XPL tracing).
    Possibly it may be needed to re-start the CVP server to be traced.
  - Review the Oracle database. All OVO configuration data exists in the central Oracle database. The `sqlplus` command may yield hints about how the data looks like in the database.
    DO NOT modify anything this way!
- Verify the OS resources
  - Insufficient disk space or full kernel tables are common causes. Such problems may also be logged in syslog (Unix) or the system event log (Windows).

Also, most efficient troubleshooting strategies involve:

- First, the perform simple steps to positively include or exclude the most likely causes.
  - Commands, which can be typed in quickly and provide fast results
  - Error log files
- More powerful capabilities like tracing usually involve more effort (configuring trace level, possibly restart services, review trace data, ...)

- If you have some suspicion, try to be certain (also determining, that something is NOT the problem, may help).

Always supply the support.zip file (see below) to the support team.

10.2 X-related problems

Since CVP is mainly web-based, there are normally no situations where this might cause problems, except

- when trying to re-direct the display of the web browser
- during installation

In both cases the usual things to be verified are:

- is on the workstation, where the GUI is supposed to appear (target), remote X access allowed (use the command `xhost +` to allow it globally)?
- is the DISPLAY variable set on the system, where the actual program is executed (source) ?

In addition, there are two other common problem:

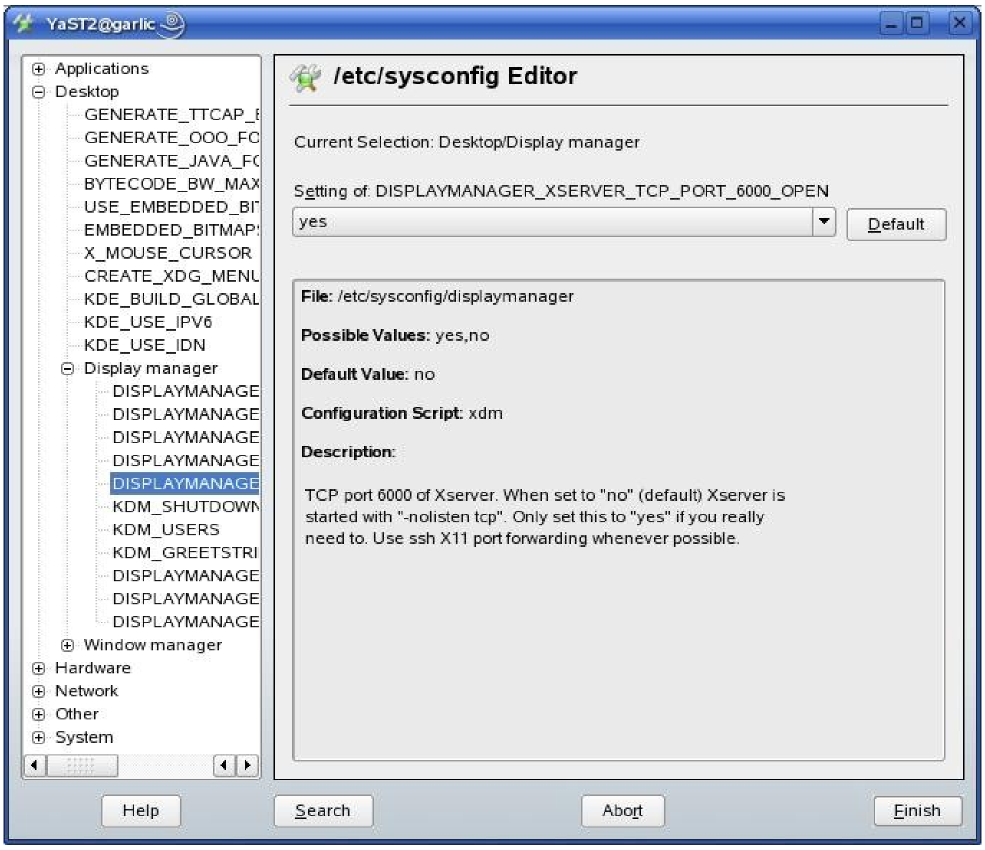
- If you see something like the following on HP-UX source systems

  ```
  Warning: Missing charsets in String to FontSet conversion
  Warning: Unable to load any usable fontset
  ```

  the following may help:

  ```
  # LANG=C.iso88591 <the command>
  ```

- On Linux target systems (particularly SuSE 10.x) it may be necessary to set various system settings related to DISPLAYMANAGER:



/etc/sysconfig Editor

Current Selection: Desktop/Display manager

Setting of: DISPLAYMANAGER_XSERVER_TCP_PORT_6000_OPEN

yes

File: /etc/sysconfig/displaymanager

Possible Values: yes,no

Default Value: no

Configuration Script: xdm

Description:

TCP port 6000 of Xserver. When set to "no" (default) Xserver is started with "-nolisten tcp". Only set this to "yes" if you really need to. Use ssh X11 port forwarding whenever possible.

10.3 Logging

Detailed information about the operation can be logged to various platform and adapter-specific log files. By default, CVP server log files are located in:

```
<MIDAS_HOME>/logs/*.log
```

In addition, there are log files written by the CVP Web Application in:

```
<MIDAS_HOME>/webapps/<comp>/work/webapp/WEB-INF/logs/*.log
```

Here, `<comp>` stands for `midas` or `exist` representing the logical CVP WebApp component or the built-in XML database.

By default, only errors should be logged. For troubleshooting it is possible to enable tracing to log even more details about the operation of the single CVP components.

For information how to set the log level, please refer to chapter 9.3 Logging configuration.

The server log files created with a standard set-up are (all are located in `<MIDAS_HOME>/logs`):

Log file	Description
audit	Directory holding daily auditing log files (no rollback, no cleanup). With log level INFO one line per transaction. With log level DEBUG everything is logged.
backend.log	
dead.log	all illegal requests go into this log
file.log	
lock.log	log if a lock has occured
midas.log	catches all log events otherwise not specified (default log)
ovo.log	
servicemix.log	if an adapter doesn't start etc check this log first
task	Directory containing individual log files written during the execution of tasks (commands, config download, ...)
task.log	general log whether a task has been run
usermgmt.log	
vcs.log	
web.log	Webapp log
wrapper.log	stdout of the wrapper process is logged here
xmldb.log	

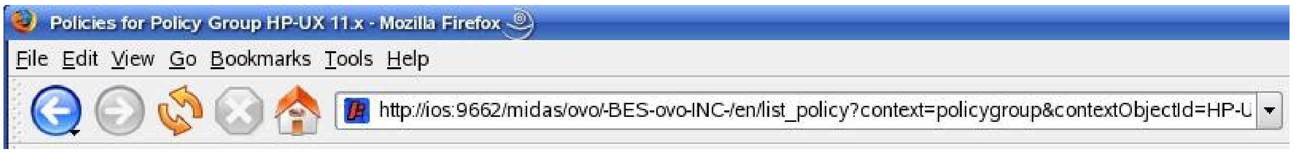
All log files in green are log files from the different adapters (e.g. ovoserver, docserver, fileserver etc.).

Therefore, if, for example, there is a problem with opc_adm or opc_op logging into OVO/Oracle check the ovo.log first.

10.4 Viewing raw XML data

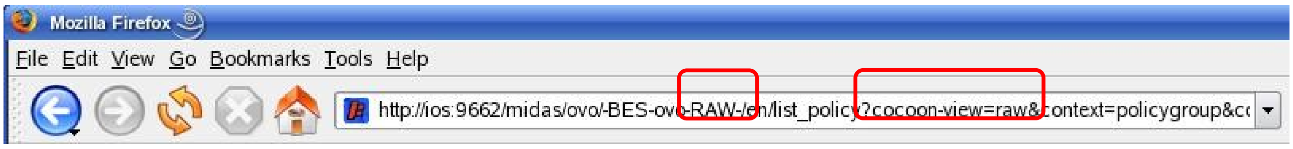
In case of problems displaying data correctly in the web browser, the raw XML data can be displayed as well.

To do this, modify the URL normally used, like:


Policies for Policy Group HP-UX 11.x - Mozilla Firefox
File Edit View Go Bookmarks Tools Help

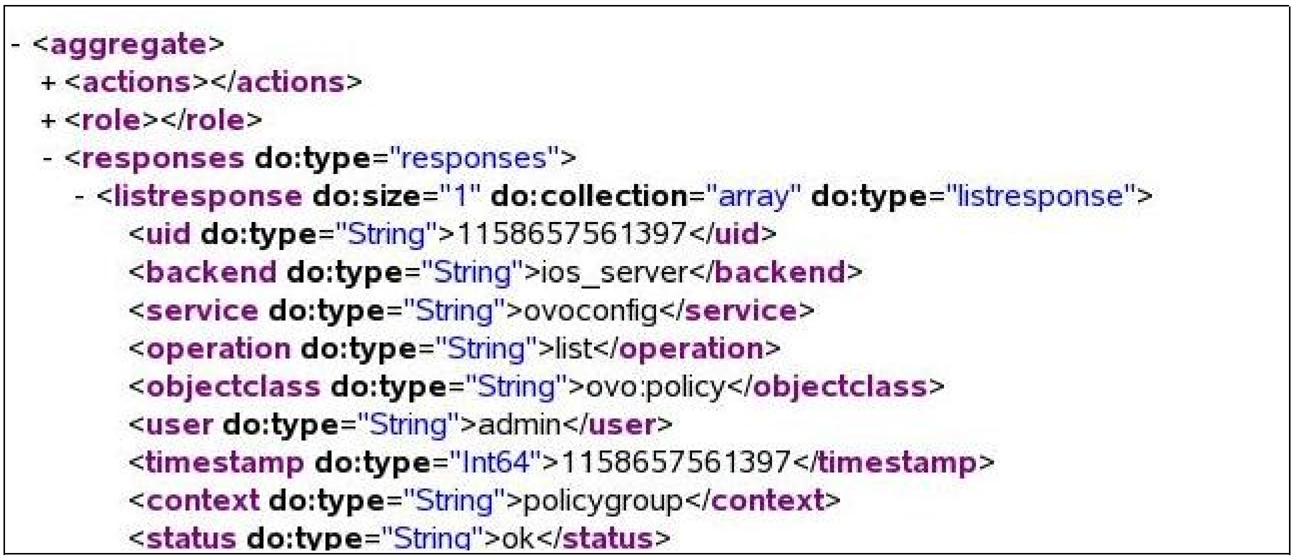
```
http://ios:9662/midas/ovo/-BES-ovo-INC-/en/list_policy?context=policygroup&contextObjectId=HP-U
```

and replace the indicated parts as:



```
http://ios:9662/midas/ovo/-BES-ovo-RAW-/en/list_policy?cocoon-view=raw&context=policygroup&c
```

which yields something like:



```
- <aggregate>
  + <actions></actions>
  + <role></role>
  - <responses do:type="responses">
    - <listresponse do:size="1" do:collection="array" do:type="listresponse">
        <uid do:type="String">1158657561397</uid>
        <backend do:type="String">ios_server</backend>
        <service do:type="String">ovoconfig</service>
        <operation do:type="String">list</operation>
        <objectclass do:type="String">ovo:policy</objectclass>
        <user do:type="String">admin</user>
        <timestamp do:type="Int64">1158657561397</timestamp>
        <context do:type="String">policygroup</context>
        <status do:type="String">ok</status>
```

This way it can be determined, whether the actual data is wrong, or the presentation (stylesheets).

10.5 Troubleshooting commands

The following commands are advanced sub-commands of midas.sh (or midas.bat), which are not to be used normally. In troubleshooting situations, however, they may be helpful. Check with product support before using them.

10.5.1 Packing up support data

The ***support*** sub-command is an important support tool. Use it if you run into any problems with CVP. It collects and packages up all important configuration and log files into one *.zip file. The file location is the MIDAS Installation directory.

Example:

```
# <MIDAS_HOME>/midas.sh support
[...]
```

```
support.zip:
     [echo] collecting support information ...
     [echo] collecting version info ...
     [echo] collecting installed files ...
     [echo] collecting Java properties ...

[propertyfile] Creating new property file:
            /opt/midas30/work/env_20061106_1711.properties

intern.unix_uname:
     [echo] checking uname ...

intern.unix_env:
     [echo] collecting environment ...

intern.win_env:

intern.linux_procinfo:

intern.checksum_check:
     [echo] checking checksums ...
     [echo] creating support zip ...
      [zip] Building zip: /opt/midas30/support_20061106_1711.zip
     [echo] cleaning up ...
   [delete] Deleting 4 files from /opt/midas30/work
     [echo] send the file /opt/midas30/support_20061106_1711.zip to support
```

The file name will contain the date and time when the command has been executed.

The zip file contains:

- Checksum information for every individual file which is part of the installed software package. This way, product support can find out, which file has been modified to quickly determine possible reasons for problems.
- All core config files and installed licenses (the <MIDAS_HOME>/conf directory)
- All core log files (the <MIDAS_HOME>/logs directory)
- All WebApp log and configuration files (<MIDAS_HOME>/webapp/*/work/webapp/WEB-INF directory)

The product support team will usually ask you to send this file via email as a first step to investigate a problem.

10.5.2 Running ANT tasks

Using the ***ant*** sub-command, integrated ANT tasks can be started similarly to what happens inside CVP generally.

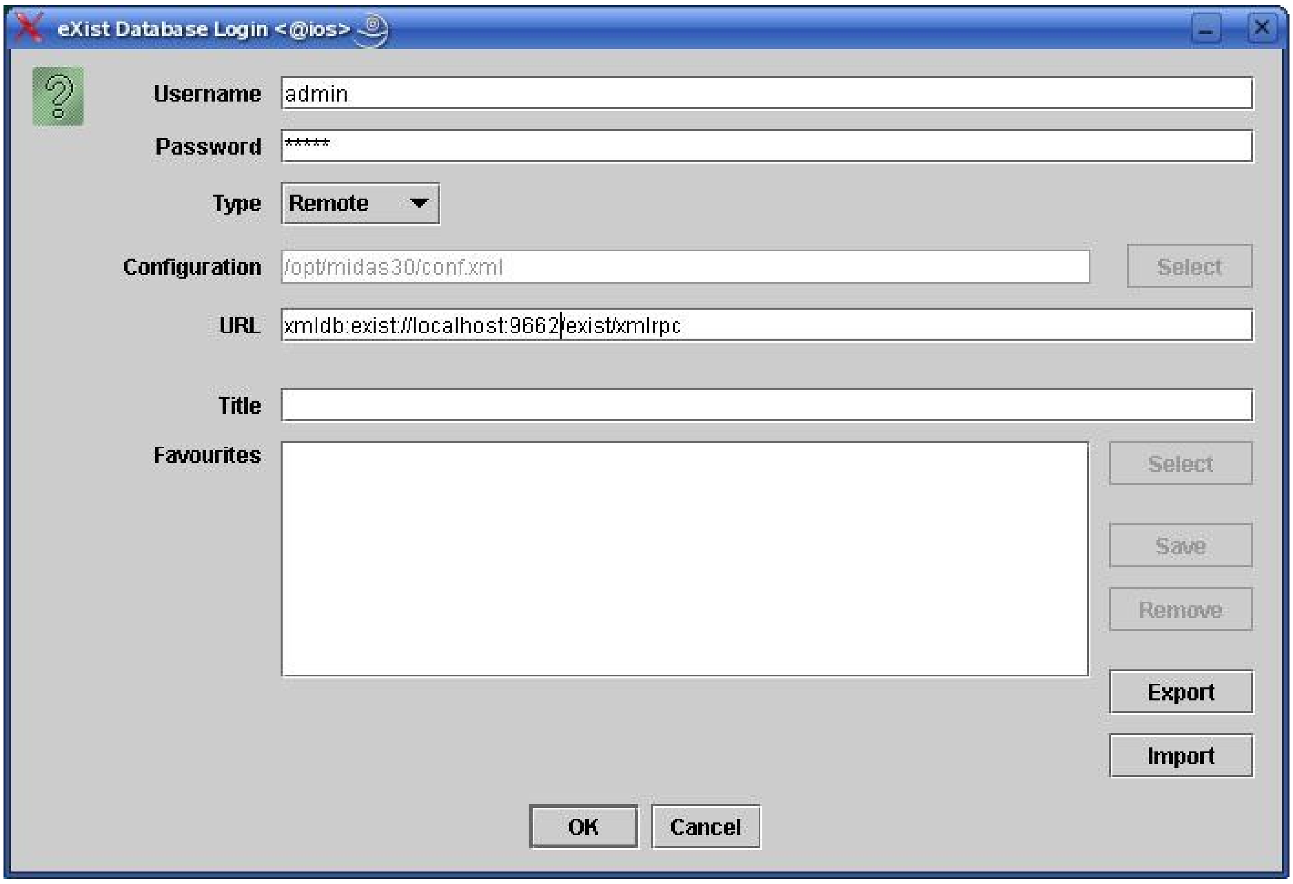
However, none of these tasks are for normal customer use.

10.5.3 Reviewing the XML DB

The ***xmldb*** sub-command starts a graphical XML DB management console. This can be used for administration of the bundled XML database that contains user model and BackEnd configuration data.

Normally there should be no need to use this, except for troubleshooting purposes. It is strongly recommended not to modify anything unless instructed by product support. Viewing at data is harmless though.

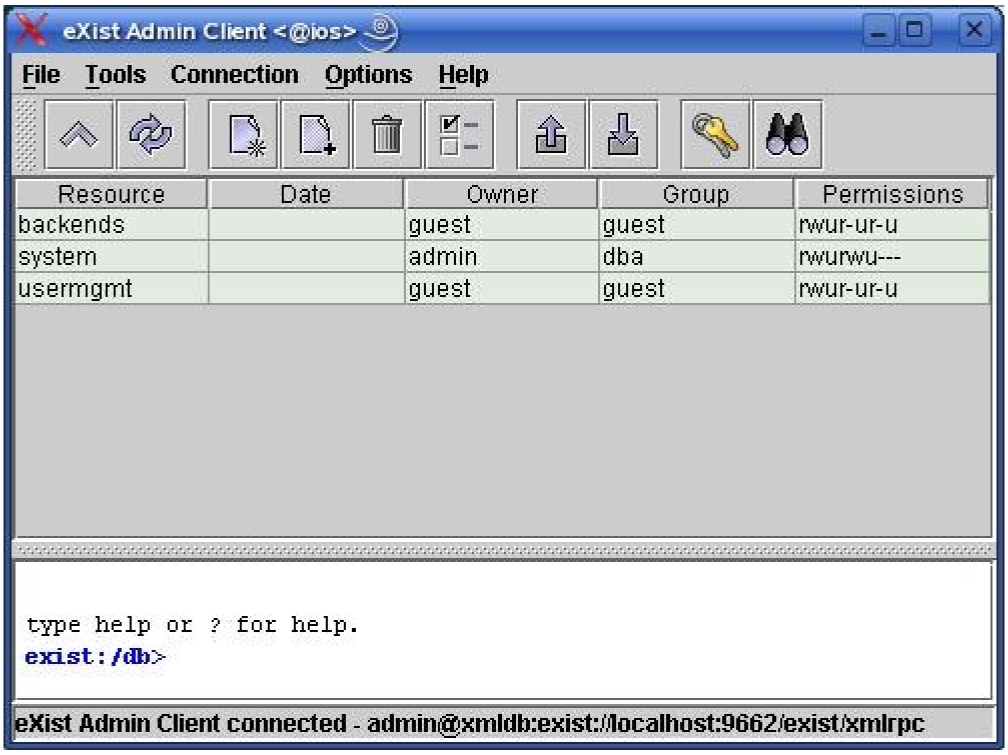
After running the `midas.sh xmldb` command a connection screen appears:



eXist Database Login <@ios>

Username	admin
Password	*****
Type	Remote
Configuration	/opt/midas30/conf.xml
URL	xmldb:exist://localhost:9662/exist/xmlrpc
Title	
Favourites	

Specify the correct URL (must contain the host name of the WebApp or localhost if started on the WebApp system and the according port, 9662 by default). Then, the actual GUI appears (if used remotely, check for the correct settings of the X DISPLAY and permission on the remote X Server):

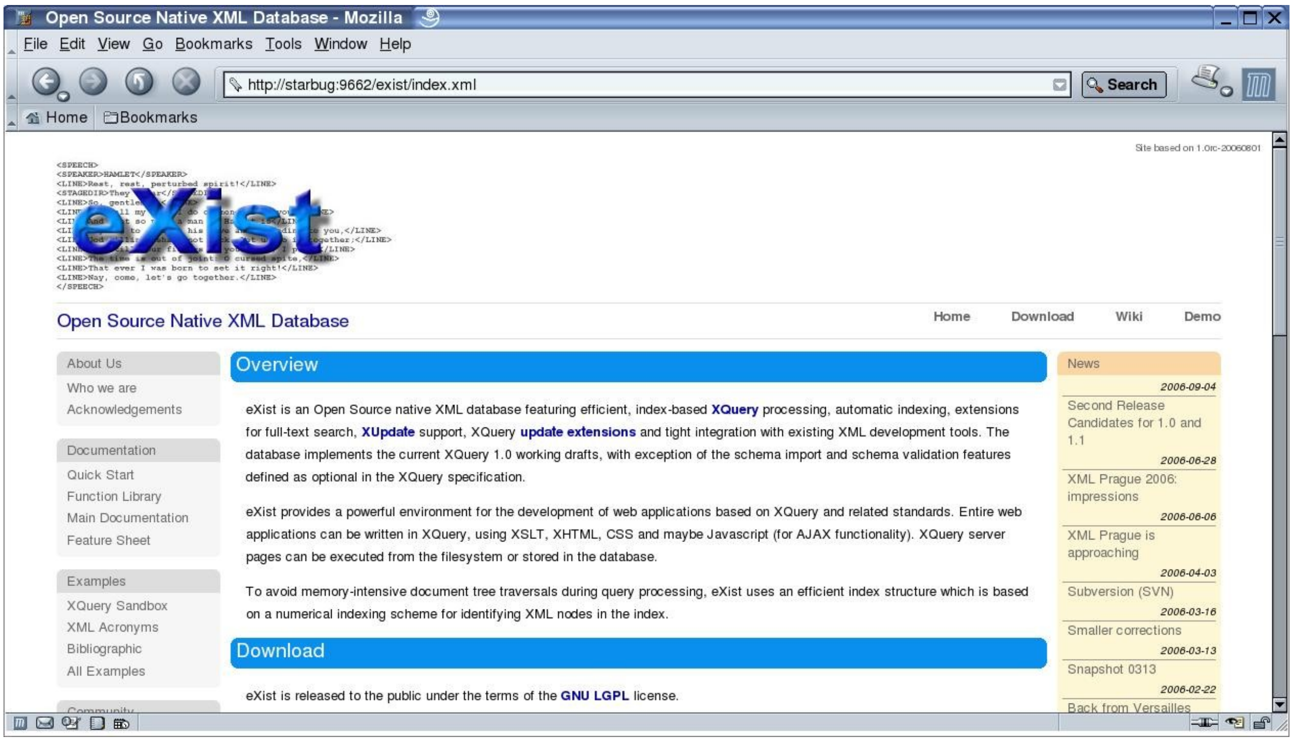


Resource	Date	Owner	Group	Permissions
backends		guest	guest	rwur-ur-u
system		admin	dba	rwurwu---
usermgmt		guest	guest	rwur-ur-u

```
type help or ? for help.
exist:/db>
```

eXist Admin Client connected - admin@xmldb:exist://localhost:9662/exist/xmlrpc

The XML DB also supports an HTML interface. In your web browser connect to http:<WebApp>:9662/exist which should look like this:



Overview

eXist is an Open Source native XML database featuring efficient, index-based XQuery processing, automatic indexing, extensions for full-text search, XUpdate support, XQuery update extensions and tight integration with existing XML development tools. The database implements the current XQuery 1.0 working drafts, with exception of the schema import and schema validation features defined as optional in the XQuery specification.

eXist provides a powerful environment for the development of web applications based on XQuery and related standards. Entire web applications can be written in XQuery, using XSLT, XHTML, CSS and maybe Javascript (for AJAX functionality). XQuery server pages can be executed from the filesystem or stored in the database.

To avoid memory-intensive document tree traversals during query processing, eXist uses an efficient index structure which is based on a numerical indexing scheme for identifying XML nodes in the index.

Download

eXist is released to the public under the terms of the GNU LGPL license.

News

2006-09-04
Second Release Candidates for 1.0 and 1.1

2006-06-28
XML Prague 2006: impressions

2006-06-06
XML Prague is approaching

2006-04-03
Subversion (SVN)

2006-03-16
Smaller corrections

2006-03-13
Snapshot 0313

2006-02-22
Back from Versailles

You need to log on (same log on data as with the native GUI above) and can then browse the contents.

For further details, see http://exist.sourceforge.net.

10.5.4 Posting a XML request

The ***post*** sub-command is a support tool to submit a request in the form of a XML document to the CVP server. Don't use this command unless instructed by product support.

10.5.5 Check component status

The ***servicemix*** sub-command displays information about installed Servicemix JBI components (binding components and services) and deployed service assemblies with their contained service units during runtime.

This way you can check whether all required CVP adapters have been successfully started. Missing service assemblies indicate a problem with the corresponding adapter. In this case review the log files `servicemix.log` and `midas.log` for more details about failed adapters.

This sub-command is not needed during normal operation but can be used for troubleshooting. It is non-destructive and can be used without damaging anything.

The following example shows parts of the output of the *servicemix* sub-command:

```
# midas.sh servicemix

servicemix:

c:
     [echo]  list-binding-components
     [echo]  Prints information about the binding components installed in
servicemix.
     [echo]     host=ios.bes-intern.com
     [echo]     port=9660
[...]

list-service-engines:
     [echo]  list-service-engines
     [echo]  Prints information about all of the Service Engines in
Servicemix.
     [echo]     host=ios.bes-intern.com
     [echo]     port=9660
[...]

list-service-assemblies:
     [echo]  list-service-assemblies
     [echo]  list deployed Service Assemblies in Servicemix.
     [echo]     host=ios.bes-intern.com
     [echo]     port=9660
     [echo]     state=
     [echo]     componentName=
     [echo]     serviceAssemblyName=
[jbi:list-service-assemblies] <?xml version='1.0'?>
[jbi:list-service-assemblies] <service-assembly-info-list
```

```
xmlns='http://java.sun.com/xml/ns/jbi/service-assembly-info-list'
version='1.0'>
[jbi:list-service-assemblies]   <service-assembly-info name='midas-ovocore'
state='Started'>
[jbi:list-service-assemblies]     <description>MIDAS OVO/U Core
components</description>
[jbi:list-service-assemblies]     <service-unit-info name='midas-ovodispatch'
state='Started' deployed-on='servicemix-eip'>
[jbi:list-service-assemblies]       <description>MIDAS OVO/U Request
Dispatching</description>
[jbi:list-service-assemblies]     </service-unit-info>
[jbi:list-service-assemblies]   </service-assembly-info>
[jbi:list-service-assemblies]   <service-assembly-info name='midas-webapp'
state='Started'>
[jbi:list-service-assemblies]     <description>MIDAS Web
Application</description>
[jbi:list-service-assemblies]     <service-unit-info name='midas-jetty'
state='Started' deployed-on='servicemix-lwcontainer'>
[jbi:list-service-assemblies]       <description>MIDAS Web
Application</description>
[jbi:list-service-assemblies]     </service-unit-info>
[jbi:list-service-assemblies]   </service-assembly-info>
[jbi:list-service-assemblies]   <service-assembly-info name='midas-beovobase'
state='Started'>

[...]
```

The first blocks `list-binding-components` and `list-service-engines` must be always present including some additional details.

The last block `list-service-assemblies` shows all deployed service assemblies. This list must match the set of files in `<MIDAS_HOME>/deploy`. If a service assembly is present in the `deploy` directory but not listed in the output of `midas.sh servicemix` as Started, it has failed to start. In this case inspect the CVP log files for details. Particularly the file `<MIDAS_HOME>/logs/servicemix.log` will contain entries like the following:

```
ERROR - 2006-12-20 13:03:16,369 |
AutoDeploymentService.updateArchive(308) | Failed to update Service
Assembly: midas-wapam
java.lang.Exception: <?xml version="1.0" encoding="UTF-8"?>
[...]
nested exception is java.lang.UnsatisfiedLinkError: no jpam in
java.library.path</loc-message>
[...]
```

In this example, the adapter `midas-wapam` (the PAM authentication adapter) has failed to start because the native library `libjpam.so` could not be found.

10.5.6 Re-initialize the XML DB

The ***init*** sub-command clears and re-initializes the XML DB.

CAUTION: All CVP user information and registered BackEnds are lost!

However, this may be a helpful command, if after an installation the XML DB has not been initialized correctly, or it has been corrupted later on.

10.5.7 Determining checksums

The checksum sub-command generates checksums of all installed files belonging to the CVP product. This command is automatically executed during the CVP installation and should not be used by the end user.

The generated checksums are stored in `<MIDAS_HOME>/checksums`. To detect modifications of original product files, the checksums stored here can be compared against the current checksum of such a file. This may be a helpful aid to isolate problems.

10.6 Licensing problems

CVP product licenses consist of a set of XML files, which are signed to ensure, that nobody can modify them. The license files and the signature files must reside in

```
# ls -l <MIDAS_HOME>/conf/licenses/BES/*
BES/MIDAS_BACKEND:
total 20
dr-xr-xr-x  2 root root 232 2006-09-26 20:58 .
drwxrwxr-x  8 root root 264 2006-09-26 20:58 ..
-r-xr-xr-x  1 root root 985 2006-09-21 13:36 1158838494359.license
-r-xr-xr-x  1 root root  46 2006-09-21 13:36 1158838494359.signature
-r--r--r--  1 root root 355 2006-09-26 20:58 install.info
-r--r--r--  1 root root 443 2006-09-26 20:58 install.key
-r--r--r--  1 root root  46 2006-09-26 20:58 install.signature
[...]
```

In addition, the following file is needed containing the public key to verify the signatures:

```
# ls -l <MIDAS_HOME>/conf/licenses/BES/key.pub
-r--r--r--  1 root root 444 2006-09-21 13:34 BES/key.pub
```

Finally, all licensed components, must have a setting like the following in their configuration:

```
licenses.holderName=blue elephant systems GmbH
```

which must match the XML tag <lic:holder> in the actual license files:

```
<?xml version='1.0'?>
<!-- License file, generated automatically. DO NOT MODIFY! -->
<lic:licenses xmlns:lic="http://blue-elephant-systems.com/midas/license/1.0">
  <lic:license>
    <lic:id>1163494049956</lic:id>
    <lic:holder>
      <lic:id>BES</lic:id>
      <lic:name>blue elephant systems GmbH</lic:name>
    </lic:holder>
    [...]
    <lic:level>Instant-On</lic:level>
    <lic:expiration>+60d</lic:expiration>
    [...]
```

You cannot change the license files themselves without breaking the signature, but make sure that your company name appears in all places correctly. Also verify, the other license attributes like level and expiration.

10.7 CVS integration problems

The following hints assumes the standard CVS repository being used locally on the WebApp. If a remote repository is used, similar steps apply.

In any case, also review the general CVP log files, particularly `midas.log` and `vcs.log` for related entries.

10.7.1 Using the CVS command line

CVS is originally a command line tool. The one and only command to be used is `cvs`, which is documented in a man page. Additional information is available under http://www.cvshome.org.

To check local operation, create a temporary directory, e.g. /tmp/cvstest. Change to this directory and initialize the test sandbox. Perform all these steps as the `midas` OS user as created during CVP installation:

```
# su midas
$ mkdir /tmp/cvstest
$ cd /tmp/cvstest
$ cvs -d :local:/opt/midas30/data/repository co .
[...]
```

This must be successful assuming the CVS repository is locally in `/opt/midas30/data/repository`. If not, evaluate the displayed errors.

NOTE: The same works identically on Windows.

10.7.2 Verify CVS configuration

If CVS works correctly standalone, verify the CVS settings in the CVP configuration. The following entries must exist in `conf/wrapper.conf`:

```
# grep -i cvs /opt/midas30/conf/wrapper.conf
[...]
wrapper.java.additional.6=-Djavacvs.multiple_commands_warning=false
wrapper.java.additional.7=-DEnv-CVS_EXE=%MIDAS_HOME%/bin/cvs
# wrapper.java.additional.12=-DcvsClientLog="%MIDAS_HOME%/logs/nbcvs.log"
[...]
```

In addition, the CVS repository used by CVP is configured in `conf/vcs.properties`:

```
# cat /opt/midas30/conf/vcs.properties
cvs.cvsRoot=:local:/opt/midas30/data/repository
licenses.holderName=blue elephant systems GmbH
```

10.7.3 Check file permissions

Both the CVS sandbox and the CVS repository must be owned by the OS user `midas` (or at least the user must be able to perform all read and write operations) as shown in the following example:

```
# /opt/midas30 [ios:tge]> ls -la work/cvs data/repository

data/repository:
total 1
drwxrwxr-x  4 midas sys    96 2006-12-18 18:58 .
drwxrwxr-x  8 root  root  208 2006-12-01 14:50 ..
drwxrwxr-x  3 midas users  80 2006-12-18 18:58 config
drwxrwxr-x  3 midas sys   824 2006-12-18 18:58 CVSROOT

work/cvs:
total 0
drwxr-xr-x  4 midas root   96 2006-12-18 18:58 .
drwxr-xr-x  6 root  root  240 2006-12-06 23:15 ..
drwxr-xr-x  5 midas users 120 2006-12-18 18:58 latestRO
drwxr-xr-x  3 midas users  80 2006-12-18 19:00 tmp
```

The CVS sandboxes used by CVP is located in `<MIDAS_HOME>/work/cvs`. There are two kinds of sandboxes:

- ./latestRO This sandbox exists, once created, for ever and is used exclusively for HEAD elements (i.e. the latest checked-in version). Once such an element is checked out to this sandbox, it remains in the sandbox. Subsequent updates only occur if needed, i.e. if a later revision exists. This is the most common case and increases performance assuming that only a small sub-set of all elements change during a certain time.
- ./tmp/<someID> These sandboxes are used temporarily for check-in operations, possibly involving sticky tags. To ensure consistent data in CVS, these sandboxes will be removed when performing the next such CVS operation. This means, that the temporary sandbox from the previous CVS operation remains. This information can be used to verify the involved files, their contents and CVS properties.

10.7.4 Trace the CVS adapter

In the `wrapper.conf` file, uncomment the following entries. Make sure, that the number (in the example below 8) is in sequence with the previous variables. Then, re-start the CVP WebApp:

```
# vi <MIDAS_HOME>/conf/wrapper.conf
[...]
wrapper.java.additional.8=-DcvsClientLog="%MIDAS_HOME%/logs/nbcvs.log"
wrapper.java.additional.8.stripquotes=TRUE
[...]
```

After performing some CVS operation, the following files must exist in the logs directory:

```
# cat <MIDAS_HOME>/logs/nbcvs.log.*
[...]
E cvs server: scheduling file
`config/sylt_server/ovo/msggroup/Apache_Admin.xml' for addition
Checked-in config/sylt_server/ovo/msggroup/
C:/midas30/data/repository/config/sylt_server/ovo/msggroup/Apache_Admin.xml
/Apache_Admin.xml/0///
E cvs server: use 'cvs commit' to add this file permanently
ok
E cvs server: Examining .
E cvs server: Examining config
E cvs server: Examining config/sylt_server
E cvs server: Examining config/sylt_server/ovo
E cvs server: Examining config/sylt_server/ovo/msggroup
M RCS file:
C:/midas30/data/repository/config/sylt_server/ovo/msggroup/Apache_Admin.xml,v
M done
M Checking in config/sylt_server/ovo/msggroup/Apache_Admin.xml;
M
C:/midas30/data/repository/config/sylt_server/ovo/msggroup/Apache_Admin.xml,v
<--  Apache_Admin.xml
M initial revision: 1.1
M done
Checked-in config/sylt_server/ovo/msggroup/
C:/midas30/data/repository/config/sylt_server/ovo/msggroup/Apache_Admin.xml
/Apache_Admin.xml/1.1///
[...]
```

containing the CVS protocol data (the `*.in` file contains data received from the CVS server, the `*.out` file data sent).

10.8 HTTPS problems

If experiencing problem with HTTPS communication, please verify the configuration as described in chapter 9.7.2 Using HTTPS.

The configuration must contain

- HTTPS ports
- keystore and truststore files

Use the `keytool` command to display the contents of the key- and truststore:

```
# keytool -v -list -keystore keystore_endpoint.jks
Enter keystore password:  password

Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
Alias name: ios_server
Creation date: Nov 21, 2006
```

```
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=ios_server
Issuer: CN=ios_server
Serial number: 4562bfeb
Valid from: Tue Nov 21 09:59:23 CET 2006 until: Mon Feb 19 09:59:23 CET 2007
Certificate fingerprints:
         MD5:  29:09:CB:98:AF:65:87:F4:7D:CD:9D:C7:4B:75:A3:F2
         SHA1: 37:FF:73:3A:1E:BF:EF:FD:EE:56:C7:6C:AB:BB:9C:A7:B8:11:94:83
```

10.9 File transfer problems

As described in chapter 4.3.1 FTP vs. SSH communication volume data can be transferred between CVP servers using FTP or SSH. Both mechanisms are selected and configured initially during installation.

For configuration steps regarding FTP transfer and information, what is required, please refer to chapter 9.9.1 Configuring FTP transfer. Similarly, for SSH-related configuration see chapter 7.3 Secure Shell (SSH).

Generally, basic network connectivity must be correct. Check name resolution, routing etc. using standard OS commands like `netstat`, `ping`, `nslookup`, ...

When initiating a data transfer in the CVP GUI, either pull or push access can be selected. This is important from the firewall perspective – in either case the network connection will be initiated from the opposite peer and firewall rules must be configured appropriately.

10.9.1 The clipboard

All transfer between CVP servers (using FTP or SSH, push or pull) copies data from the clipboard of the source server to the clipboard of the target server. For details about the structure of the clipboard see chapter 4.6 Temporary storage.

Make sure that the contents of the clipboard can be accessed. This is particularly important for SSH transfer as explained in chapter 4.3.1 FTP vs. SSH communication because the source or target data will be accessed as OS user `midas`. Therefore, the clipboard should be owned by the user `midas` and look as in the following example:

```
# ls -l /opt/midas30 [ios:tge]> ll data/clipboard

total 0
drwxrwxr-x 2 midas root  48 2006-12-01 14:49 .
drwxrwxr-x 8 root  root 208 2006-12-01 14:50 ..
drwxr-xr-x 3 root  sys   96 Dec 19 20:32 opccfg_1166556736227_bla
drwxr-xr-x 3 root  sys   96 Dec 19 21:12 opccfg_1166559172367_Rel-1.3
[...]
drwxr-xr-x 3 midas users 96 Dec 19 21:38 opccfg_1166559543001_ssh-test
[...]
```

In this example, two download operations have been performed earlier which have created the sub-directory structures named `opccfg_<ID>_<comment>`, where the `<ID>` is a unique ID and `<comment>` is the comment entered by the user in the GUI when performing the download.

The third element in the example above showing the comment “ssh-test” has been received from another CVP server via SSH transfer. Therefore it is owned by the OS user `midas`.

10.9.2 Data up- and download

All download and transfer operations create files in the clipboard, which can be reviewed for troubleshooting. This also applies to failing upload operations – verify whether the data to be uploaded is correct.

Up- and download operations are executed as ANT tasks. General error logging and tracing data of ANT tasks will be written into the log file `<MIDAS_HOME>/logs/task.log`. In addition, the output of each executed ANT task will be logged into an individual session log file under `<MIDAS_HOME>/logs/task`. In case of trouble review the file with an associated time stamp.

10.9.3 FTP transfer

The most important steps are to make sure that the right ports are configured and network connectivity exists.

To review the FTP server port on the target CVP server, inspect the file `<MIDAS_HOME>/conf/ftpd.xml` as described in chapter 9.9.1 Configuring FTP transfer.

Make sure that the address (or if it is a host name, the address this name resolves to) and port specified as server address/port can be reached from other CVP servers.

Check the host names and IP addresses listed in the `ip.properties` file – these are the ones which are allowed to access this FTP server. Make sure, all names are resolvable with name service and all desired CVP servers acting as FTP client are registered.

Check the file `<MIDAS_HOME>/conf/user.properties`:

```
# cat <MIDAS_HOME>/conf/user.properties
[...]
FtpServer.user.file.userpassword=94C6DE8E7F0A3881C14ED4BBBCB514AB
FtpServer.user.admin.downloadrate=0
FtpServer.user.midas.homedirectory=/opt/midas30
FtpServer.user.anonymous.writepermission=false
FtpServer.user.admin.enableflag=true
FtpServer.user.anonymous.downloadrate=4800
FtpServer.user.midas.userpassword=94C6DE8E7F0A3881C14ED4BBBCB514AB
FtpServer.user.anonymous.uploadrate=4800
FtpServer.user.admin.homedirectory=/opt/midas30
FtpServer.user.file.uploadrate=0
FtpServer.user.file.homedirectory=/opt/midas30/data
FtpServer.user.admin.idletime=300
FtpServer.user.midas.idletime=300
FtpServer.user.admin.userpassword=21232F297A57A5A743894A0E4A801FC3
FtpServer.user.admin.writepermission=true
FtpServer.user.anonymous.idletime=300
FtpServer.user.admin.uploadrate=0
FtpServer.user.anonymous.userpassword=D41D8CD98F00B204E9800998ECF8427
FtpServer.user.midas.enableflag=true
FtpServer.user.file.downloadrate=0
FtpServer.user.anonymous.homedirectory=./res/home
FtpServer.user.midas.writepermission=false
FtpServer.user.file.writepermission=true
```

Do not change anything, just make sure the file exists and looks like the one above. Also verify, that the contained directories like `/opt/midas30` match your set-up.

For testing, the standard `ftp` command on OS level can be used to connect a target FTP server.

Both the FTP server and client are built into the CVP server and run under the same identity, i.e. as root on UNIX systems and SYSTEM on Windows. Therefore no problems accessing data should occur.

10.9.4 SSH transfer

If experiencing problems with SSH transfer, first review chapter 7.3 Secure Shell (SSH). Here the necessary configuration and verification steps are explained.

The main steps to verify and test are:

- Is there an `sshd` running on the target SSH server
- Can you connect the SSH server using the `ssh` command on OS level (independently from CVP)
- Does the `$HOME` of the midas user match the CVP installation root (on both the SSH client and server)
  - Does this directory contain a sub-directory .ssh containing all key files
  - Are SSH keys exchanged
- Are the SSH-related entries in `<MIDAS_HOME>/conf/ant/config.xml` correct
- Is the OS user `midas` able to access the source data and to create target data

10.10 Authentication problems

If you cannot log on to CVP, check the following:

- Is the CVP WebApp running on the desired system and does it use the right ports. Verify whether the CVP processes are running and the port configuration. Check whether the ports are allocated using the `netstat` or `lsof` commands.
  Also, if possible, start a web browser locally on the WebApp system and try to connect.
- Can you reach the WebApp system from the basic network perspective. Use the `ping` and `telnet` commands.
- Does the desired CVP user account exist? If an external authentication system is used via PAM or LDAP, the user also must exist in that system.
  Is that CVP user member of least one CVP group? Does that group have at least one CVP user role assigned? If the effective set of objects visible to the user is

empty, the user cannot log on.

10.10.1 PAM integration

If experiencing problems with a configured PAM module, check the following:

- Are there any related entries in the log file `<MIDAS_HOME/logs/usermgmt.log`, for example:

```
DEBUG - 2006-12-20 14:03:49,002 |
UserModelRequestTransformer.transform(?) | rewriting request to
service pam
DEBUG - 2006-12-20 14:03:49,087 | PamServer.authenticate(?) |
Authenticating user admin with PAM service midas ...
DEBUG - 2006-12-20 14:03:49,088 | Pam.authenticate(160) | Debug
mode active.
ERROR - 2006-12-20 14:03:51,126 | PamServer.authenticate(?) |
Authentication of user admin failed: Underlying authentication
service can not retrieve authentication information.
DEBUG - 2006-12-20 14:03:51,163 |
UserMgmtFilter.onMessageExchange(?) | clearing response { DOType:
errorresponse
extra : com.bes.itm.comp.usermgmt.AuthenticationFailedException:
Could not authenticate user admin via PAM. PAM error: Underlying
authentication service can not retrieve authentication
information.
```

- To enable additional tracing in the PAM adapter module, configure the following in `<MIDAS_HOME>/conf/log4j.xml`:

```
<logger name="net.sf.jpam" additivity="false">
 <level value="DEBUG"/>
 <appender-ref ref="usermgmt"/>
</logger>
```

  This will make the PAM module write additional debug statements into the log file `<MIDAS_HOME>/logs/usermgmt.log`.

- Test the authentication method standalone

```
# cd /opt/midas30/support
# export LD_LIBRARY_PATH=/opt/midas30/lib/midas
# /opt/midas30/jre/bin/java -cp .:../work/service-
assemblies/midas-wapam/version_1/sus/servicemix-lwcontainer/midas-
pam/lib/jpam-0.5.jar:../lib/commons-logging-1.0.4.jar
com/bes/itm/comp/usermgmt/TestPam <user name> <password>

**** Starting ...
**** Authenticating user admin ...
**** Authentication done.
**** Success: false
**** Result: Underlying authentication service can not retrieve
authentication information.
```

```
**** Exit.
```

This test class performs a pure PAM authentication of <user name> with <password> and prints the results to stdout.

- Check the PAM configuration as described in chapter 7.4 Authentication software.
- Make sure that all dependencies of the native library `<MIDAS_HOME>/lib/midas/libjpam.so` are satisfied, for example:

```
# ldd <MIDAS_HOME>/lib/midas/libjpam.so
  linux-gate.so.1 =>  (0xffffe000)
  libpam.so.0 => /lib/libpam.so.0 (0x40016000)
  libpam_misc.so.0 => /lib/libpam_misc.so.0 (0x40020000)
  libdl.so.2 => /lib/libdl.so.2 (0x40023000)
  libc.so.6 => /lib/tls/libc.so.6 (0x40027000)
  /lib/ld-linux.so.2 (0x80000000)
```

- Review the UNIX syslog log file, the native PAM library logs messages to syslog, for example:

```
# grep -i pam /var/log/messages
[...]
Dec  1 18:15:41 garlic midas.pam(pam_unix)[25305]: authentication
failure; logname= uid=0 euid=0 tty= ruser= rhost=  user=admin
Dec  1 18:17:40 garlic midas.pam(pam_unix)[25305]: authentication
failure; logname= uid=0 euid=0 tty= ruser= rhost=  user=admin
[...]
```

- To enable tracing the native PAM library, turn on debugging on syslog level (the steps needed may vary for the different UNIX flavors). The related service name is `auth`. To turn on tracing, configure `syslogd` as in the following example:

```
# touch /etc/pam_debug
# vi /etc/syslog.conf
auth.debug      /tmp/pam_auth.log
[...]
```

Make the `syslogd` process re-read its configuration, for example by executing:

```
# kill -HUP `cat /var/run/syslog.pid`
```

The resulting debug output looks as in the following example:

```
# tail -f /tmp/pam_auth.log
Dec 20 15:41:16 ios PAM: pam_start(midas admin)
Dec 20 15:41:16 ios PAM: pam_set_item(1)
Dec 20 15:41:16 ios PAM: pam_set_item(2)
Dec 20 15:41:16 ios PAM: pam_set_item(5)
Dec 20 15:41:16 ios PAM: pam_set_item(6)
Dec 20 15:41:16 ios PAM: pam_authenticate()
Dec 20 15:41:16 ios PAM: load_modules:
/usr/lib/security/hpux32/libpam_unix.so.1
Dec 20 15:41:16 ios PAM: load_function: successful load of
pam_sm_authenticate
Dec 20 15:41:16 ios PAM: pam_get_username(ux)
Dec 20 15:41:16 ios PAM: pam_mapping_in_use()
Dec 20 15:41:16 ios PAM: pam_set_item(6)
Dec 20 15:41:16 ios PAM: pam_acct_mgmt()
Dec 20 15:41:16 ios PAM: load_modules:
/usr/lib/security/hpux32/libpam_unix.so.1
Dec 20 15:41:16 ios PAM: load_function: successful load of
pam_sm_acct_mgmt
Dec 20 15:41:16 ios PAM: pam_get_username(ux)
Dec 20 15:41:16 ios PAM: pam_mapping_in_use()
Dec 20 15:41:16 ios PAM: pam_end(): status = Success
```

10.11 Process crashes

If the JRE process running CVP server crashes, collect the following data and send it to product support:

- The support.zip generated as described above
- The related `hs_err_pid<PID>.log` file

If the crash occurs regularly, also the core file written by the JRE may help. By default, creating core files on UNIX is disabled. To enable the creation of core files, edit the file `<MIDAS_HOME>/bin/server.sh` and comment out the following line:

```
ulimit -c 0
```

Then, if the abort occurs again, save the core file for later evaluation. If a core file exists, it can be analyzed using the OVO utility stacktrace, as shown in the following example (this example applies to an HP-UX Itanium system):

```
# /opt/OV/contrib/OpC/stacktrace /opt/midas30/core
                                 /opt/midas30/jre/bin/IA64N/java
```

This may particularly yield precise information where the problem occured.

11 References

11.1 Directory Structure

The complete CVP product will be installed to the location specified by the installing user. This location is variable (though, it is recommended to keep the default value) and is referred to as <MIDAS_HOME> throughout this document.

This is identical no matter whether installing CVP on UNIX or Windows.

The actual contents changes with the roles, the CVP server is acting in (WebApp and/or BackEnd) and the deployed service assemblies.

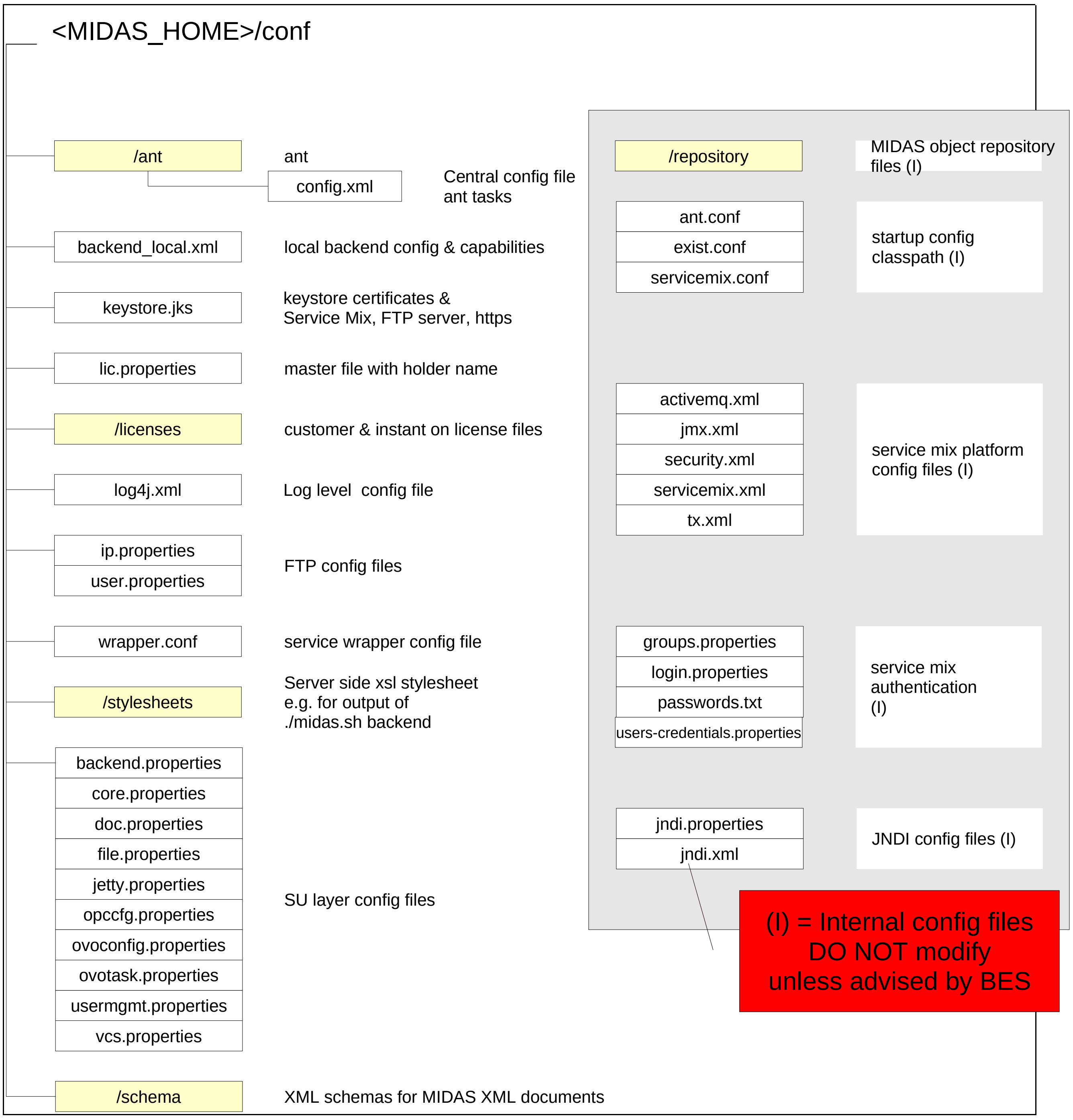
Within <MIDAS_HOME> the main directories and files are as follows:

<MIDAS_HOME>

Midas_InstallLog.log	one of the Installer log files
/Uninstall_MIDAS	contains Uninstaller
/assemblies	available service assemblies
/bin	MIDAS scripts, binaries etc
/components	available JBI components (Java Business Integration)
/conf	config files + License files, layer config files, ant config file in subdirs.
/data	see separate diagram for details
/deploy	actively deployed service assemblies
/docs	3rd party licenses (open source)
/install	actively deployed JBI components
/jre	bundled Java SDK
/libs	shared jar files & native libs
/logs	log files & task log files
midas.bat	central midas script - Windows
midas.sh	central midas script – HP-UX SUN Linux
midas_env.bat	contains environment variables - Windows
midas_env.sh	contains environment variables – HP-UX SUN Linux
midas_server.pid	contains MIDAS server PID at runtime
run.xml	ant file used by midas.sh or midas.bat
/ovo	MIDAS SPI – OVO configuration upload (MIDAS OVO backend only)
/webapps	deployed Webapp (MIDAS Webapp only)
/work	Service Mix deployment files & other temp files (policies etc pp)
/wrapper	Service wrapper to run MIDAS as daemon or service

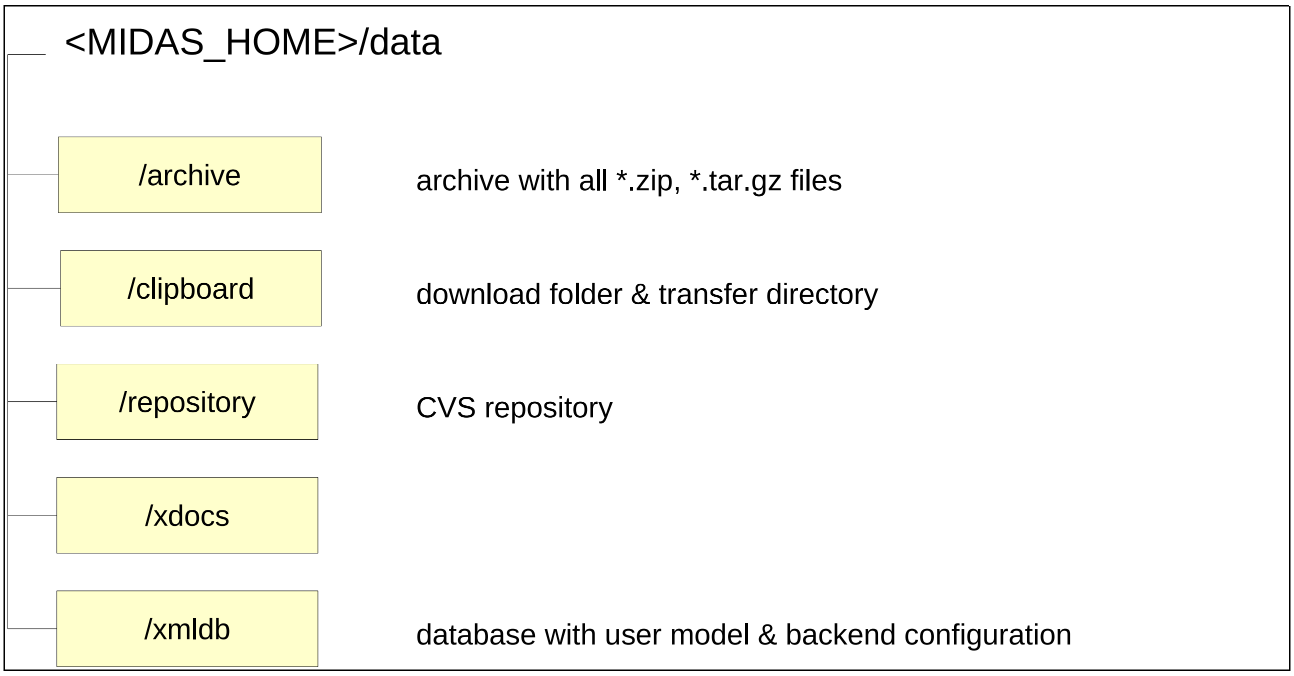
11.1.1 The configuration directory

The directory <MIDAS_HOME>/conf contains various configuration data. See chapter 9 CVP Configuration for details.



11.1.2 The data directory

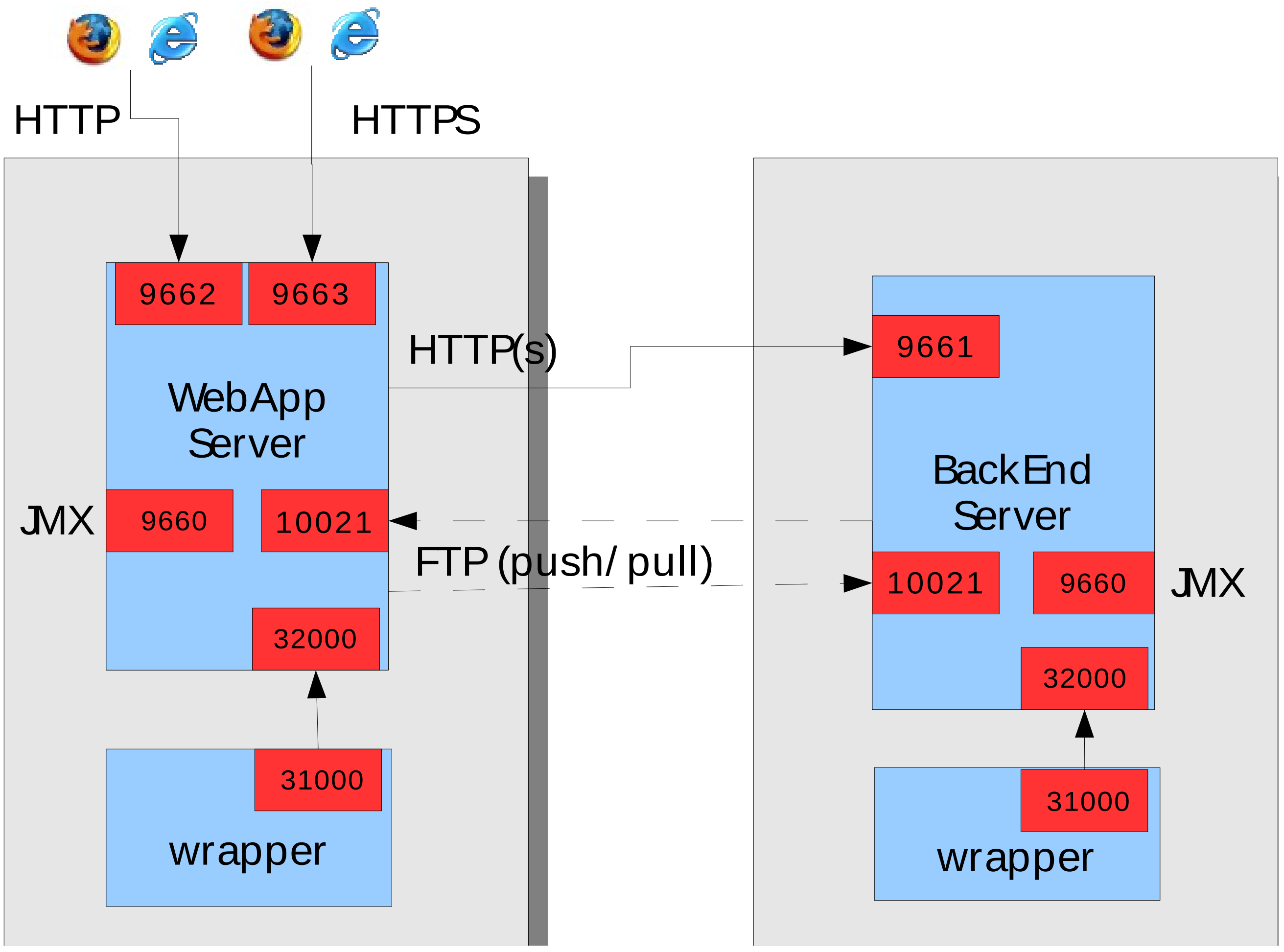
The data directory contains user-specific data and is special in a way that it will not be touched during installation (except some initialization, if not existing yet) and de-installation. This data is usually created during downloads, transfers or when using CVS.



<MIDAS_HOME>/data
/archive
archive with all *.zip, *.tar.gz files
/clipboard
download folder & transfer directory
/repository
CVS repository
/xdocs
/xmldb
database with user model & backend configuration

11.2 Communication

The following picture illustrates the general communication flow between all CVP components. It assumes that the default ports are used (the port numbers used by CVP may be changed by user):



This can be verified by using the `netstat` command, as shown in the following example:

```
# netstat -an | grep LISTEN
[...]
> tcp        0      0  *.9660                 *.*          LISTEN
> tcp        0      0  *.9662                 *.*          LISTEN
> tcp        0      0  *.9663                 *.*          LISTEN
> tcp        0      0  127.0.0.1.32000        *.*          LISTEN
> tcp        0      0  *.10021                *.*          LISTEN
> tcp        0      0  *.9661                 *.*          LISTEN
[...]
```

If FTP is not used in passive mode, in addition port 10020 will be used temporarily during FTP transfers.

The default ports used by CVP are:

Port(s)	Purpose	Random number	Local use only

9660	JMX console, for troubleshooting only		x
9661	Server port to which CVP WebApps connect to. HTTP or HTTPS.		
9662	CVP HTTP WebApp port to which the user connects with the web browser		
9663	Same as 9662, but HTTPS.		
10021	FTP server port for file transfers with other CVP servers		
32000	Communication between the `wrapper` process and the JRE running in a `java` process e.g. stop, dump etc.	x	x

Ports marked as random above will be allocated as available. The actual numbers may vary.

For information regarding configuration of all used ports, see chapter 9.7.1 Changing default ports.

12 1-Page Installation Guide

- login as user 'root' (Windows: Administrator privileges) & export DISPLAY if needed
- `export IATEMPDIR=/dir/with/enough/diskspace` if /tmp too small (~600mb needed)
- Start CVP installer: `#./install.bin`
- Install set: FULL (Webapp & OVO Backend), OVO Backend only, Webapp only (separate Webapp and Backend installations are recommended).
- Installation directory ________________________________ (~600mb needed)
- Local Server Identifier
  Server Hostname: ________________________________
  (full DNS recommended,**cluster: use cluster hostname!**
  Local Server Identifier: ________________________________
  (e.g. shortname_server, ovo1_server)
  Description (optional): ________________________________
- Port Settings
  Server port (default 9661): ____________
  JMX port (default 9660): ____________
  Use HTTPS: [ ]
- FTP or SCP transfer.
  Use FTP [ ]FTP port: ________
  Use SSH [ ] SSH Binary location: ________________
- Automatic Startup [ ] integrated into ovstart/stop
  [ ] integrate in OS boot sequence (/etc/init.d or Windows service)
  [ ] manual startup
- Oracle OVO configuration (default detection is generally correct), but **check** the Oracle DB name and the Openview DB intance name!
  You need to provide passwords for: opc_op, opc_adm users
- Self Monitoring Settings (BackEnd only). Name and port of initial WebApp.
- Location of CVS binary: ______________________
  Location of CVS repository: __________________
  Is CVSNT used as CVS software? [ ]
- Paper format (A4 or US Letter)
- WebApp ports (user connects via his browser to this port of the WebApp)
  HTTP (default 9662): _______________
  HTTPS (default 9663): _______________
- Pre-Installation summary. After that the actual CVP installation starts.
- Important Note. Shows again all used configuration settings: server identifier, ports
  Write these down here and print this page!