



Open Build Service



Open Build Service: Administrator Guide

by Karsten Keil

Publication Date: 09/07/2016

SUSE LLC

10 Canal Park Drive

Suite 200

Cambridge MA 02141


USA

<https://www.suse.com/documentation> 

Copyright © 2016 B1 Systems GmbH

Copyright © 2006– 2016 Novell, Inc. and contributors. All rights reserved.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled “GNU Free Documentation License”.

For Novell trademarks, see the Novell Trademark and Service Mark list <http://www.novell.com/company/legal/trademarks/tmlist.html> . Linux* is a registered trademark of Linus Torvalds. All other third party trademarks are the property of their respective owners. A trademark symbol (®, ™ etc.) denotes a Novell trademark; an asterisk (*) denotes a third party trademark.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither Novell, Inc., SUSE LINUX Products GmbH, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

Contents

About this Guide vi

1 Installation and configuration 1

1.1 Planning 1

Resource planning 1

1.2 Simple installation 2

Backend installation 2 • Frontend installation 4 • Online configuration 7

1.3 Worker farm 10

1.4 Distributed setup 10

1.5 Monitoring 13

Endpoint Checks 14 • Common Checks 14 • Other Checks 16

2 Overview filesystem 18

2.1 Configuration files 18

Frontend Configuration 18 • Backend Configuration 25

2.2 Log files 46

Frontend 46 • Backend 47

2.3 /src/obs tree 47

build directory 48 • db directory 48 • diffcache directory 49 • events directory 49 • info directory 49 • jobs directory 49 • log directory 49 • projects directory 49 • remotecache directory 50 • repos directory 50 • repos_sync directory 50 • run directory 50 • sources directory 50 • trees directory 51 • upload directory 51 • workers directory 51

2.4 Meta data 51

OBS revision control 51 • Project meta data 53 • Package meta data 54 • Attribute meta data 55 • Job files 55

3 Administration 57

3.1 Tools 57

obs_admin 57 • osc 60

3.2 Managing build targets 64

Interconnect 64 • Importing distributions 65

3.3 Source Services 68

Using services for validation 69 • Different Modes when using services 69 • How are source service definitions stored 70 • Dropping a source service again 71

3.4 Dispatch Priorities 71

/build/_dispatchprios API call 72 • dispatch_adjust array 73

3.5 Publisher Hooks 74

Publisher Hooks configuration 74 • Example Publisher Scripts 76

3.6 Unpublisher Hooks 78

Unpublisher Hooks configuration 78 • Example Unpublisher Scripts 80

3.7 User and group management 81

User and group roles 82 • Standalone user and group database 82 • Proxy mode 82 • LDAP/Active Directory 83 • Authentication methods 90

3.8 Backup 91

4 Troubleshooting 92

4.1 General hints 92

4.2 Debug frontend problems 93

A GNU Licenses 94

A.1 GNU General Public License 94

About this Guide


This book is part of the official Open Build Service documentation. These books are considered to contain only reviewed content, establishing the reference documentation of OBS.

These books are not considered to be focused on a special OBS version. They are also not a replacement of the documentation inside of the openSUSE wiki. But content from the wiki may get consolidated and included in these books.

Furthermore these books get written by the OBS community, please check the chapter how to work on these books. We request esp. experienced users and administrators to join and to contribute to these books. It is not required to be a very good or even native English speaker, because we rely on community editors to improve the language.

1 Installation and configuration

1.1 Planning

For testing an own OBS instance and for small setups like only packaging some scripts from your administrators into RPMS and creating proper installation sources from them, the ready to use obs-server appliance images are the easiest way. You can download them from <http://openbuildservice.org/download/> .

If you want to use the OBS for your Linux software development with many packages, projects and users you should build your own installation. Depending on how many users and projects you have and for how many architectures you want to build packages you can split up the backend (called partitioning) and have separate hosts for the frontend and database.

But for most installations it is still OK to run everything but workers one host with enough resources.

For flexibility and if you want some kind of high availability it is recommended to use virtualisation for the different components.

1.1.1 Resource planning

Normally for a small to middle installation a setup with everything except workers on one host is sufficient. You should have separate /srv volume for the backend data, XFS as filesystem is best choice.

For each scheduler architecture you should add 4 GB RAM and one CPU core. For each build distribution you should add at least 50GB disk space per architecture.

A medium instance with about 50 users can easily run on a machine with 16GB RAM 4 cores and 1 TB storage. The storage of course depend on the size of your projects and how often you have new versions.

For bigger installations you can use separate networks for backend communication, workers and frontend.

The reference installation on **build.opensuse.org** with lot of users, distributions runs on a partitioned setup with:

- a mysql cluster as database
- api-server: 16GB RAM 4 cores 50GB disk
- separate binary backends (scheduler, dispatcher, reposerver, publisher, warden)
- source server 11 GB RAM, 4 cores, 3 TB disk (RAM used mainly for caching)
- main backend: 62 GB RAM (oversized), 16TB disk
- lot of workers (see - <https://build.opensuse.org/monitor> ↗)

For build time and performance the count and performance of available worker hosts more important as the remaining parts.

1.2 Simple installation

Simple installation means, all OBS services (except workers) running on the same machine.

Important

It is very important that you read the **README.SETUP** file coming with your OBS version and follow the instructions there, because here maybe changes to this version.

Before you start the installation of the OBS, you should make sure that your hosts have the correct full qualified hostname and DNS is working and can resolve all names.

1.2.1 Backend installation

The backend hosts all sources and built packages. It also schedules the jobs. You need to install the "obs-server" package for this. You need to check the `/usr/lib/obs/server/BSConfig.pm` file, but the defaults should be good enough for the simple case.

You can control the different backend components via systemctl. Basically you can enable/disable the service during booting the system and start/stop/restart it in a running system. Have

a look at the <https://www.freedesktop.org/software/systemd/man/systemctl.html#Commands> man page for more information. For instance to restart the repository server use

```
systemctl restart obsrepserver.service
```

TABLE 1.1: SERVICE NAMES

Component	Service Name	Remarks
Source Server	obssrcserver.service	
Repository	Server obsrepserver.service	
Source	Services obsservice.service	
Download	obsdodup.service	since 2.7
Delta Storage	obsdeltastore.service	since 2.7
Scheduler	obsscheduler.service	
Dispatcher	obsdispatcher.service	
Publisher	obspublisher.service	
Signer	obssigner.service	
Warden	obswarden.service	

The sequence in the table reflects the start sequence, you need to enable the services with

```
systemctl start <name>
```

first and then you can start them:

```
systemctl start obssrcserver.service
systemctl start obsrepserver.service
systemctl start obsservice.service
systemctl start obsdodup.service
systemctl start obsdeltastore.service
systemctl start obsscheduler.service
systemctl start obsdispatcher.service
systemctl start obspublisher.service
systemctl start obssigner.service
systemctl start obswarden.service
```



Warning

The commands start services which are accessible from the outside. Do not do this on a system connected to an untrusted network or make sure to block the ports with a firewall.

1.2.2 Frontend installation

You need to install the "obs-api" package for this and a MySQL server.

1.2.2.1 MySQL setup

Make sure that the mysql server is started on every system reboot (use "insserv mysql" for permanent start). You should run `mysql_secure_installation` and follow the instructions.

Create the empty production databases:

```
# mysql -u root -p
mysql> create database api_production;
mysql> quit
```

You should use an own MySQL user (e.g. "obs") for the OBS access:

```
# mysql -u root -p
mysql> create user 'obs'@'%' identified by 'TopSecretPassword';
mysql> create user 'obs'@'localhost' identified by 'TopSecretPassword';
mysql> GRANT all privileges ON api_production.*
      TO 'obs'@'%', 'obs'@'localhost';
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Configure your MySQL user and password in the "production" section of the api config: **`/srv/www/obs/api/config/database.yml`**

Example:

```
# MySQL (default setup). Versions 4.1 and 5.0 are recommended.
#
# Get the fast C bindings:
#   gem install mysql
```

```
# (on OS X: gem install mysql -- --include=/usr/local/lib)
# And be sure to use new-style password hashing:
# http://dev.mysql.com/doc/refman/5.0/en/old-client.html

production:
  adapter: mysql2
  database: api_production
  username: obs
  password: TopSecretPassword
  encoding: utf8
  timeout: 15
  pool: 30
```

Now populate the database

```
cd /srv/www/obs/api/
sudo RAILS_ENV="production" rake db:setup
sudo RAILS_ENV="production" rake writeconfiguration
sudo chown -R wwwrun.www log tmp
```

Now you are done with the database setup.

1.2.2.2 Apache setup

Now we need to configure the webserver. By default, you can reach the familiar web user interface and also api both on port 443 speaking https. Repositories can be accessed via http on port 82 (once some packages are built). An overview page about your OBS instance can be found behind *'http://localhost'*.

The obs-api package comes with a apache vhost file, which does not need to get modified when you stay with these defaults: */etc/apache2/vhosts.d/obs.conf*

Install the required packages via

```
zypper in obs-api apache2 apache2-mod_xforward rubygem-passenger-apache2 memcached
```

Add the follwing apache modules in */etc/sysconfig/apache2*:

```
APACHE_MODULES="... passenger rewrite proxy proxy_http xforward headers
socache_shmcb"
```

Enable SSL in */etc/sysconfig/apache2* via

```
APACHE_SERVER_FLAGS="SSL"
```

For production systems you should order official SSL certificates. For testing follow the instructions to create a self signed SSL certificate:

```
mkdir /srv/obs/certs
openssl genrsa -out /srv/obs/certs/server.key 1024
openssl req -new -key /srv/obs/certs/server.key \
    -out /srv/obs/certs/server.csr
openssl x509 -req -days 365 -in /srv/obs/certs/server.csr \
    -signkey /srv/obs/certs/server.key -out /srv/obs/certs/server.crt
cat /srv/obs/certs/server.key /srv/obs/certs/server.crt \
    > /srv/obs/certs/server.pem
```

To allow the usage of https api in webui code you need to trust your certificate as well:

```
cp /srv/obs/certs/server.pem /etc/ssl/certs/
c_rehash /etc/ssl/certs/
```

1.2.2.3 API configuration

Check and edit `/srv/www/obs/api/config/options.yml`

If you change the hostnames/ips of the api, you need to adjust **frontend_host** accordingly. If you want to use LDAP, you need to change the LDAP settings as well. Look at the [Section 3.7, "User and group management"](#) for details. You will find examples and more details in the [Section 2.1, "Configuration files"](#).

It is recommended to enable

```
use_xforward: true
```

as well here.

Afterwards you can start the OBS web api and make it permanent via

```
systemctl enable apache2
systemctl start apache2

systemctl enable obsapidelayed.service
systemctl start obsapidelayed.service
```

```
systemctl enable memcached.service
systemctl start memcached.service
```

Now you have your own empty instance running and you can do some online configuration steps.

1.2.3 Online configuration

To customize the OBS instance you may need to configure some settings via the OBS API and Web user interface.

First you should change the password of the Admin account, for this you need first login as user Admin in the WebUI with the default password "opensuse". Click on the Admin link (right top of the page), here you can change the password.

After you changed the Admin password, you need to setup osc to use the Admin account for more changes. Here an example:

```
osc -c ~/.obsadmin_osc.rc -A https://api.testobs.org
```

Follow the instructions on the terminal.



Warning

The password is stored in clear text in this file by default, so you need to give this file restrictive access rights, only read/write access for your user should be allowed. **osc** allows to store the password in other ways (in keyrings for example), refer to the osc documentation for this.

Now you can checkout the main configuration of the OBS:

```
osc -c ~/.obsadmin_osc.rc api /configuration >/tmp/obs.config
cat /tmp/obs.config
<configuration>
  <title>Open Build Service</title>
  <description>  <p class="description">
    The <a href="http://openbuildservice.org"> Open Build Service (OBS)</a>
    is an open and complete distribution development platform that provides a
    transparent
```

```

    infrastructure for development of Linux distributions, used by openSUSE, MeeGo
    and other distributions.
    Supporting also Fedora, Debian, Ubuntu, RedHat and other Linux distributions.
</p>
<p class="description">
    The OBS is developed under the umbrella of the <a href="http://
www.opensuse.org">openSUSE project<
    /a>. Please find further informations on the <
    a href="http://wiki.opensuse.org/openSUSE:Build_Service">openSUSE Project
wiki pages</a>.
    </p>

<p class="description">
    The Open Build Service developer team is greeting you. In case you use your OBS
productive
    in your facility, please do us a favor and add yourself at <
    a href="http://wiki.opensuse.org/openSUSE:Build_Service_installations">
this wiki page</a>. Have fun and fast build times!
    </p>
</description>
<name>private</name>
<download_on_demand>on</download_on_demand>
<enforce_project_keys>off</enforce_project_keys>
<anonymous>on</anonymous>
<registration>allow</registration>
<default_access_disabled>off</default_access_disabled>
<allow_user_to_create_home_project>on</allow_user_to_create_home_project>
<disallow_group_creation>off</disallow_group_creation>
<change_password>on</change_password>
<hide_private_options>off</hide_private_options>
<gravatar>on</gravatar>
<cleanup_empty_projects>on</cleanup_empty_projects>
<disable_publish_for_branches>on</disable_publish_for_branches>
<admin_email>unconfigured@openbuildservice.org</admin_email>
<schedulers>
    <arch>armv7l</arch>
    <arch>i586</arch>
    <arch>x86_64</arch>
</schedulers>
</configuration>

```

You should edit this file according to your preferences, then sent it back to the server:

```
osc -c ~/.obsadmin_osc.rc api /configuration -T /tmp/obs.config
```

If you want to use an interconnect to another OBS instance to reuse the build targets you can do this as Admin via the Web UI or create a project with a **remoteurl** tag (see [Section 2.4.2, "Project meta data"](#))

```
<project name="openSUSE.org">
  <title>openSUSE.org Project</title>
  <description>
    This project refers to projects hosted on the Build Service
    ...

    Use openSUSE.org:openSUSE:12.3 for example to build against the
    openSUSE:12.3 project as specified on the opensuse.org Build Service.
  </description>
  <remoteurl>https://api.opensuse.org/public</remoteurl>
</project>
```

You can create the project using a file with the above content with **osc** like this:

```
osc -c ~/.obsadmin_osc.rc meta prj openSUSE.org -F /tmp/openSUSE.org.meta
```

You also can import binary distribution, see [Section 3.2.2, "Importing distributions"](#) for this.

The OBS has a list of available distributions used for build. This list is displayed to user, if they are adding repositories to their projects. This list can be managed via the API path `/distributions`

```
osc -c ~/.obsadmin_osc.rc api /distributions > /tmp/distributions.xml
```

Example distributions.xml file:

```
<distributions>
  <distribution vendor="SUSE" version="SLE-12-SP1" id="137">
    <name>SLE-12-SP1</name>
    <project>SUSE:SLE-12-SP1</project>
    <reponame>SLE-12-SP1</reponame>
    <repository>standard</repository>
    <link>http://www.suse.com/</link>
    <icon url="https://static.opensuse.org/distributions/logos/suse-SLE-12-8.png"
width="8" height="8"/>
    <icon url="https://static.opensuse.org/distributions/logos/suse-SLE-12-16.png"
width="16" height="16"/>
    <architecture>x86_64</architecture>
  </distribution>
```

```
</distributions>
```

You can add your own distributions here and update the list on the server:

```
osc -c ~/.obsadmin_osc.rc api /distributions -T /tmp/distributions.xml
```

1.3 Worker farm

To not burden your OBS backend daemons with the unpredictable load package builds can produce (think someone builds a monstrous package like LibreOffice) you should not run OBS workers on the same host as the rest of the backend daemons.

Important

You backend need to be configured to use the correct hostnames for the repo and source server and the ports need to be reachable by the workes. Also the IP addresses of the workers need to be allowed to connect the services. (look at the `/usr/lib/obs/server/BSConfig.pm::ipaccess` array).

You can deploy workers quite simply using the worker appliance. Or install a minimum system plus the obs-worker package on the hardware.

Edit the `/etc/sysconfig/obs-server` file, at least **OBS_SRC_SERVER**, **OBS_REPO_SERVERS** and **OBS_WORKER_INSTANCES** need to be set. More details in the [Section 2.1, "Configuration files"](#).

start the worker:

```
systemctl enable obsworker  
systemctl start obsworker
```

1.4 Distributed setup

All OBS backend daemons can also be started on individual machines in your network. Also the frontend webserver and the MySQL server can run on different machines. Especially for large scale OBS installations this is the recommended setup.

A setup with partitioning is very similar to the steps of the simple setup. Here we are only mention the differences to the simple setup.



Note

You need to make sure that the different machines can communicate via the network, it is very recommended to use a separate network for this to isolate it from the public part.

On all backend hosts you need to install the obs-server package. On the frontend host you need to install the obs-api package.



Important

Only one source server instance can be exist on a single OBS installation.

The binary backend can be splitted on project level, this is called partitioning.

One one partition following services needs to be configured and run:

1. repserver
2. schedulers
3. dispatcher
4. warden
5. publisher

You do not need to share any directories on filesystem level between the partions.

Here some example for partitioning:

1. A main partition for everything not in the others (host mainbackend)
2. A home partition for all home projects of the users (host homebackend)
3. A release partition for released software projects (host releasebackend)

The configuration is done in the backen config file `/usr/lib/obs/server/BSConfig.pm`. Most parts of the file can be shared between the backends.

Here the important parts of the mainbackend of our testobs.org installation:

```
...
my $hostname = Net::Domain::hostfqdn() || 'localhost';
# IP corresponding to hostname (only used for $ipaccess); fallback to localhost
  since inet_aton may fail to resolve at shutdown.
my $ip = quotemeta inet_ntoa(inet_aton($hostname) || inet_aton("localhost"));

my $frontend = 'api.testobs.org'; # FQDN of the WebUI/API server if it's not
  $hostname

# If defined, restrict access to the backend servers (bs_repserver, bs_srcserver,
  bs_service)
our $ipaccess = {
  '127\..*' => 'rw', # only the localhost can write to the backend
  "^$ip" => 'rw',    # Permit IP of FQDN
  "10.20.1.100" => 'rw',    # Permit IP of srcsrv.testobs.org
  "10.20.1.101" => 'rw',    # Permit IP of mainbackend.testobs.org
  "10.20.1.102" => 'rw',    # Permit IP of homebackend.testobs.org
  "10.20.1.103" => 'rw',    # Permit IP of releasebackend.testobs.org
  '10.20.2.*' => 'worker', # build results can be delivered from any client in the
    network
};

# IP of the WebUI/API Server (only used for $ipaccess)
if ($frontend) {
  my $frontendip = quotemeta inet_ntoa(inet_aton($frontend) ||
    inet_aton("localhost"));
  $ipaccess->{$frontendip} = 'rw' ; # in dotted.quad format
}

# Change also the SLP reg files in /etc/slp.reg.d/ when you touch hostname or port
our $srcserver = "http://srcsrv.testobs.org:5352";
our $reposerver = "http://mainbackend.testobs.org:5252";
our $serviceserver = "http://service.testobs.org:5152";
#
our @reposervers = (
  http://mainbackend.testobs.org:5252,
  http://homebackend.testobs.org:5252,
  http://releasebackend.testobs.org:5252
);

# you can use different ports for worker connections
our $workersrcserver = "http://w-srcsrv.testobs.org:5353";
```

```

our $workerreposerver = "http://w-mainbackend.testobs.org:5253";
...
our $partition = 'main';
#
# this defines how the projects are split. All home: projects are hosted
# on an own server in this example. Order is important.
our $partitioning = [
    'home:' => 'home',
    'release' => 'release'
    '.*'     => 'main',
];
our $partitionservers = {
    'home' => 'http://homebackend.testobs.org:5252',
    'release' => 'http://releasebackend.testobs.org:5252',
    'main' => 'http://mainbackend.testobs.org:5252',
};
...

```

On the other partition server you need to change "**our \$reposerver**", "**our \$workerreposerver**" and "**our \$partition**".

On all partition servers you need to start:

```

systemctl start obsrepserver.service
systemctl start obscheduler.service
systemctl start obsdispatcher.service
systemctl start obspublisher.service
systemctl start obswarden.service

```

On the worker machines you should set of repo servers in the **OBS_REPO_SERVERS** variable. You can also define workers with a subset of the repo servers to prioritize partitions.

1.5 Monitoring

In this chapter you will find some general monitoring instructions for the Open Build Service. All examples are based on Nagios plugins, but the information provided should be easily adaptable for other monitoring solutions.

1.5.1 Endpoint Checks

1.5.1.1 HTTP Checks - Check if the HTTP server responds

This check will output a critical if the HTTP server with ip address 172.19.19.19 (-I 172.19.19.19) listening on port 80 (-p 80) does not answer and output a warning if the HTTP return code is not 200. The servername that will be used is server (-H server) which is important if different virtual hosts are listening on the same port.

```
check_http -H server -I 172.19.19.19 -p 80 -u http://server
```

The same check, but this time it will check a ssl enabled HTTP server.

```
check_http -S -H server -I 172.19.19.19 -p 443 -u https://server
```

It is also possible to check the presence of a certain string in the HTTP response. In this case it will check for the string *Source Service Server*.

```
check_http -s "Source Service Server" -S -H server -I 172.19.19.19 -p 5152
```

Open Build Service HTTP endpoints that should be checked:

1. Web Interface / API: port 443
2. Repository Server: port 82
3. Package Repository Server: port 5252
4. Source Repository Server: port 5352
5. Source Service Server: port 5152

1.5.2 Common Checks

This is a list of common checks that should be run on each individual server.

1.5.2.1 Disk Space - Check for available disk space

This check will output a warning if less than 10 percent disk space is available (-w 10) and output a critical if less than 5 percent disk space are available (-c 5). It will check all filesystems except filesystems with type *none* (-x none).

```
check_disk -w 10 -c 5 -x none
```

1.5.2.2 Memory Usage - Check for available memory

This check will output a warning if less than 10 percent memory is available (-w 10) and output a critical if less than 5 percent memory is available (-c 5). OS caches will be counted as free memory (-C) and it will check the available memory (-f). `check_mem.pl` is not a standard Nagios plugin and can be downloaded at <https://exchange.nagios.org/>.

```
check_mem.pl -f -C -w 10 -c 5
```

1.5.2.3 NTP - Check Date and Time

This check will compare the local time with the time provided by the NTP server `pool.ntp.org` (-H `pool.ntp.org`). It will output a warning if the time differs by 0.5 seconds (-w 0.5) and output a critical if the time differs by 1 seconds (-c 1).

```
check_ntp_time -H pool.ntp.org -w 0.5 -c 1
```

1.5.2.4 Ping - Check that the server is alive

This plugin checks if the server responds to a ping request and it will output a warning if the respond time exceeds 200ms or 30 percent package loss (-w 200.0,30%) and output a critical if the respond time exceeds 500ms or 60 percent package loss.


```
check_icmp -H server -w 200.0,30% -c 500.0,60%
```

1.5.2.5 Load - Check the load on the server

This check will output a warning if the load value exceeded 7.0 in the last minute, 6.0 in the last 5 minutes or 5.0 in the last 15 minutes (-w 7.0,6.0,5.0). It will output a critical if the load value exceeded 12.0 in the last minute, 8.0 in the last 5 minutes or 6.0 in the last 15 minutes (-c 12.0,8.0,6.0).

```
check_load -w 7.0,6.0,5.0 -c 12.0,8.0,6.0
```

1.5.2.6 Disk Health - Check the health of local hard disks

This check is only relevant on physical systems with local storage attached to it. It will check the disk status utilizing the S.M.A.R.T interface and it will output a critical if any of the S.M.A.R.T values exceeds critical limits. `check_smartmon` is not a standard Nagios plugin and can be downloaded at <https://exchange.nagios.org/> .

```
check_smartmon --drive /dev/sda --drive /dev/sdb
```

1.5.3 Other Checks

1.5.3.1 MySQL - Check that the MySQL database is responding

This check will check that the MySQL database server is running and that the database *api_production* is available.

```
check_mysql -H localhost -u nagios -p xxxxxx -d api_production
```

MySQL Databases to check:

1. `api_production`
2. `mysql`

1.5.3.2 Backup Status - Check that a valid backup is available

It is always advisable to check that the last backup run was successful and a recent backup is available. The check itself depends on the Backup solution that is used.

2 Overview filesystem

2.1 Configuration files

2.1.1 Frontend Configuration

The Frontend is configured with 3 files:

- /srv/www/obs/api/config/database.yml
- /srv/www/obs/api/config/options.yml
- /etc/apache2/vhosts.d/obs.conf

2.1.1.1 database.yml

This file has the information needed to access the database. It contain credentials for the database access and should be only readable by root and the group running the webserver (www).

The file has settings for the production, development and test ruby environment, for production systems only the production section is important.

Example production section

```
production:
  adapter: mysql2
  database: api_production
  username: obsapiuser
  password: topsecret
  encoding: utf8
  timeout: 15
  pool: 30
```

TABLE 2.1: DATABASE CONFIGURATION KEYWORDS

keyword	Description	Remarks
adapter	Database driver	only mysql databases are supported

keyword	Description	Remarks
database	Database name	do not change !
username	mysql user name	database user, not a system user
password	password for this user	clear text
encoding	codetable	
timeout	wait time in milliseconds	
pool	number of open connections per thread	
socket	path to the mysql socket	same host only
host	Ip address or hostname of the mysql server	for remote servers
port	port number of the mysql server	for remote servers

2.1.1.2 options.yml

The configuration file `/srv/www/obs/api/config/options.yml` is the default configuration file for the Open Build Service WebUI and API. It contains configuration parameters for example for backend connections and connection to the API. Important are the configurations for source and frontend hosts. The configuration for LDAP authentication is also located in this file.



Note

More and more configurations will be moved to the database and do not longer exist in this file. The database configuration can be accessed via the API `/configuration` path.

TABLE 2.2: **OPTION.YML CONFIGURATION ITEMS**

Config item	Description	Values <u>default</u>	Remarks
use_xforward	Use mod_xforward module	<u>true</u> false	apache only, should be true

Config item	Description	Values <u>default</u>	Remarks
use_nginx_redirect	Use X-Accel-Redirect	<u>/</u> <u>internal_redirect</u>	Nginx only
min_votes_for_rating	Minimum votes for a rating	integer <u>3</u>	
response_schema_validation	Set to true to verify XML reponses comply to the schema	true <u>false</u>	test/debug option
source_host	backend source server host	<u>localhost</u>	
source_port	backend source server port	integer <u>5352</u>	
source_protocol	backend source server protocol	<u>http</u> , <u>https</u>	
frontend_host	Frontend host	<u>localhost</u>	
frontend_port	Frontend port	integer <u>443</u>	
frontend_protocol	Frontend protocol	http <u>https</u>	
external_frontend_host	External Frontend host		if your users access the hosts through a proxy or different name
external_frontend_port	External Frontend port	integer <u>443</u>	
external_frontend_protocol	External Frontend protocol	http <u>https</u>	
extended_backend_log	Extended backen log	<u>true</u> false	test/debug option
proxy_auth_mode:	turn proxy mode on/off	<u>:off</u> :on	see LDAP section

Config item	Description	Values <u>default</u>	Remarks
proxy_auth_test_user	Test user	<u>coolguy</u>	test/debug option
proxy_auth_test_email	Email of Test user	<u>coolguy@</u> <u>example.com</u>	test/debug option
global_write_through	if set to false, the API will only fake writes to backend	<u>true</u> false	test/debug option
auto_cleanup_after_days	not longer used	<u>30</u>	moved to / configuration API
errbit_api_key	API key of the application		test/debug option
errbit_host	installation of errbit.com a Ruby error catcher		test/debug option
errbit_api_key	API key of the application		test/debug option
ldap_mode:	OBS LDAP mode on/off	<u>:off</u> :on	see LDAP setion

Example options.yml

```
#
# This file contains the default configuration of the Open Build Service
# API.
#

# Make use of mod_xforward module in apache
use_xforward: true

# Make use of X-Accel-Redirect for Nginx.
# http://kovyrin.net/2010/07/24/nginx-fu-x-accel-redirect-remote
#use_nginx_redirect: /internal_redirect

# Minimum count of rating votes a project/package needs to # be taken in
```

```
# account
# for global statistics:
min_votes_for_rating: 3

# Set to true to verify XML reponses comply to the schema
response_schema_validation: false

# backend source server
source_host: localhost
source_port: 5352
#source_protocol: https

# api access to this instance
frontend_host: localhost
frontend_port: 443
frontend_protocol: https
# if your users access the hosts through a proxy (or just a different name,
# use this to
# overwrite the settings for users)
#external_frontend_host: api.opensuse.org
#external_frontend_port: 443
#external_frontend_protocol: https

extended_backend_log: true

# proxy_auth_mode can be :off, :on or :simulate
proxy_auth_mode: :off

# ATTENTION: If proxy_auth_mode is :on, the frontend takes the user
# name that is coming as headervalue X-username as a
# valid user does no further authentication. So take care...
proxy_auth_test_user: coolguy
proxy_auth_test_email: coolguy@example.com

# set this to enable auto cleanup requests after the given days
auto_cleanup_after_days: 30

#schema_location

#version

# if set to false, the API will only fake writes to backend (useful in
# testing)
# global_write_through: true
```

```
# see
# http://colszowka.herokuapp.com/2011/02/22/setting-up-your-custom-hoptoad-notifier-
endpoint-for-free-using-errbit-on-heroku
#errbit_api_key: api_key_of_your_app
#errbit_host: installation.of.errbit.com
```

2.1.1.3 Apache vhost obs.conf

The apache configuration depends on the apache version and what extra options are used, so please use the documentation of the apache version you are using.

Here as example the standard configuration used by the appliance: Apache vhost example

```
Listen 82
# May needed on old distributions or after an update from them.
#Listen 443

# Passenger defaults
PassengerSpawnMethod "smart"
PassengerMaxPoolSize 20
#RailsEnv "development"

# allow long request urls and being part of headers
LimitRequestLine 20000
LimitRequestFieldSize 20000

# Just the overview page
<VirtualHost *:80>
    # just give an overview about this OBS instance via static web page
    DocumentRoot "/srv/www/obs/overview"

    <Directory /srv/www/obs/overview>
        Options Indexes
        Require all granted
    </Directory>
</VirtualHost>

# Build Results
<VirtualHost *:82>
    # The resulting repositories
    DocumentRoot "/srv/obs/repos"
```

```

    <Directory /srv/obs/repos>
        Options Indexes FollowSymLinks
        Require all granted
    </Directory>
</VirtualHost>

# OBS WEBUI & API
<VirtualHost *:443>
    ServerName api

    # General setup for the virtual host
    DocumentRoot "/srv/www/obs/api/public"
    ErrorLog /srv/www/obs/api/log/apache_error.log
    TransferLog /srv/www/obs/api/log/apache_access.log

    PassengerMinInstances 2
    PassengerPreStart https://api

    SSLEngine on

    # SSL protocols
    # Supporting TLS only is adequate nowadays
    SSLProtocol all -SSLv2 -SSLv3

    # SSL Cipher Suite:
    # List the ciphers that the client is permitted to negotiate.
    # We disable weak ciphers by default.
    # See the mod_ssl documentation or "openssl ciphers -v" for a
    # complete list.
    SSLCipherSuite ALL:!aNULL:!eNULL:!SSLv2:!LOW:!EXP:!MD5:@STRENGTH

    SSLCertificateFile /srv/obs/certs/server.crt
    SSLCertificateKeyFile /srv/obs/certs/server.key

    <Directory /srv/www/obs/api/public>
        AllowOverride all
        Options -MultiViews

        # This requires mod_xforward loaded in apache
        # Enable the usage via options.yml
        # This will decrease the load due to long running requests a lot
        (unloading from rails stack)
        XForward on

```

```

        Require all granted
    </Directory>

    SetEnvIf User-Agent ".*MSIE [1-5].*" \
        nokeepalive ssl-unclean-shutdown \
        downgrade-1.0 force-response-1.0

    CustomLog /var/log/apache2/ssl_request_log    ssl_combined

    # from http://guides.rubyonrails.org/asset_pipeline.html
    <LocationMatch "^/assets/.*$">
        Header unset ETag
        FileETag None
        # RFC says only cache for 1 year
        ExpiresActive On
        ExpiresDefault "access plus 1 year"
    </LocationMatch>

    SetEnvIf User-Agent ".*MSIE [1-5].*" \
        nokeepalive ssl-unclean-shutdown \
        downgrade-1.0 force-response-1.0

    ## Older firefox versions needs this, otherwise it wont cache anything over
    SSL.
    Header append Cache-Control "public"

</VirtualHost>

```

2.1.2 Backend Configuration

The Backend is configured with 2 files:

- /etc/sysconfig/obs-server - a shell script used for workers and the OBS start scripts
- /usr/lib/obs/server/BSConfig.pm - a perl script defining some global variables

2.1.2.1 `/etc/sysconfig/obs-server`

This script is used to setup the basic pathes and the worker. the most important settings are the **OBS_SRC_SERVER** and **OBS_REPO_SERVERS** and the **OBS_WORKER_INSTANCES**.

TABLE 2.3: **OBS-SERVER VARIABLES**

Variable	Description	Values <u>default</u>	Remarks
OBS_BACKENDCODE_DIR	Path to the backend scripts	/usr/lib/obs/server/	
OBS_RUN_DIR	communication directory base	/srv/obs/run	
OBS_LOG_DIR	logging directory	/srv/obs/log	
OBS_BASE_DIR	base directory	/srv/obs	
OBS_API_AUTOSETUP	Automatically setup api and webui	yes <u>no</u>	appliance only, will overwrite config files
OBS_SRC_SERVER	source server host	localhost:5352	only one
OBS_REPO_SERVERS	repository server host	localhost:5252	maybe a list
OBS_WORKER_INSTANCES	number of build instances	integer <u>0</u>	
OBS_WORKER_INSTANCE_NAMES	names of the workers		space separated list
OBS_WORKER_DIRECTORY	worker base directory		
OBS_WORKER_PORTBASE	The base for port numbers used by worker	integer <u>0</u>	0 OS assign number
OBS_WORKER_JOBS	Number of parallel compile jobs	integer <u>1</u>	

Variable	Description	Values <u>default</u>	Remarks
OBS_WORKER_TEST_MODE	Run in test mode	yes <u>no</u>	
OBS_WORKER_HOST LABELS	one or more labels for the build host		may used by constraints
OBS_USE_SLP	Register in SLP server	<u>yes</u> no	
OBS_CACHE_DIR	cache directory for downloaded packages		
OBS_CACHE_SIZE	package cache size		in MB
OBS_WORKER_NICE_LEVEL	nice level of running workers	<u>18</u>	
OBS_VM_TYPE	VM type	auto xen kvm lxc zvm emulator: \$arch none	
OBS_VM_KERNEL	Set kernel used by worker	<u>none</u> (/boot/ vmlinuz)	KVM option
OBS_VM_INITRD	initrd used by worker	<u>none</u> (/boot/ vmlinuz)	KVM option
OBS_VM_DISK_AUTOSSETUP _ROOT_FILESIZE	Autosetup disk size	<u>4096</u>	in MB
OBS_VM_DISK_AUTOSSETUP _SWAP_FILESIZE	Autosetup swap size	<u>1024</u>	on MB
OBS_VM_DISK_AUTOSSETUP _FILESYSTEM	Filesystem used with autosetup	<u>ext3</u>	
OBS_VM_DISK_AUTOSSETUP _MOUNT_OPTIONS	Special mountoptions		
OBS_VM_USE_TMPFS	Enable build in memory	yes <u>no</u>	requires much memory

Variable	Description	Values <u>default</u>	Remarks
OBS_INSTANCE_MEMORY	Memory allocated for a VM	<u>512</u>	
OBS_STORAGE_AUTOSETUP	storage auto configuration	yes <u>no</u>	may destroy disk content
OBS_SETUP_WORKER_PARTITIONS	LVM via obsstoragesetup	take_all <u>use_obs_vg</u> none	may destroy disk content
OBS_WORKER_CACHE_SIZE	LVM partition for cache size		
OBS_WORKER_ROOT_SIZE	LVM partition for root size		
OBS_WORKER_SWAP_SIZE	LVM partition for swap size		
OBS_WORKER_BINARIES_PROXY	proxy service for caching binaries		
OBS_ROOT_SSHD_KEY_URL	ssh pub key to allow root user login		for mass deployment
OBS_WORKER_SCRIPT_URL	URL to the initial script		

For workers the settings could be declared in the */etc/buildhost.config* file as well.

```
#
# NOTE: all these options can be also declared in /etc/buildhost.config on each
# worker differently.
#

## Path:      Applications/OBS
## Description: The OBS backend code directory
## Type:      string
## Default:   ""
## Config:    OBS
#
```

```

# An empty dir will lead to the fall back directory, typically /usr/lib/obs/server/
#
OBS_BACKENDCODE_DIR=""

## Path:      Applications/OBS
## Description: The base for OBS communication directory
## Type:      string
## Default:   ""
## Config:    OBS
#
# An empty dir will lead to the fall back directory, typically /srv/obs/run
#
OBS_RUN_DIR="/srv/obs/run"

## Path:      Applications/OBS
## Description: The base for OBS logging directory
## Type:      string
## Default:   ""
## Config:    OBS
#
# An empty dir will lead to the fall back directory, typically /srv/obs/log
#
OBS_LOG_DIR="/srv/obs/log"

## Path:      Applications/OBS
## Description: The base directory for OBS
## Type:      string
## Default:   ""
## Config:    OBS
#
# An empty dir will lead to the fall back directory, typically /srv/obs
#
OBS_BASE_DIR=""

## Path:      Applications/OBS
## Description: Automatically setup api and webui for OBS server, be warned, this
               will replace config files !
## Type:      ("yes" | "no")
## Default:   "no"
## Config:    OBS
#
# This is usually only enabled on the OBS Appliance
#
OBS_API_AUTOSETUP="yes"
#

```

```

# NOTE: all these options can be also declared in /etc/buildhost.config on each
worker differently.
#

## Path:      Applications/OBS
## Description: define source server host to be used
## Type:      string
## Default:   ""
## Config:    OBS
#
# An empty setting will point to localhost:5352 by default
#
OBS_SRC_SERVER=""

## Path:      Applications/OBS
## Description: define repository server host to be used
## Type:      string
## Default:   ""
## Config:    OBS
#
# An empty setting will point to localhost:5252 by default
#
OBS_REPO_SERVERS=""

## Path:      Applications/OBS
## Description: define number of build instances
## Type:      integer
## Default:   0
## Config:    OBS
#
# 0 instances will automatically use the number of CPU's
#
OBS_WORKER_INSTANCES="0"

## Path:      Applications/OBS
## Description: define names of build instances for z/VM
## Type:      string
## Default:   ""
## Config:    OBS
#
# The names of the workers as defined in z/VM. These must have two minidisks
# assigned, and have a secondary console configured to the local machine:
# 0150 is the root device
# 0250 is the swap device
#

```

```

#OBS_WORKER_INSTANCE_NAMES="LINUX075 LINUX076 LINUX077"
OBS_WORKER_INSTANCE_NAMES=""

## Path:      Applications/OBS
## Description: The base directory, where sub directories for each worker will get
               created
## Type:      string
## Default:   ""
## Config:    OBS
#
#
OBS_WORKER_DIRECTORY=""

## Path:      Applications/OBS
## Description: The base for port numbers used by worker instances
## Type:      integer
## Default:   "0"
## Config:    OBS
#
# 0 means let the operating system assign a port number
#
OBS_WORKER_PORTBASE="0"

## Path:      Applications/OBS
## Description: Number of parallel compile jobs per worker
## Type:      integer
## Default:   "1"
## Config:    OBS
#
# this maps usually to "make -j1" during build
#
OBS_WORKER_JOBS="1"

## Path:      Applications/OBS
## Description: Run in test mode (build results will be ignore, no job blocking)
## Type:      ("yes" | "")
## Default:   ""
## Config:    OBS
#
OBS_WORKER_TEST_MODE=""

## Path:      Applications/OBS
## Description: define one or more labels for the build host.
## Type:      string
## Default:   ""

```

```

## Config:      OBS
#
# A label can be used to build specific packages only on dedicated hosts.
# For example for benchmarking.
#
OBS_WORKER_HOSTLABELS=""

## Path:        Applications/OBS
## Description: Register in SLP server
## Type:        ("yes" | "no")
## Default:     "yes"
## Config:      OBS
#
#
OBS_USE_SLP="yes"

## Path:        Applications/OBS
## Description: Use a common cache directory for downloaded packages
## Type:        string
## Default:     ""
## Config:      OBS
#
# Enable caching requires a given directory here. Be warned, content will be
# removed there !
#
OBS_CACHE_DIR=""

## Path:        Applications/OBS
## Description: Defines the package cache size
## Type:        size in MB
## Default:     ""
## Config:      OBS
#
# Set the size to 50% of the maximum usable size of this partition
#
OBS_CACHE_SIZE=""

## Path:        Applications/OBS
## Description: Defines the nice level of running workers
## Type:        integer
## Default:     18
## Config:      OBS
#
# Nicenesses range from -20 (most favorable scheduling) to 19 (least
# favorable).

```

```

# Default to 18 as some testsuites depend on being able to switch to
# one priority below (19) _and_ having changed the numeric level
# (so going from 19->19 makes them fail).
#
OBS_WORKER_NICE_LEVEL=18

## Path:      Applications/OBS
## Description: Set used VM type by worker
## Type:      ("auto" | "xen" | "kvm" | "lxc" | "zvm" | "emulator:$arch" |
  "emulator:$arch:$script" | "none")
## Default:   "auto"
## Config:    OBS
#
#
OBS_VM_TYPE="auto"

## Path:      Applications/OBS
## Description: Set kernel used by worker (kvm)
## Type:      ("none" | "/boot/vmlinuz" | "/foo/bar/vmlinuz")
## Default:   "none"
## Config:    OBS
#
# For z/VM this is normally /boot/image
#
OBS_VM_KERNEL="none"

## Path:      Applications/OBS
## Description: Set initrd used by worker (kvm)
## Type:      ("none" | "/boot/initrd" | "/foo/bar/initrd-foo")
## Default:   "none"
## Config:    OBS
#
# for KVM, you have to create with (example for openSUSE 11.2):
#
# export rootfstype="ext4"
# mkinitrd -d /dev/null -m "ext4 binfmt_misc virtio_pci virtio_blk" -k
  vmlinuz-2.6.31.12-0.2-default -i initrd-2.6.31.12-0.2-default-obs_worker
#
# a working initrd file which includes virtio and binfmt_misc for OBS in order to
  work fine
#
# for z/VM, the build script will create a initrd at the given location if
# it does not yet exist.
#
OBS_VM_INITRD="none"

```

```

## Path:      Applications/OBS
## Description: Autoseup for XEN/KVM/TMPFS disk (root) - Filesize in MB
## Type:      integer
## Default:   "4096"
## Config:    OBS
#
#
OBS_VM_DISK_AUTOSETUP_ROOT_FILESIZE="4096"

## Path:      Applications/OBS
## Description: Autoseup for XEN/KVM disk (swap) - Filesize in MB
## Type:      integer
## Default:   "1024"
## Config:    OBS
#
#
OBS_VM_DISK_AUTOSETUP_SWAP_FILESIZE="1024"

## Path:      Applications/OBS
## Description: Filesystem to use for autoseup {none,ext4}=ext4, ext3=ext3
## Type:      string
## Default:   "ext3"
## Config:    OBS
#
#
OBS_VM_DISK_AUTOSETUP_FILESYSTEM="ext3"

## Path:      Applications/OBS
## Description: Filesystem mount options to use for autoseup
## Type:      string
## Default:   ""
## Config:    OBS
#
#
OBS_VM_DISK_AUTOSETUP_MOUNT_OPTIONS=""

## Path:      Applications/OBS
## Description: Enable build in memory
## Type:      ("yes" | "")
## Default:   ""
## Config:    OBS
#
# WARNING: this requires much memory!
#

```



```

OBS_VM_USE_TMPFS=""

## Path:      Applications/OBS
## Description: Memory allocated for each VM (512) if not set
## Type:      integer
## Default:   ""
## Config:    OBS
#
#
OBS_INSTANCE_MEMORY=""

## Path:      Applications/OBS
## Description: Enable storage auto configuration
## Type:      ("yes" | "")
## Default:   ""
## Config:    OBS
#
# WARNING: this may destroy data on your hard disk !
# This is usually only used on mass deployed worker instances
#
OBS_STORAGE_AUTOSETUP="yes"

## Path:      Applications/OBS
## Description: Setup LVM via obsstoragesetup
## Type:      ("take_all" | "use_obs_vg" | "none")
## Default:   "use_obs_vg"
## Config:    OBS
#
# take_all: WARNING: all LVM partitions will be used and all data erased !
# use_obs_vg: A lvm volume group named "OBS" will be re-setup for the workers.
#
OBS_SETUP_WORKER_PARTITIONS="use_obs_vg"

## Path:      Applications/OBS
## Description: Size in MB when creating LVM partition for cache partition
## Type:      integer
## Default:   ""
## Config:    OBS
#
#
OBS_WORKER_CACHE_SIZE=""

## Path:      Applications/OBS
## Description: Size in MB when creating LVM partition for each worker root
partition

```

```

## Type:      integer
## Default:    ""
## Config:     OBS
#
#
OBS_WORKER_ROOT_SIZE=""

## Path:      Applications/OBS
## Description: Size in MB when creating LVM partition for each worker swap
partition
## Type:      integer
## Default:    ""
## Config:     OBS
#
#
OBS_WORKER_SWAP_SIZE=""

## Path:      Applications/OBS
## Description: URL to a proxy service for caching binaries used by worker
## Type:      string
## Default:    ""
## Config:     OBS
#
#
OBS_WORKER_BINARIES_PROXY=""

## Path:      Applications/OBS
## Description: URL to a ssh pub key to allow root user login
## Type:      string
## Default:    ""
## Config:     OBS
#
# This is usually used on mass (PXE) deployed workers)
#
OBS_ROOT_SSHD_KEY_URL=""

## Path:      Applications/OBS
## Description: URL to a script to be downloaded and executed
## Type:      string
## Default:    ""
## Config:     OBS
#
# This is a hook for doing special things in your setup at boot time
#

```

```
OBS_WORKER_SCRIPT_URL=""
```

2.1.2.2 BSConfig.pm

This file is a perl module used by most backend scripts, it mainly defines global variables. Since it is a perl module, after changes the backend servers need to be restarted to become aware of the changes.



Caution

If you have an perl syntax error in this file, the services won't start. Most likely you forgot the semicolon on the end of a statement.

TABLE 2.4: BSCONFIG.PM VARIABLES

Variable	Description	Values <u>default</u>	Remarks
\$hostname	FQDN of the backend host		leave as it is
\$ip	IP address of the backen host		leave as it is
\$frontend	FQDN of the frontend host	<u>undef</u>	set only if the frontend runs on another host
\$ipaccess	Map of IP access rules		Add all hosts if partition are used
\$srcserver	URL of the source server	<u>'http:// \$hostname: 5352'</u>	
\$reposerver	URL of the repo server	<u>'http:// \$hostname: 5252'</u>	partition specific
\$serviceserver	URL of the service server	<u>'http:// \$hostname: 5152'</u>	

Variable	Description	Values <u>default</u>	Remarks
\$workersrcserver	URL of the source server		optional for worker access
\$workerreposerver	URL of the repo server		optional for worker access
\$servicedir	Path to the service scripts	<u>/usr/lib/obs/service/</u>	
\$servicetempdir	Path to service temp dir	<u>/var/tmp/</u>	optional
\$serviceroot	Prefix to servicedir		optional
\$service_maxchild	Maximum number of concurrent jobs for source service	integer	unlimited if not set
\$gpg_standard_key	Path to the standard sign key		
\$hermesserver	URL of the notification server		optional
\$hermesnamespace	Namespace for the notifications		optional
\$notification_plugin	notification plugins		optional
@reposervers	List of reposervers	<u>("http://\$hostname: 5252")</u>	
\$bsdir	Path to the backend directory	<u>/srv/obs</u>	
\$bsuser	OS user running the backend	<u>obsrun</u>	
\$bsgroup	OS group running the backend	<u>obsrun</u>	

Variable	Description	Values <u>default</u>	Remarks
\$bsquotafile	Package quota for projects		optional
\$sched_asyncmode	Use asynchronous scheduler		Avoid issues with remote projects on slow networks
\$sched_startupmode	Cold start mode	<u>0</u> 1 2	
\$disable_data_sync	fdatsync		may cause data corruption
\$rundir	backend communication	<u>\$bsdir/run</u>	
\$logdir	log directory	<u>\$bsdir/log</u>	
\$nosharedtrees	Shared trees 0 = shared 1 = not shared 2 = not shared with fallback	0 1 <u>2</u>	optional for non-acl systems, should be set for access control
\$packtrack	enable binary release tracking	<u>[]</u>	
\$limit_projects	limit visibility of projects for some architectures		optional
\$relsync_pool	allow separation of releasenumbersyncing per architecture		
\$stageserver	stage server		rsync URI
\$stageserver_sync	Extra stage sync server		rsync URI
\$sign	Path to sign script		

Variable	Description	Values <u>default</u>	Remarks
\$sign_project	call sign with -- project <project>	0 <u>1</u>	
\$keyfile	Global sign key		
\$localarch	Local architecture for product building		
\$buildlog_maxsize	worker max buildlog size	<u>'500 * 1000000'</u>	in bytes
\$buildlog_maxidle	Time with no changes in the buildlog will kill the job	<u>'8 * 3600'</u>	in sec
\$xenstore_maxsize	xenstore size	<u>'20 * 1000000'</u>	current XEN has no xenstore anymore
\$gettimeout	Max timeout for get	<u>'1 * 3600'</u>	in sec
\$workerhostcheck	check script for worker		
\$powerhosts	Worker with more resources		obsolete use constraints
\$powerpkgs	packages which need workers with more resources		obsolete use constraints
\$norootexceptions	List of packages need to build as root		
\$old_style_services	Use old style source service handling	<u>0</u> 1	
\$partition	Current partition		see <i>Section 1.4,</i> <i>"Distributed setup"</i>

Variable	Description	Values <u>default</u>	Remarks
\$partitioning	Partition project mapping		see Section 1.4 , “Distributed setup”
\$partitionservers	Partition server mapping		see Section 1.4 , “Distributed setup”
\$dispatch_adjust	Adjust dispatch priority		see Section 3.4.2 , “dispatch_adjust array”
\$publishedhook_use_regex	Use regular expressions in publish hook map	<u>0</u> 1	see Section 3.5 , “Publisher Hooks”
\$publishedhook	Publish hook map		see Section 3.5 , “Publisher Hooks”
\$unpublishedhook_use_regex	Use regular expressions in unpublish hook map	<u>0</u> 1	see Section 3.6 , “Unpublisher Hooks”
\$unpublishedhook	Unpublish hook map		see Section 3.6 , “Unpublisher Hooks”

Example BSConfig.pm

```
#
# Copyright (c) 2006, 2007 Michael Schroeder, Novell Inc.
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License version 2 as
# published by the Free Software Foundation.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program (see the file COPYING); if not, write to the
# Free Software Foundation, Inc.,
# 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA
```

```

#
#####
#
# Open Build Service Configuration
#

package BSConfig;

use Net::Domain;
use Socket;

my $hostname = Net::Domain::hostfqdn() || 'localhost';
# IP corresponding to hostname (only used for $ipaccess); fallback to localhost
# since inet_aton may fail to resolve at shutdown.
my $ip = quotemeta inet_ntoa(inet_aton($hostname) || inet_aton("localhost"));

my $frontend = undef; # FQDN of the WebUI/API server if it's not $hostname

# If defined, restrict access to the backend servers (bs_repserver, bs_srcserver,
# bs_service)
our $ipaccess = {
    '127\..*' => 'rw', # only the localhost can write to the backend
    "^$ip" => 'rw',    # Permit IP of FQDN
    '.*' => 'worker', # build results can be delivered from any client in the
    network
};

# IP of the WebUI/API Server (only used for $ipaccess)
if ($frontend) {
    my $frontendip = quotemeta inet_ntoa(inet_aton($frontend) ||
    inet_aton("localhost"));
    $ipaccess->{$frontendip} = 'rw' ; # in dotted.quad format
}

# Change also the SLP reg files in /etc/slp.reg.d/ when you touch hostname or port
our $srcserver = "http://$hostname:5352";
our $reposerver = "http://$hostname:5252";
our $serviceserver = "http://$hostname:5152";

# you can use different ports for worker connections
#our $workersrcserver = "http://$hostname:5353";
#our $workerreposerver = "http://$hostname:5253";

our $servicedir = "/usr/lib/obs/service/";
#our $servicetempdir = "/var/temp/";

```



```

#our $serviceroot = "/opt/obs/MyServiceSystem";

# Maximum number of concurrent jobs for source service
#our $service_maxchild = 20;

our $gpg_standard_key = "/srv/obs/obs-default-gpg.asc";
# optional notification service:
#our $hermesserver = "http://$hostname/hermes";
#our $hermesnamespace = "OBS";
#
# Notification Plugin, multiple plugins supported, separated by space
#our $notification_plugin = "notify_hermes notify_rabbitmq";
#

# For the workers only, it is possible to define multiple repository servers here.
# But only one source server is possible yet.
our @reposervers = ("http://$hostname:5252");

# Package defaults
our $bsdir = '/srv/obs';
our $bsuser = 'obsrun';
our $bsgroup = 'obsrun';
#our $bsquotafile = '/srv/obs/quota.xml';

# Use asynchronus scheduler. This avoids hanging schedulers on remote projects,
# when the network is slow or broken. This will become the default in future
# our $sched_asyncmode = 1;

# Define how the scheduler does a cold start. The default (0) is to request the
# data for all packages, (1) means that only the non-remote packages are fetched,
# (2) means that all of the package data fetches get delayed.
# our $sched_startupmode = 0;

# Disable fdatsync calls, increases the speed, but may lead to data
# corruption on system crash when the filesystem does not guarantees
# data write before rename.
# It is esp. required on XFS filesystem.
# It is safe to be disabled on ext4 and btrfs filesystems.
#our $disable_data_sync = 1;

# Package rc script / backend communication + log files
our $rundir = "$bsdir/run";
our $logdir = "$bsdir/log";

# optional for non-acl systems, should be set for access control

```

```

# 0: trees are shared between projects (built-in default)
# 1: trees are not shared (only usable for new installations)
# 2: new trees are not shared, in case of a missing tree the shared
#    location is also tried (package default)
our $nosharedtrees = 2;

# enable binary release tracking by default for release projects
our $packtrack = [];

# optional: limit visibility of projects for some architectures
#our $limit_projects = {
# "ppc" => [ "openSUSE:Factory", "FATE" ],
# "ppc64" => [ "openSUSE:Factory", "FATE" ],
#};

# optional: allow separation of releasnumber syncing per architecture
# one counter pool for all ppc architectures, one for i586/x86_64,
# arm archs are separated and one for the rest in this example
our $relysync_pool = {
  "local" => "local",
  "i586" => "i586",
  "x86_64" => "i586",
  "ppc" => "ppc",
  "ppc64" => "ppc",
  "ppc64le" => "ppc",
  "mips" => "mips",
  "mips64" => "mips",
  "mipsel" => "mipsel",
  "mips64el" => "mipsel",
  "aarch64" => "arm",
  "aarch64_ilp32" => "arm",
  "armv4l" => "arm",
  "armv5l" => "arm",
  "armv6l" => "arm",
  "armv6hl" => "arm",
  "armv7l" => "arm",
  "armv7hl" => "arm",
  "armv5el" => "armv5el", # they do not exist
  "armv6el" => "armv6el",
  "armv7el" => "armv7el",
  "armv8el" => "armv8el",
  "sparcv9" => "sparcv9",
  "sparc64" => "sparcv9",
};

```

```

#No extra stage server sync
#our $stageserver = 'rsync://127.0.0.1/put-repos-main';
#our $stageserver_sync = 'rsync://127.0.0.1/trigger-repos-sync';

#No package signing server
our $sign = "/usr/bin/sign";
#Extend sign call with project name as argument "--project $NAME"
#our $sign_project = 1;
#Global sign key
our $keyfile = "/srv/obs/obs-default-gpg.asc";

# Use a special local arch for product building
# our $localarch = "x86_64";

# config options for the bs_worker
#
#our buildlog_maxsize = 500 * 1000000;
#our buildlog_maxidle = 8 * 3600;
#our xenstore_maxsize = 20 * 1000000;
#our gettimeout = 1 * 3600;
#
# run a script to check if the worker is good enough for the job
#our workerhostcheck = 'my_check_script';
#
# Allow to build as root, exceptions per package
# the keys are actually anchored regexes
# our $norootexceptions = { "my_project/my_package" => 1, "openSUSE:Factory.*/
installation-images" => 1 };

# Use old style source service handling
# our $old_style_services = 1;

###
# Optional support to split the binary backend. This can be used on large servers
# to separate projects for better scalability.
# There is still just one source server, but there can be multiple servers which
# run each repserver, schedulers, dispatcher, warden and publisher
#
# This repo service is the 'home' server for all home:* projects. This and the
# $reposerver setting must be different on the binary backend servers.
# our $partition = 'home';
#
# this defines how the projects are split. All home: projects are hosted
# on an own server in this example. Order is important.
# our $partitioning = [ 'home:' => 'home',

```

```

#             '.*'      => 'main',
#             ];
#
# our $partitionservers = { 'home' => 'http://home-backend-server:5252',
#                           'main' => 'http://main-backend-server:5252',
#                           };

# Publish hooks
our $publishedhook_use_regex = 1;
our $publishedhook = {
    "Product\SLES12"      => "/usr/local/bin/script2run_sles12",
    "Product\SLES11.*"    => "/usr/local/bin/script2run_sles11",
};

# host specific configs
my $hostconfig = __FILE__;
$hostconfig =~ s/[^\/]*$/bsconfig.$hostname/;
if (-r $hostconfig) {
    print STDERR "reading $hostconfig...\n";
    require $hostconfig;
}

1;

```

2.2 Log files

2.2.1 Frontend

The frontend log files are found under */srv/www/obs/api/log*.

The following frontend logfiles exist:

- *apache_access.log* - apache requests
- *apache_error.log* - errors from apache
- *backend_access.log* - API → backend requests
- *clockworkd.clock.output* → timer event log

- `delayed_job.log` → delayed job log
- `production.log` → main ruby log
- `production.searchd.log` - search daemon log
- `production.searchd.query.log` - search request logs

2.2.2 Backend

The backend logfiles are found by default under `/srv/obs/log/`.

The following backend logfiles exist:

- `dispatcher.log` - dispatcher log
- `dodup.log` - download on demand log (since 2.7)
- `publisher.log` - publisher log
- `rep_server.log` - repo server log
- `scheduler_<arch>.log` - scheduler log for each architecture
- `signer.log` - sign service log
- `src_server.log` - source server log
- `src_service.log` - source service daemon log
- `warden.log` - warden log

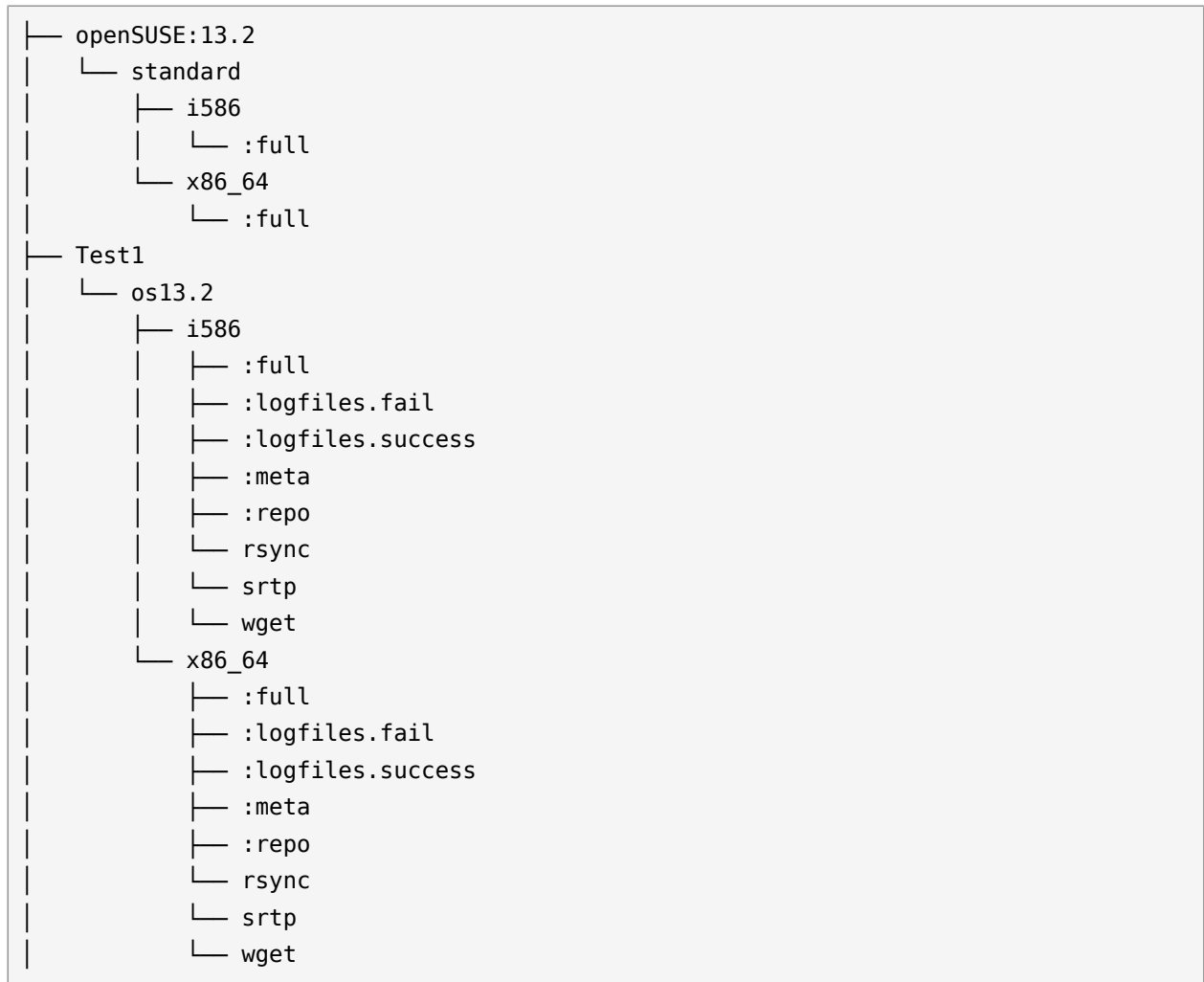
2.3 `/src/obs` tree

The default backend data directory is located under `/srv/obs/`. Here are a bunch of subdirectories used for communication between the different server, to store data, status information und logs. Here is one file **`configuration.xml`** in the top directory, which stores the global OBS configuration for the backend. You should not modify this file directly, but use the API / configuration interface instead, since this information needs to kept in sync with the frontend.

2.3.1 build directory

In this subdirectory managed by the repo server daemon, all repository data, meta data and build results are stored in a hierarchical tree.

Example build directory tree of a binary imported distribution (OpenSuSE:13.2) and a small test project with 3 packages:



2.3.2 db directory

Backend database root directory use by the source server, repo server scheduler and publisher. Nobody should touch this.

2.3.3 diffcache directory

Cache for source server compare operations.

2.3.4 events directory

Communication between services.

2.3.5 info directory

Scheduler information managed by the scheduler and used by the repo server.

2.3.6 jobs directory

The build jobs are stored in the `/srv/obs/jobs` directory. They are organized by build architecture:

```
jobs
├─ armv7l
├─ i586
├─ load
└─ x86_64
    └─ Release:Stable::SLE-12_GA::CI-demo-36db80552b735e193dced13f058f866f
```

The `jobs/load` file contains statistical data about the build jobs.

2.3.7 log directory

Contains the log files of the backend daemons.

2.3.8 projects directory

Contains the project hierarchy and meta data under revision control.

2.3.9 remotecache directory

Cache for remote repository information.

2.3.10 repos directory

Directory managed by the publisher to collect build results, also used by the repo server and scheduler to find build results.

2.3.11 repos_sync directory

Directory with files pointing to the project root directories, helper for publisher rsync.

2.3.12 run directory

State and lock information for the backend daemons

2.3.13 sources directory

All package sources under revision control in one directory per package, managed by the source server. Package sources are by default deduplicated across all projects, as long a source file has the same MD5 sum, it is only stored once. A pseudo '**_project**' package exist in the directory containing the project meta data revisions. '**:service**' and '**:upload**' are temporary directories used by the source server.

Example sources directory structure:

```
sources/  
├─ CI-demo  
...  
├─ srtp  
├─ test1  
├─ _project  
├─ :service  
├─ :upload
```


2.3.14 trees directory

Revision control data for project and packages, managed by the source server.

2.3.15 upload directory

Temporary directory for uploading files for other backend components.

2.3.16 workers directory

Worker information

2.4 Meta data

2.4.1 OBS revision control

This section gives a short generic overview how the revision informations are stored in the OBS backend for packages and projects. The OBS backend stores all files in a light weight content based hierarchical tree. Each file is hashed (with MD5) and stored with the hash as part of the filename under the */srv/obs/tree* or */srv/obs/sources* directories. The revision information is stored in separate files by the Source Server in the */srv/obs/projects* directory.

2.4.1.1 OBS revision control files

The revision information is stored in simple CSV like file format with a bar (|) as delimiter between the 8 columns. The files do have the extension *.rev* for package/project revision data and *.mref* for meta revision data. The hash then points to a *<hash>-MD5SUMS* file in the */srv/obs/tree/* directories which have the file list with MD5 hashes of this revision. The hashes in this file list are pointing to the source files in the */srv/obs/sources* tree.

An example revision file:

```
1|1|56cdd3adb778089d1fcc49b92bb93e5b|0.9|1464005086|user4|initial version|
```

```
2|2|fe7aa1ade5c9d005de738c234c90bc90|0.9|1464005304|user4|fix spec file|
3|1|72c7986e694f45ab1a62779e64e92a8f|1.0|1464005339|user4|new version|
4|2|699e9931e6f167d78e65bbe5853f592f|1.0|1464006221|user4|add patch file|
5|1|0cfc3a2297f38d2aa9d8d0e98fc22a38|1.1|1464007797|user4|new version|
```

TABLE 2.5: THE 8 COLUMNS

Column	Content	XML tag	may empty
1	revision number	ref	no
2	version revision number	vref	yes
3	hash	srcmd5	no
4	version	version	yes
5	time stamp	time	no
6	user	user	no
7	commit message	comment	yes
8	request id	requestid	yes

Depending for what kind of target (package, project or meta data) is used, fields can be empty or have special values e.g. **unknown** for the version.

Example MD5SUMS file

```
/srv/obs # cat trees/Test1/package1/56cdd3adb778089d1fcc49b92bb93e5b-MD5SUMS
0a17daaa913df9e50ee65e83a1898363 package1.spec
1f810b3521242a98333b7bbf6b2b7ef7 test1.sh
```

2.4.1.2 OBS revision API

The revision info can be retrieved via API calls for the specific package, e.g using /source/<project>/<package>/_history.

Specific revisions of files can be retrieved with the optional "rev=N" parameter e.g. /source/<project>/<package>/<file>?rev=N.

On PUT and POST methods for files the optional "comment = some + comment" can be used to set a commit message.

2.4.2 Project meta data

Project meta data are XML files containing the meta project information, such as title, description, related user and groups with roles, build settings, repository settings, publish settings, debug settings and more.

TABLE 2.6: PROJECT META XML

XML tag	Attributes	Content
project	name	project name
title		Short description
description		Developer information
person	userid	login name
person	role	role (maintainer, bugowner, ...)
group	groupid	group name
group	role	role (maintainer, bugowner, ...)
devel		An optional devel project
build		optional build flags
publish		optional publish flags
useforbuild		optional useforbuild flags
debuginfo		optional debuginfo flags
binarydownload		optional binarydownload flags
repository	name	name of the repository for build results
repository path	project	name of the source project for remaining build requires
repository path	repository	name of repository in the source project

XML tag	Attributes	Content
repository arch		architecture name
remoteurl		path to a remote OBS API for interconnect

Example project meta data:

```
<project name="Test11">
  <title>Test project 11</title>
  <description>Project for demo</description>
  <person userid="Admin" role="maintainer"/>
  <person userid="user0" role="maintainer"/>
  <group groupid="obsprj3" role="maintainer"/>
  <repository name="openSUSE_13.2">
    <path project="openSUSE.org:openSUSE:13.2" repository="standard"/>
    <arch>x86_64</arch>
  </repository>
</project>
```

2.4.3 Package meta data

XML file about package meta informations, like Title, description, related user and groups with roles, build settings, publish settings, debug settings and more. Most XML tags are the same as for projects.

Example package meta data:

```
<package name="test1" project="Test11">
  <title>A test package for learning</title>
  <description>A example test package for learning.&#13;
</description>
  <person userid="user5" role="bugowner"/>
  <person userid="user5" role="maintainer"/>
  <build>
    <enable repository="openSUSE_13.2"/>
  </build>
  <publish>
    <disable repository="openSUSE_13.2"/>
  </publish>
  <debuginfo>
    <disable/>
```

```
</debuginfo>
</package>
```

2.4.4 Attribute meta data

Attributes can be used to add special information to packages. Attributes can be used to trigger special actions.

Example attribute data:

```
<attributes>
  <attribute name="Issues" namespace="OBS"/>
  <attribute name="AutoCleanup" namespace="OBS">
    <value>2016-06-30 00:00:00</value>
  </attribute>
  <attribute name="AutoCleanup" namespace="OBS">
    <value></value>
  </attribute>
</attributes>
```

2.4.5 Job files

Jobs are stored bei the scheduler in the `/srv/obs/jobs` directory and contain the build setup information for the package, e.g. reference to the exact source version, build dependencies, build repository information, timestamps.

Sample job file:

```
<buildinfo project="Release:Stable" repository="SLE-12_GA" package="CI-demo"
srcserver="http://obs.b1-systems.de:5352"
reposer="http://obs.b1-systems.de:5252">
  <job>Release:Stable::SLE-12_GA::
CI-demo-36db80552b735e193dced13f058f866f</job>
  <arch>x86_64</arch>
  <srcmd5>36db80552b735e193dced13f058f866f</srcmd5>
  <verifymd5>36db80552b735e193dced13f058f866f</verifymd5>
  <rev>2</rev>
  <disturl>obs://b1-systems.de/Release:Stable/SLE-12_GA/
36db80552b735e193dced13f058f866f-CI-demo</disturl>
  <reason>new build</reason>
```

```

<needed>0</needed>
<revtime>1461077600</revtime>
<readytime>1461077708</readytime>
<file>CI-demo.spec</file>
<versrel>0.1.9-2</versrel>
<bcnt>1</bcnt>
<release>2.1</release>
<debuginfo>1</debuginfo>
<prjconfconstraint>linux:version:min 3.0.0</prjconfconstraint>
<bdep name="aaa_base" preinstall="1" runscripts="1" notmeta="1" />
<bdep name="attr" preinstall="1" notmeta="1" />
<bdep name="bash" preinstall="1" notmeta="1" />
<bdep name="coreutils" preinstall="1" notmeta="1" />
<bdep name="diffutils" preinstall="1" notmeta="1" />
<bdep name="filesystem" preinstall="1" notmeta="1" />
<bdep name="fillup" preinstall="1" notmeta="1" />
<bdep name="glibc" preinstall="1" notmeta="1" />
<bdep name="grep" preinstall="1" notmeta="1" />
<bdep name="libbz2-1" preinstall="1" notmeta="1" />
<bdep name="libgcc_s1" preinstall="1" notmeta="1" />
<bdep name="m4" preinstall="1" notmeta="1" />
...
<path project="Release:Stable" repository="SLE-12_GA"
server="http://obs.b1-systems.de:5252" />
<path project="SUSE:SLE-12:GA" repository="standard"
server="http://obs.b1-systems.de:5252" />
</buildinfo>

```

3 Administration

3.1 Tools

3.1.1 obs_admin

`obs_admin` is a commandline tool used on the backend server(s) to manage the running services, submit maintenance tasks and debugging. It should be only used by experienced admins.

It has an builtin help which you can display with `obs_admin --help`.

Options to control the running services:

```
Job Controlling
=====

--shutdown-scheduler <architecture>
  Stops the scheduler nicely with dumping out its current state
  for fast startup.

--check-project <project> <architecture>
--check-project <project> <repository> <architecture>
--check-all-projects <architecture>
  Check status of a project and its repositories again

--deep-check-project <project> <architecture>
--deep-check-project <project> <repository> <architecture>
  Check status of a project and its repositories again
  This deep check includes also the sources, in case of lost events.

--check-package <project> <package> <architecture>
  Check status of a package in all repositories

--publish-repository <project> <repository>
  Creates an event for the publisher. The scheduler is NOT scanning for new
  packages.
  The publisher may skip the event, if nothing has changed.
  Use --republish-repository when you want to enforce a publish.
```

```

--unpublish-repository <project> <repository>
  Removes the prepared :repo collection and let the publisher remove the result.
This
  is also updating the search database.
  WARNING: this works also for locked projects!

--prefer-publish-event <name>
  prefers a publish event to be next. <name> is the file name inside of the publish
  event directory.

--republish-repository <project> <repository>
  enforce to publish a repository

--rebuild-full-tree <project> <repository> <arch>
  rebuild the content of :full/ directory

--clone-repository <source project> <source repository> <destination repository>
--clone-repository <source project> <source repository> <destination project>
<destination repository>
  Clone an existing repo into another existing repository.
  Usefull for creating snapshots.

--rescan-repository <project> <repository> <architecture>
  Asks the scheduler to scan a repository for new packages and add
  them to the cache file.

--force-check-project <project> <repository> <architecture>
  Enforces the check of an repository, even when it is currently blocked due to
amount of
  calculating time.

--create-patchinfo-from-updateinfo
  creates a patchinfo submission based on an updateinfo information.

```

Options for maintenance are:

Maintenance Tasks

=====

Note: the --update-*-db calls are usually only needed when corrupt data has been created, for
 example after a file system corruption.


```

--update-source-db [<project>]
    Update the index for all source files.

--update-request-db
    Updates the index for all requests.

--remove-old-sources <days> <y> (--debug)
    WARNING: this is an experimental feature atm. It may trash your data, but you
    have anyway
        a backup, right?
    remove sources older than <x> days, but keep <y> number of revisions
    --debug for debug output

```

Options for debugging:

Debug Options

=====

```

--dump-cache <project> <repository> <architecture>
    Dumps out the content of a binary cache file.
    This shows all the content of a repository, including all provides
    and requires.

--dump-state <architecture>

--dump-project-from-state <project> <arch>
    dump the state of a project.

--dump-relsync <file>
    To dump content of :relsync files.

--set-relsync <file> <key> <value>
    Modify key content in a a :relsync file.

--check-meta-xml <project>
--check-meta-xml <project> <package>
    Is parsing a project or package xml file and puts out error messages, in case of
    errors.

--check-product-xml <file>
    Is parsing a product xml file and puts out error messages, in case of errors.
    It does expand all xi:include references and validates the result.

--check-product-group-xml <file>

```

```

    Is parsing a group xml file from a product definition and puts out error
    messages, in case of errors.

--check-kiwi-xml <file>
--check-kiwi-xml <project> <package>
    Is parsing a kiwi xml file and puts out error messages, in case of errors.

--check-constraints <file>
--check-constraints <project> <package>
    Validates a _constraints file

--check-pattern-xml <file>
    Is parsing a pattern xml file and puts out error messages, in case of errors.

--check-request-xml <file>
    Is parsing a request xml file and puts out error messages, in case of errors.

--parse-build-desc <file> [<arch> [<buildconfigfile>]]
    Parse a spec, dsc or kiwi file with the Build script parser.

--show-scheduler-architectures
    Show all architectures which are configured in configuration.xml to be supported
    by this instance.

--show-delta-file <file>
    Show all instructions of a OBS delta file

--show-delta-store <file>
    Show delta store statistics

```

3.1.2 **osc**

The **osc** commandline client is mainly used by developer and packager. But for some tasks also admin people need this tool. It has an builtin help simple call *osc --help*. The tool need to be configured first to know the OBS API URL and your user details.

To configure the **osc** tool the first time you need to call it with

```

osc -A <URL to the OBS API>
For example:
osc -A https://api.testobs.org

```

Follow the instructions on the terminal.



Warning

The password is stored in clear text in the `.osrc` file by default, so you need to give this file restrictive access rights, only read/write access for your user should be allowed. **osc** allows to store the password in other ways (in keyrings for example) and may use different methods for authentication like kerberos see [Section 3.7.5.2, "Kerberos"](#)

For the admins the most important **osc** subcommands are:

- **meta** - to create or update projects or package data
- **api** - to read and write online configuration data

3.1.2.1 **osc meta subcommand**

meta: Show meta information, or edit it

Show or edit build service metadata of type `<prj|pkg|prjconf|user|pattern>`.

This command displays metadata on buildservice objects like projects, packages, or users. The type of metadata is specified by the word after "meta", like e.g. "meta prj".

`prj` denotes metadata of a buildservice project.

`prjconf` denotes the (build) configuration of a project.

`pkg` denotes metadata of a buildservice package.

`user` denotes the metadata of a user.

`pattern` denotes installation patterns defined for a project.

To list patterns, use '`osc meta pattern PRJ`'. An additional argument will be the pattern file to view or edit.

With the `--edit` switch, the metadata can be edited. Per default, `osc` opens the program specified by the environmental variable `EDITOR` with a temporary file. Alternatively, content to be saved can be supplied via the `--file` switch. If the argument is '-', input is taken from stdin:
`osc meta prjconf home:user | sed ... | osc meta prjconf home:user -F -`

For meta `prj` and `prjconf` updates optional commit messages can be applied

with --message.

When trying to edit a non-existing resource, it is created implicitly.

Examples:

```
osc meta prj PRJ
osc meta pkg PRJ PKG
osc meta pkg PRJ PKG -e
```

Usage:

```
osc meta <prj|prjconf> [-r|--revision REV] ARGS...
osc meta <prj|pkg|prjconf|user|pattern> ARGS...
osc meta <prj|pkg|prjconf|user|pattern> [-m|--message TEXT] -e|--edit
ARGS...
osc meta <prj|pkg|prjconf|user|pattern> [-m|--message TEXT] -F|--file
ARGS...
osc meta pattern --delete PRJ PATTERN
osc meta attribute PRJ [PKG [SUBPACKAGE]] [--attribute ATTRIBUTE]
[--create|--delete|--set [value_list]]
```

Options:

```
-h, --help          show this help message and exit
--delete            delete a pattern or attribute
-s ATTRIBUTE_VALUES, --set=ATTRIBUTE_VALUES
                    set attribute values
-R, --remove-linking-repositories
                    Try to remove also all repositories building against
                    remove ones.
-c, --create        create attribute without values
-e, --edit          edit metadata
-m TEXT, --message=TEXT
                    specify log message TEXT. For prj and prjconf meta
                    only
-r REV, --revision=REV
                    checkout given revision instead of head revision.
```

For

```
prj and prjconf meta only
-F FILE, --file=FILE
                    read metadata from FILE, instead of opening an
editor.
```

'-' denotes standard input.

```
-f, --force        force the save operation, allows one to ignores some
errors like depending repositories. For prj meta
only.
```

```
--attribute-project
```

```
            include project values, if missing in packages
--attribute-defaults
            include defined attribute defaults
-a ATTRIBUTE, --attribute=ATTRIBUTE
            affect only a given attribute
```

3.1.2.2 osc api subcommand

api: Issue an arbitrary request to the API

Useful for testing.

URL can be specified either partially (only the path component), or fully with URL scheme and hostname ('http://...').

Note the global -A and -H options (see osc help).

Examples:


```
osc api /source/home:user
osc api -X PUT -T /etc/fstab source/home:user/test5/myfstab
osc api -e /configuration
```

Usage:

```
osc api URL
```

Options:

```
-h, --help          show this help message and exit
-a NAME STRING, --add-header=NAME STRING
                    add the specified header to the request
-T FILE, -f FILE, --file=FILE
                    specify filename to upload, uses PUT mode by default
-d STRING, --data=STRING
                    specify string data for e.g. POST
-e, --edit          GET, edit and PUT the location
-X HTTP_METHOD, -m HTTP_METHOD, --method=HTTP_METHOD
                    specify HTTP method to use (GET|PUT|DELETE|POST)
```

The online API documentation is available at <https://build.opensuse.org/apidocs> 

Some examples for admin stuff:

```
# Read the global configuration file
osc api /configuration
```

```
# Update the global configuration
osc api /configuration -T /tmp/configuration.xml

# Read the distributions list
osc api /distributions
# Udate the distributions list
osc api /distributions -T /tmp/distributions.xml

# retrieve statistics
osc api /statistics/latest_added
```

3.2 Managing build targets

3.2.1 Interconnect

Using another Open Build Service as source for build targets is the easiest way to start. The advantage is, that you save local resources and you do not need to build everything from scratch. The disadvantage is that you depend on the remote instance, if it has a downtime your instance cannot do any builds for these targets, if the remote admins decide to remove some targets you cannot use them anymore.

The easiest way to interconnect with some of the public OBS instances is to use the Web UI. You need to login on an administrator account of your instance to do this. On the start page of an administrator account you will find a **Configuration** link. On the Configuration page you find a Interconnect tab on the top, use this and select the public side you want.

If you want to connect to a not listed instance, you can simple create a remote project using the *osc meta prj* command. A remote project differs from a local project the it has a **remoteurl** tag (see [Section 2.4.2, "Project meta data"](#)).

Example:

```
<project name="openSUSE.org">
  <title>openSUSE.org Project Link</title>
  <description>
This project refers to projects hosted on the openSUSE Build Service
</description>
  <remoteurl>https://api.opensuse.org/public</remoteurl>
```

```
</project>
```

Sending this via osc to the server:

```
osc meta prj -m "add openSUSE.org remote" -F /tmp/openSUSE.org.prj
```

3.2.2 Importing distributions

With local hosted distributions packages you are independent from other parties. On sides with no or bad internet connections, this is the only way to go. You do not need to build the distribution packages on your instance, you can use binary packages for this. Here are different ways to get a local build repository:

1. mirror a distribution from another OBS instance
2. mirror a binary distribution from a public mirror and import the binaries
3. use already existing local install repositories (e.g. from an SMT)
4. use the install media to import the binaries

These tasks need to be run on the obs backend. In a partition setup you need to run it on the partition which would be the owner for the project.

3.2.2.1 Mirror from a remote OBS instance

Mirroring a project from a remote OBS instance can be done with the **obs_mirror_project** script which is supplied with the obs sources and via the obs-utils package. You can get the latest version from GitHub: https://raw.githubusercontent.com/openSUSE/open-build-service/master/dist/obs_mirror_project.

The usage:

```
Usage: obs_mirror_project.rb -p PROJECT -r REPOSITORY
                        [-a ARCHITECTURE] [-d DESTINATION] [-A APIURL] [-t] [-v]
```

```
Example: (mirror openSUSE 13.1 as base distro)
obs_mirror_project -p openSUSE:13.1 -r standard -a i586,x86_64
```

```
Options help:
  -p, --proj PROJECT      Project Name: eg. openSUSE:13.1,Ubuntu:14.04,etc.
  -r, --repo REPOSITORY   Repository Name:  eg. standard,qemu,etc.
  -a, --arch Architecture Architecture Name: eg. i586,x86_64,etc.
  -d, --dest DESTINATION  Destination Path:  eg. /obs
                           Default: PWD (current working directory)
  -A, --api APIURL        OSC API URL :Default: https://api.opensuse.org
  -t, --trialrun          Trial run: not executing actions
  -v, --verbose           Verbose
  -h, --help              Display this screen
```

3.2.2.2 Import binary packages

This is the same procedure for all local sources. If you have a local copy of a distribution, you can either use symbolic links to the binary packages or copy them in a directory on the backend repo server under the `/srv/obs/build` directory. You should follow the common name schema for build repository here. As first step you should create a empty project for the distribution, you can use the WebUI or the **osc** commandline tool. Then you add a repository with name *standard* and the build architectures you want. Here a example project meta file:

```
<project name="SUSE:13.2">
  <title>openSUSE 13.2 build repositories</title>
  <description>openSUSE 13.2 build repositories</description>
  <person userid="Admin" role="maintainer"/>
  <build>
    <disable repository="standard"/>
  </build>
  <publish>
    <disable/>
  </publish>
  <repository name="standard">
    <arch>x86_64</arch>
    <arch>i586</arch>
  </repository>
</project>
```

After you have created the project with these settings, the `/srv/obs/build` directory should have a tree for SUSE:13.2:

```
/srv/obs/
└─ build
```



```

└─ SUSE:13.2
   └─ standard
      ├── i586
      │   ├── :bininfo
      │   └── :schedulerstate
      └── x86_64
          ├── :bininfo
          └── :schedulerstate

```



Warning

All the directories under `/srv/obs/build` have to be owned by the **obsrun** user and group. The **obsrun** user need write access to them. **If not the scheduler process will crash on your instance.**

You need to import the project configuration as well, you can get them for example from the openSUSE buildserver.

```

osc -A https://api.opensuse.org meta prjconf openSUSE:13.2 >/tmp/13.2.prjconf
osc meta prjconf -m 'Original version from openSUSE' SUSE:13.2 -F /tmp/13.2.prjconf

```

Now you need to create the directory *'full'* for the binary sources under each architecture, this should be owned by **obsrun** too.

```

testobs:/srv/www/obs/api # mkdir /srv/obs/build/SUSE\:13.2/standard/i586/:full
testobs:/srv/www/obs/api # mkdir /srv/obs/build/SUSE\:13.2/standard/x86_64/:full
testobs:/srv/www/obs/api # chown obsrun:obsrun \
                           /srv/obs/build/SUSE\:13.2/standard/i586/:full
testobs:/srv/www/obs/api # chown obsrun:obsrun \
                           /srv/obs/build/SUSE\:13.2/standard/x86_64/:full

```

Now you can copy (or link) all binary packages for the architecture in the *:full* directory. You need the architecture specific package and the *noarch* packages as well.



Important

If you import packages for enterprise distributions like SLES12 you also need the packages from the SDK. Maybe you need packages from addon products as well, depending what software you want build.

Finally you should trigger a rescan for the project on the backend server using **obs_admin**:

```
testobs # obs_admin --rescan-repository SUSE:13.2 standard i586
testobs # obs_admin --rescan-repository SUSE:13.2 standard x86_64
```

This reads all packages and creates the dependency tree.

3.3 Source Services

Source Services are tools to validate, generate or modify sources in a trustable way. They are designed as smallest possible tools and can be combined following the powerful idea of the classic UNIX design.

Design goals of source services were:

- server side generated files must be easy to identify and must not be modifiable by the user. This way others user can trust them to be generated in the documented way without modifications.
- generated files must never create merge conflicts
- generated files must be a separate commit to the user change
- services must be runnable at any time without user commit
- services must be runnable on server and client side in the same way
- services must be designed in a safe way. A source checkout and service run must never harm the system of a user.
- services shall be designed in a way to avoid unnecessary commits. This means there shall be no time-dependent changes. In case the package already contains the same file, the newly generated file must be dropped.
- local services can be added and used by everybody.
- server side services must be installed by the admin of the OBS server.
- service can be defined per package or project wide.

3.3.1 Using services for validation

Source Services may be used to validate sources. This can happen per package, which is useful when the packager wants to validate that downloaded sources are really from the original maintainer. Or validation can happen for an entire project to apply general policies. These services can't get skipped in any package

Validation can happen by validating files (for example using the `verify_file` or `source_validator` service. These services just fail in the error case which leads to the build state "broken". Or validation can happen by redoing a certain action and store the result as new file as `download_files` is doing. In this case the newly generated file will be used instead of the committed one during build.

3.3.2 Different Modes when using services

Each service can be used in a special mode defining when it should run and how to use the result. This can be done per package or globally for an entire project.

3.3.2.1 Default Mode

The default mode of a service is to always run after each commit on the server side and locally before every local build.

3.3.2.2 trylocal Mode

The trylocal mode is running the service locally when using current osc versions. The result gets committed as standard files and not named with `_service:` prefix. Additionally the service runs on the server by default, but usually the service should detect that the result is the same and skip the generated files. In case they differ for any reason (because the webui or api was used for example) they get generated and added on the server.

3.3.2.3 localonly Mode

The localonly mode is running the service locally when using current osc versions. The result gets committed as standard files and not named with `_service:` prefix. The service is never running on the server side. It is also not possible to trigger it manually.

3.3.2.4 serveronly Mode

The serviceonly mode is running the service on the service only. This can be usefull, when the service is not available or can not work on developer workstations.

3.3.2.5 buildtime Mode

The service is running inside of the build job, for local and server side builds. A side effect is that the service package is becoming a build dependency and must be available. Every user can provide and use a service this way in their projects. The generated sources are not part of the source repository, but part of the generated source packages. Network access is not be available when the workers are running in a secure mode.

3.3.2.6 disabled Mode

The disabled mode is neither running the service locally or on the server side. It can be used to temporarily disable the service but keeping the definition as part of the service definition. Or it can be used to define the way how to generate the sources and doing so by manually calling

```
osc service disabledrun
```

The result will get committed as standard files again.

3.3.3 How are source service definitions stored

The called services are always defined in a _service file. It is either part of the package sources or used project-wide when stored inside the _project package.

The _service file contains a list of services which get called in this order. Each service may define a list of parameters and a mode. The project wide services get called after the per package defined services. The _service file is an xml file like this example:

```
<services>
  <service name="download_files" mode="trylocal" />
  <service name="verify_file">
    <param name="file">krabber-1.0.tar.gz</param>
```

```

    <param name="verifier">sha256</param>
    <param
name="checksum">7f535a96a834b31ba2201a90c4d365990785dead92be02d4cf846713be938b78</
param>
  </service>
  <service name="update_source" mode="disabled" />
</services>

```

This example downloads the files via `download_files` service via the given URLs from the spec file. When using `osc` this file gets committed as part of the commit. Afterwards the `krabber-1.0.tar.gz` file will always be compared with the sha256 checksum. And last but not least there is the `update_source` service mentioned, which is usually not executed. Except when `osc service disabledrun` is called, which will try to upgrade the package to a newer source version available online.

3.3.4 Dropping a source service again

Sometimes it is useful to continued to work on generated files manually. In this situation the `_service` file needs to be dropped, but all generated files need to be committed as standard files. The OBS provides the "mergeservice" command for this. It can also be used via `osc` by calling `osc service merge`.

3.4 Dispatch Priorities

The dispatcher takes a job from the scheduler and assign it to a free worker. It tries to share the available build time fair between all the project repositories with pending jobs. To achieve this the dispatcher calculates a **load** per project repository of the used build time (similar to the system load in Unix operating systems). The dispatcher assigned jobs to build clients from the repository with the lowest load (thereby increasing the its load). It is possible to tweak this mechanism via dispatching priorities assigned to the repositories via the `/build/_dispatchpriosAPI` call or via the `dispatch_adjust` array in the *BSConfig.pm* [Section 2.1.2.2](#), "*BSConfig.pm*" configuration file.

3.4.1 /build/_dispatchprios API call

The `/build/_dispatchprios` API call allows an Admin to set a priority for defined projects and repositories using the HTML put method. With the HTML get method the current XML priority file can be read.

```
<dispatchprios>
  <prio project="ProjectName" repository="RepoName" arch="Architecture"
    adjust="Number" />
</dispatchprios>
```

The attributes *project*, *repository* and *arch* are all optional, if for example *arch* and *repository* are missing the entry is used for all repositories and architectures for the given project. It is not supported to use regular expressions for the names. The adjust value is taken as logarithmic scale factor to the current load of the repositories during the compare. Projects without any entry get a default priority of 0, higher values cause the matching projects to get more build time.

Example dispatchprios XML file

```
<dispatchprios>
  <prio project="DemoProject1" repository="openSUSE_Leap_42.1" adjust="10" />
  <prio project="Test1" adjust="5" />
  <prio project="Test11" repository="openSUSE_13.2" arch="i586" adjust="-10"/>
</dispatchprios>
```

TABLE 3.1: **ROUNDED SCALE FACTORS RESULTING FROM A PRIORITY**

priority	scale factor	priority	scale factor
-50	100000	3	0.5
-30	1000	5	0.3
-20	100	7	0.2
-15	30	10	0.1
-10	10	15	0.03
-7	5	20	0.01
-5	3	30	0.001
-3	2	40	0.0001

priority	scale factor	priority	scale factor
0	1	50	0.00001

3.4.2 dispatch_adjust array

With the **dispatch_adjust** array in the *BSConfig.pm* file the dispatch priorities of project repositories based on regular expressions for the project, repository name and maybe architecture. Each match will add or subtract a value to the priority of the repository. The default priority is 0, higher values cause the matching projects to get more build time.

Each entry in the dispatch_adjust array has the format

```
'regex string' => priority adjustment
```

The full name of a build repository looks like

```
Project:Subproject/Repository/Architecture
```

Examples:

```
Devel:Science/SLES-11/i586
home:king:test/Leap42/x86_64
```

If a repository match a string the adjustment is added to the current value. The final value is the sum of the adjustments of all matched entries. This sum is the same logarithmic scale factor as described in the previous section.

Example dispatch_adjust definition in the BSConfig.pm

```
our $dispatch_adjust = [
    'Devel:' => 7,
    'HotFix:' => +20,
    '.*:test.*' => -10,
    'home:' => -3,
    'home:king' => +30,
    '.*SLE12-SP2' => -40,
];
```

The above example could have the following background: All Devel projects should get some higher priority so the developer jobs getting more build time. The projects under HotFix are

very important fixes for customers and so they should get a worker as soon as possible. All projects with test in the name get some penalty, also home projects are getting only about half of the build time as a normal project, with the exception of the home project from king, the user account of the boss. The SLES12-SP2 repository is not in real use yet, but if here is nothing else to do build for it as well.

Important

The dispatcher calculates the values from the '**dispatch_adjust**' array first, if the same project and repository also has an entry in the dispatchprios XML file, the XML file entry will overwrite the calculated priority. The best practice is to only use one of the methods.

3.5 Publisher Hooks

The job of the publisher service is to publish the build packages and/or images by creating repositories that are made available through a web server.

It can be configured to use custom scripts to copy the build results to different servers or do anything with them that comes to mind. These scripts are called **publisher hooks**.

3.5.1 Publisher Hooks configuration

Hooks are configured via the configuration file `/usr/lib/obs/server/BSConfig.pm`, where one script per project is linked to the repository that should be run if the project/repository combination is published. It is possible to use regular expressions here.

The script is called by the **obsrun** user with the following parameters:

1. information about the project and its repository (e.g. *training/SLE11-SP1*)
2. path to published repository (e.g. */srv/obs/repos/training/SLE11-SP1*)
3. changed packages (e.g. *x86 64/test.rpm x86 64/utils.rpm*)

The hooks are configured by adding a hash reference named `$publishedhook` to the `BSConfig.pm` configuration file. The key contains the project, and the value references the accompanying

script. If the value is written as an array reference it is possible to call the hook with self-defined parameters.

The publisher will add the 3 listed parameters at the end (after the self defined parameters).

```
/usr/lib/obs/server/BSConfig.pm

our $publishedhook = {
    "Product/SLES12"      => "/usr/local/bin/script2run_sles12",
    "Product/SLES11-SP3" => "/usr/local/bin/script2run_sles11",
    "Product/SLES11-SP4" => "/usr/local/bin/script2run_sles11",
};
```

Regular expressions or substrings can be used to define a script for more than one repository in one project. The use of regular expressions has to be activated by defining *\$publishedhook use regex = 1*; as follows:

```
/usr/lib/obs/server/BSConfig.pm

our $publishedhook_use_regex = 1;
our $publishedhook = {
    "Product\SLES12"      => "/usr/local/bin/script2run_sles12",
    "Product\SLES11.*"    => "/usr/local/bin/script2run_sles11",
};
```

With self defined parameters:

```
our $publishedhook_use_regex = 1;
our $publishedhook = {
    "Product\SLES11.*" => ["/usr/local/bin/script2run", "sles11", "/srv/www/
public_mirror"],
};
```

The configuration is read by the publisher at startup only, so it has to be restarted after configuration changes have been made. The hook script's output is not logged by the publisher and should be written to a logfile by the script itself. In case of a broken script, this is logged in the publisher's logfile (/srv/obs/log/publisher.log by default):

```
Mon Mar 7 14:34:17 2016 publishing Product/SLES12
    fetched 0 patterns
    running createrepo
    calling published hook /usr/local/bin/script2run_sles12
```

```
/usr/local/bin/script2run_sles12 failed: 65280  
syncing database (6 ops)
```

Interactive scripts are not working and will fail immediately.

If you need to do a lot of work in the hook script and don't want to block the publisher all the time, you should consider using a separate daemon that does all the work and just gets triggered by the configured hook script.

The scripts are called without a timeout.

3.5.2 Example Publisher Scripts

3.5.2.1 Simple Publisher Hook

The following example script ignores the packages that have changed and copies all RPMs from the repository directory to a target directory:

```
#!/bin/bash  
OBSHOME="/srv/obs"  
SRC_REPO_DIR="$OBSHOME/repos"  
LOGFILE="$OBSHOME/log/reposync.log"  
$DST_REPO_DIR="/srv/repo-mirror"  
# Global substitution! To handle strings like Foo:Bar:testing - two  
#+double-colons!  
PRJ_PATH=${1//:/\}  
PATH_TO_REPO=$2  
rsync -a --log-file=$LOGFILE $PATH_TO_REPO/ $DST_REPO_DIR/$PRJ_PATH/
```

For testing purposes, it can be invoked as follows:

```
$ sudo -u obsrun /usr/local/bin/publish-hook.sh Product/SLES11-SP1 \  
/srv/obs/repos/Product/SLE11-SP1
```

3.5.2.2 Advanced Publisher Hook

The following example script reads the destination path from a parameter that is configured with the hook script:

```
#!/bin/bash
LOGFILE="/srv/obs/log/reposync.log"
DST_REPO_DIR=$1
# Global substition! To handle strings like Foo:Bar:testing - two
#+double-colons!
PRJ_PATH=${2//:/\}
PATH_TO_REPO=$3
mkdir -p $DST_REPO_DIR/$PRJ_PATH
rsync -a --log-file=$LOGFILE $PATH_TO_REPO/ $DST_REPO_DIR/$PRJ_PATH/
```

For testing purposes, it can be invoked as follows:

```
$ sudo -u obsrun /usr/local/bin/publish-hook.sh \
    /srv/www/public_mirror/Product/SLES11-SP1 \
    /srv/obs/repos/Product/SLE11SP1
```

The following example script only copies packages that have changed, but does not delete packages that have been removed:

```
#!/bin/bash

DST_REPO_DIR=$1
PRJ_PATH=${2//:/\}
PATH_TO_REPO=$3
shift 3

mkdir -p $DST_REPO_DIR/$PRJ_PATH

while [ $# -gt 0 ]
do
    dir=(${1//\// })
    if [ ! -d "$DST_REPO_DIR/$PRJ_PATH/$dir" ]; then
        mkdir -p $DST_REPO_DIR/$PRJ_PATH/$dir
    fi
    cp $PATH_TO_REPO/$1 $DST_REPO_DIR/$PRJ_PATH/$1
    shift
done

createrepo $DST_REPO_DIR/$PRJ_PATH/.
```

For testing purposes, it can be invoked as follows:

```
$ sudo -o obsrun /usr/local/bin/publish-hook.sh /srv/www/public_mirror \
```

```
Product/SLES11-SP1 /srv/obs/repos/Product/SLE11-SP1 \  
src/icinga-1.13.3-1.3.src.rpm x86_64/icinga-1.13.3-1.3.x86_64.rpm \  
x86_64/icinga-devel-1.13.3-1.3.x86_64.rpm
```

3.6 Unpublisher Hooks

The job of the publisher service is to publish the build packages and/or images by creating repositories that are made available through a web server.

The OBS Publisher can be configured to use custom scripts to be called whenever already published packages get removed. These scripts are called **unpublisher hooks**. **Unpublisher hooks** are run before the **publisher hooks**.

3.6.1 Unpublisher Hooks configuration

Hooks are configured via the configuration file `/usr/lib/obs/server/BSConfig.pm`, where one script per project is linked to the repository that should be run if the project/repository combination is removed. It is possible to use regular expressions here.

The script is called by the **obsrun** user with the following parameters:

1. information about the project and its repository (e.g. *training/SLE11-SP1*)
2. repository path (e.g. */srv/obs/repos/training/SLE11-SP1*)
3. removed packages (e.g. *x86 64/test.rpm x86 64/utils.rpm*)

The hooks are configured by adding a hash reference named `$unpublishedhook` to the `BSConfig.pm` configuration file. The key contains the project and the value references the accompanying script. If the value is written as an array reference, it is possible to call the hook with custom parameters.

The publisher adds the three listed parameters at the end, right behind the custom parameters.

```
/usr/lib/obs/server/BSConfig.pm  
  
our $unpublishedhook = {  
    "Product/SLES12"    => "/usr/local/bin/script2run_sles12",
```

```

    "Product/SLES11-SP3" => "/usr/local/bin/script2run_sles11",
    "Product/SLES11-SP4" => "/usr/local/bin/script2run_sles11",
};

```

Regular expressions or substrings can be used to define a script for more than one repository in one project. The use of regular expressions needs to be activated by defining *\$unpublishedhook use regex = 1*:

```

/usr/lib/obs/server/BSConfig.pm

our $unpublishedhook_use_regex = 1;
our $unpublishedhook = {
    "Product\SLES12"      => "/usr/local/bin/script2run_sles12",
    "Product\SLES11.*"    => "/usr/local/bin/script2run_sles11",
};

```

With custom parameters:

```

our $unpublishedhook_use_regex = 1;
our $unpublishedhook = {
    "Product\SLES11.*" => [
        "/usr/local/bin/script2run", "sles11", "/srv/www/public_mirror"
    ],
};

```

The configuration is read by the publisher at startup only, so it has to be restarted after configuration changes have been made. The hook script's output is not logged by the publisher and should be written to a logfile by the script itself. In case of a broken script, this is logged in the publisher's logfile (/srv/obs/log/publisher.log by default):

```

Mon Mar  7 14:34:17 2016 publishing Product/SLES12
    fetched 0 patterns
    running createrepo
    calling unpublished hook /usr/local/bin/script2run_sles12
    /usr/local/bin/script2run_sles12 failed: 65280
    syncing database (6 ops)

```

Interactive scripts are not working and will fail immediately.

If you need to do a lot of work in the hook script and don't want to block the publisher all the time, consider using a separate daemon that does all the work and just gets triggered by the configured hook script.

The scripts are called without a timeout.



Note

Reminder: If *unpublish hooks* and *publish hooks* are defined, the *unpublish hook* runs before the *publish hook*.

3.6.2 Example Unpublisher Scripts

3.6.2.1 Simple Unpublisher Hook

The following example script deletes all packages from the target directory that have been removed from the repository.

```
#!/bin/bash
OBSHOME="/srv/obs"
LOGFILE="$OBSHOME/log/reposync.log"
DST_REPO_DIR="/srv/repo-mirror"
# Global substitution! To handle strings like Foo:Bar:testing - two
#+double-colons!
PRJ_PATH=${1//:/\}
PATH_TO_REPO=$2

shift 2

while [ $# -gt 0 ]
do
    rm -v $DST_REPO_DIR/$PRJ_PATH/$1 >>$LOGFILE 2>&1
    shift
done
```

For testing purposes, it can be invoked as follows:

```
$ sudo -u obsrun /usr/local/bin/unpublish-hook.sh \  
    Product/SLES11-SP1 \  
    /srv/obs/repos/Product/SLE11-SP1 \  
    src/icinga-1.13.3-1.3.src.rpm \  
    x86_64/icinga-1.13.3-1.3.x86_64.rpm \  
    \
```

```
x86_64/icinga-devel-1.13.3-1.3.x86_64.rpm
```

3.6.2.2 Advanced Unpublisher Hook

The following example script reads the destination path from a parameter that is configured via the hook script:

```
#!/bin/bash
OBSHOME="/srv/obs"
LOGFILE="$OBSHOME/log/reposync.log"
DST_REPO_DIR=$1
# Global substitution! To handle strings like Foo:Bar:testing - two
#+double-colons!
PRJ_PATH=${1//:/:/}
PATH_TO_REPO=$2

shift 3

while [ $# -gt 0 ]
do
    rm -v $DST_REPO_DIR/$PRJ_PATH/$1 >>$LOGFILE 2>&1
    shift
done
```

For testing purposes, it can be invoked as follows:

```
$ sudo -u obsrun /usr/local/bin/unpublish-hook.sh \  
    /srv/www/public_mirror/Product/SLES11-SP1      \  
    /srv/obs/repos/Product/SLE11SP1               \  
    src/icinga-1.13.3-1.3.src.rpm                  \  
    x86_64/icinga-1.13.3-1.3.x86_64.rpm           \  
    x86_64/icinga-devel-1.13.3-1.3.x86_64.rpm
```

3.7 User and group management

The OBS has an integrated user and group management with a role based access rights model. In every OBS instance at least one user need to exist and have the global Admin role assigned. Groups can be defined by the Admin and instead of adding a list of users to a project/package role user can be added to a group and the group will be added to a project or package role.

3.7.1 User and group roles

The OBS role model has one global role: Admin, which can be granted to users. An OBS admin has access to all projects and packages via the API interface and the web user interface. Some menus in the web ui do not allow changes by an Admin (e.g. the Repository Menu) as long as the Admin is not a Maintainer for the project as well. But the same change can be done via editing the meta data directly. The other roles are specific to projects and packages and can be assigned to an user or a group.

TABLE 3.2: OBS ROLES

Role	Description	Remarks
Maintainer	Read and write access to projects or packages	
Bugowner	Read access to projects or packages	should be unique per package
Reader	Read access to sources	
Downloader	Read access to the binaries	
Reviewer	Default reviewer for a package or project	

3.7.2 Standalone user and group database

OBS provides its own user database which can also store a password. The authentication to the api happens via HTTP BASIC AUTH. Please see the api documentation to find out how to create, modify or delete user data. Also a call for changing the password exists.

Users can be added by the maintainer or if registration is allowed via the registration menu on the Web UI. It can be configured that a confirmation is needed after registration before the user may login.

3.7.3 Proxy mode

The proxy mode can be used for esp. secured instances, where the OBS web server shall not get connected to the network directly. There are authentication proxy products out there which

do the authentication and send the user name via a HTTP header to OBS. Original this was developed for IChain - a legacy single login authentication method from Novell. This has also the advantage that the user password never reaches OBS.

The proxy mode can also be used for LDAP or Active Directory, but only for authentication.



Caution

With enabled proxy mode the OBS trust the username in the http header. Since this was verified by the Web server and the the Web server only forward requests for a verified and authenticated session, this is safe, as long you make sure that the direct web/API interface of the OBS is not reachable from the outside.

With the proxy mode the user still need to be registered in the OBS and all OBS roles and user properties are managed inside the OBS.

3.7.3.1 OBS proxy mode configuration

Currently the LDAP configuration is in the *option.yml* file.

TABLE 3.3: PROXY MODE CONFIGURATION OPTION

Config item	Description	Values <u>default</u>	Remarks
proxy_auth_mode	turn proxy mode on/ off	<u>:off</u> :on	need to be :off if ldap_mode: is :on

3.7.4 LDAP/Active Directory

Using LDAP or Active Directory as source for user and optional group information in environments which already have such a server has the advantage for the admin people that the user related information only need to be maintained in one place. In the following sections we are writing LDAP, but this includes Microsofts Active Directory as well. Only in parts where differences exists Active Directory (AD) will be explicit mentioned.

In this mode the OBS contact the LDAP server directly from the OBS API, if the user was found and provides the correct password the user is added transparently to the OBS user database. The password or password hash is not stored in the OBS database. Because the user database password field is mandatory, a random hash is stored instead. The LDAP interface allows to restrict the access to users which are in a special LDAP group. Optional also groups can be discovered from the LDAP server. This can be also filtered.

Before anybody can add a user to a package or project with a role, the user need to had logged in at least one time, since the check for available users is local only. If the LDAP group mode is enabled, LDAP groups are also added transparently, if a existing group on the LDAP server is added to a project or package.

On bigger installations this mode can result in many search requests to the LDAP server and slow down the the access to projects and packages, because on every role check a LDAP search operation will contact the LDAP server. As alternative method group mirroring was implemented. This allows that the internal OBS group database is updated with the group membership information during the user authentication. All role test are made local against the OBS database and do not need additional LDAPOperations.



Note

The local user group membership in :mirror mode is updated as follows: When the user logs in, the user *memberOf* attributes are parsed and compared with the global OBS grouplist, if a group match the user is added, if he does not longer be a group member, he is removed. since this maybe a costly operation, depending on the group counts, this is only done on a full login. After a full login the user status is cached for 2 minutes, if the user do a login during this time, nothing will be checked or updated. Here is a second mechanism to update user membership: If somebody adds a new Group in the OBS, the *member* attributes of the group are parsed and all current users which are in the local database become members.

3.7.4.1 OBS LDAP configuration

Currently the main OBS LDAP configuration is in the *option.yml* file. Beside the settings here also the *openldap* config file is evaluated by the Ruby LDAP implementation. This configfile is usually located at */etc/openldap/ldap.conf*. You can set here additional TLS/SSL directives like

TLS_CACERT, **TLS_CACERTDIR** and **TLS_REQCERT**. For more information refer to the `openldap` `ldap.conf` man page.

TABLE 3.4: LDAP CONFIGURATION OPTIONS

Config item	Description	Values default	Remarks
<code>ldap_mode</code>	OBS LDAP mode on/off	<u>:off</u> :on	
<code>ldap_servers</code>	List of LDAP servers		colon seperated list
<code>ldap_max_attempts</code>	tries to ping LDAP server	int <u>15</u>	
<code>ldap_search_timeout</code>	timeout of a LDAP search	int 0...N <u>5</u>	0 wait for ever
<code>ldap_user_memberof_attr</code>	User attribute for Group membership	<u>memberOf</u>	case senitive
<code>ldap_group_member_attr</code>	Group attribute for members	<u>member</u>	
<code>ldap_ssl</code>	use ldaps port and protocol	<u>:off</u> :on	
<code>ldap_start_tls</code>	usr Start TLS on ldap protocol	:off <u>:on</u>	
<code>ldap_port</code>	LDAP portnumbers		if not set 389 for ldap, 636 for ldaps
<code>ldap_referrals</code>	Windows 2003 AD requires	<u>:off</u> :on	
<code>ldap_search_base</code>	company's ldap search base for the users who will use OBS	<u>none</u>	
<code>ldap_search_attr</code>	user ID attribute	<u>sAMAccountName</u> uid	sAMAccountName for AD, uid for openldap

Config item	Description	Values <u>default</u>	Remarks
ldap_name_attr	Full user name	<u>cn</u>	
ldap_mail_attr	Attribute for users email	<u>mail</u>	
ldap_search_user	Bind user for LDAP search		e.g. cn=ldapbind, ou=system, dc=mycompany, dc=com
ldap_search_auth	Password for the ldap_search_user		
ldap_user_filter	Search filter for OBS users		e.g. a group membership, empty all users allowed
ldap_authenticate	How user how the credentials are verified	<u>:ldap</u> :local	only use :ldap
ldap_auth_mech	Used auth mech	<u>:md5</u> :cleartext	only if local
ldap_auth_attr	Used auth attribute for :local	<u>userPassword</u>	do not use
ldap_update_support	Update password or other user setting on the LDAP server	:on <u>:off</u>	do not use
ldap_object_class	Object class for user	<u>inetOrgPerson</u>	only for ldap_update_support: :on
ldap_entry_base	Base dn for the new added entry		only for ldap_update_support: :on
ldap_sn_attr_required	Does sn attribute required	:off <u>:on</u>	only for ldap_update_support: :on

Config item	Description	Values <u>default</u>	Remarks
ldap_group_support	Import OBS groups from LDAP	<u>:off</u> :on :mirror	see text
ldap_group_search_base	company's ldap search base for groups		
ldap_group_title_attr	Attribute of the group name	<u>cn</u>	
ldap_group_objectclass_attr	Object class for group	<u>Group</u>	
ldap_obs_admin_group	Group name for OBS Admins		if set, members of that group become OBS admin role

Example LDAP section of the *option.yml* file:

```
...
#####
# LDAP options
#####

ldap_mode: :on
# LDAP Servers separated by ':'.
# OVERRIDE with your company's ldap servers. Servers are picked randomly for
# each connection to distribute load.
ldap_servers: ldap1.mycompany.com:ldap2.mycompany.com

# Max number of times to attempt to contact the LDAP servers
ldap_max_attempts: 15

# timeout of a ldap search requests to avoid infinitely lookups (in seconds, 0 no
# timeout)
ldap_search_timeout: 5

# The attribute the user member of is stored in (case sensitive !)
ldap_user_memberof_attr: memberOf

# Perform the group_user search with the member attribute of group entry or memberof
# attribute of user entry
```

```

# It depends on your ldap define
# The attribute the group member is stored in
ldap_group_member_attr: member

# If you're using ldap_authenticate=:ldap then you should ensure that
# ldaps is used to transfer the credentials over SSL or use the StartTLS extension
ldap_ssl: :on

# Use StartTLS extension of LDAP
ldap_start_tls: :off

# LDAP port defaults to 636 for ldaps and 389 for ldap and ldap with StartTLS
#ldap_port:
# Authentication with Windows 2003 AD requires
ldap_referrals: :off

# OVERRIDE with your company's ldap search base for the users who will use OBS
ldap_search_base: ou=developmentt,dc=mycompany,dc=com
# Account name attribute (sAMAccountName for Active Directory, uid for openLDAP)
ldap_search_attr: sAMAccountName
# The attribute the users name is stored in
ldap_name_attr: cn
# The attribute the users email is stored in
ldap_mail_attr: mail
# Credentials to use to search ldap for the username
ldap_search_user: "cn=ldapbind,ou=system,dc=mycompany,dc=com"
ldap_search_auth: "top secret"

# By default any LDAP user can be used to authenticate to the OBS
# In some deployments this may be too broad and certain criteria should
# be met; eg group membership
#
# To allow only users in a specific group uncomment this line:
ldap_user_filter: (memberof=cn=obsusers,ou=groups,dc=mycompany,dc=com)
#
# Note this is joined to the normal selection like so:
# (&({dap_search_attr}={login})#{ldap_user_filter})
# giving an ldap search of:
# (&(sAMAccountName={login})(memberof=CN=group,OU=Groups,DC=Domain Component))
#
# Also note that openLDAP must be configured to use the memberOf overlay

# ldap_authenticate says how the credentials are verified:
# :ldap = attempt to bind to ldap as user using supplied credentials
# :local = compare the credentials supplied with those in

```

```

#         LDAP using #{ldap_auth_attr} & #{ldap_auth_mech}
#         if :local is used then ldap_auth_mech can be
#         :md5
#         :cleartext
ldap_authenticate: :ldap
ldap_auth_mech: :md5
# This is a string
ldap_auth_attr: userPassword

# Whether to update the user info to LDAP server, it does not take effect
# when ldap_mode is not set.
# Since adding new entry operation are more depend on your slapd db define, it might
not
# compatible with all LDAP server settings, you can use other LDAP client tools for
your specific usage
ldap_update_support: :off
# ObjectClass, used for adding new entry
ldap_object_class: inetOrgPerson
# Base dn for the new added entry
ldap_entry_base: ou=OBSUSERS,dc=EXAMPLE,dc=COM
# Does sn attribute required, it is a necessary attribute for most of people
objectclass,
# used for adding new entry
ldap_sn_attr_required: :on

# Whether to search group info from ldap, it does not take effect it is not set
# Please also set below ldap_group_* configs correctly to ensure the operation works
properly
# Possible values:
#         :off      disabled
#         :on       enabled; every group member operation ask the LDAP server
#         :mirror   enabled; group membership is mirrored and updated on user login
#
ldap_group_support: :mirror

# OVERRIDE with your company's ldap search base for groups
ldap_group_search_base: ou=obsgroups,dc=mycompany,dc=com

# The attribute the group name is stored in
ldap_group_title_attr: cn

# The value of the group objectclass attribute
# group for Active Directory, groupOfNames in openLDAP
ldap_group_objectclass_attr: group

```

```
# The LDAP group for obs admins
# if this group is set and a user belongs to this group he get the global admin role
#
ldap_obs_admin_group: obsadmins
```

3.7.5 Authentication methods

3.7.5.1 LDAP methods

The LDAP mode has 2 methods implemented to check authorization:

1. LDAP bind method. With the provided credentials a LDAP bind request is tried.
2. Local method. The provided credentials checked locally against the content of the of the *userPassword* attribute.



Caution

The local method should be not used, since the *userPassword* attribute in most LDAP installations will not be available until you are bind with a priviledge user.

3.7.5.2 Kerberos

For connecting the API you can use single sign on via kerberos tickets.

- Need more information **

3.7.5.3 OBS Token Authorization

OBS 2.5 provides a mechanism to create tokens for specific operations. This can be used to allow certain operations in the name of a user to others. This is esp. useful when integrating external infrastructure. The create token should be kept secret by default, but it can also be revoked at any time if it became obsolete or leaked.

3.7.5.3.1 Manage tokens of a user

Tokens belong always to a user. A list of active tokens can be received via

```
osc token
```

```
osc token --delete <TOKEN>
```

3.7.5.3.2 Execute a source service

A token can be used to execute a source service. The source service has to be setup for the package first, check the source service chapter for this. A typical example is to update sources of a package from git. A source service for that can be setup with

```
osc add git://....
```

A token can be registered as generic token, means allowing to execute all source services in OBS if the user has permissions. You can create such a token and execute operation with

```
osc token --create
```

```
osc token --trigger <TOKEN> <PROJECT> <PACKAGE>
```

```
osc api -X POST /trigger/runservice?  
token=<TOKEN>&project=<PROJECT>&package=<PACKAGE>
```

You can also limit the token to a specific package. The advantage is that the operation is limited to that package, so less bad things can happen when the token leaks. Also you do not need to specify the package on execution time. Create and execute it with

```
osc token --create <PROJECT> <PACKAGE>
```

```
osc token --trigger <TOKEN>
```

```
osc api -X POST /trigger/runservice?token=<TOKEN>
```

3.8 Backup

4 Troubleshooting


Here are two major classes of problems regarding the **Open Build Service**

1. Normal package build errors
2. Bugs, resource shortage or config issues caused issues

The first category are errors like missing dependend packages in the build environment, errors during compiling or linking, errors in the buid description and so on. Most of them should not happen if the packager does test the build locally before committing it to the OBS. This type of problems is not covered by this chapter.

4.1 General hints

If you detect unexpected behavior of the open build service, you should follow some rules to locate the problem:

1. Consult the logfiles, for the backend look at `/srv/obs/log` for the backend log files and `/srv/www/obs/api/log` for the frontend log files. See the Log files [Section 2.2, "Log files"](#) for more details.
2. Consult the normal OS system logs and the kernel log (dmesg) if here are reported system or HW problems.
3. Check if all services are running on the backend and frontend. See the OBS Architecture in reference book for details.
4. Try to find an easy way to reproduce the problem.
5. Look on <https://github.com/openSUSE/open-build-service>  if this issue was already reported.
6. Use search machines (Google) to find out if others did also run into this problem. If you are lucky, you will find a fix or workaround as well.
7. If you create a new bug report, include all information to reproduce the problem and the complete error message/error log if here are any.

4.2 Debug frontend problems

If you get unexpected results from submitting commands with the **osc** tool, you can use the debug feature of the tools to get some more informations what happen.

osc debug options

<code>--debugger</code>	jump into the debugger before executing anything
<code>--post-mortem</code>	jump into the debugger in case of errors
<code>-t, --traceback</code>	print call trace in case of errors
<code>-H, --http-debug</code>	debug HTTP traffic (filters some headers)
<code>--http-full-debug</code>	debug HTTP traffic (filters no headers)
<code>-d, --debug</code>	print info useful for debugging

The **--debugger** and **--post-mortem** are only suitable for *osc* developers. If you get a error message from *osc* the **-t, --traceback** can give the developer some more information about the problem. The **-H, --http-debug** and **--http-full-debug** options are useful to see the raw answers of OBS API, often this gives a hint what maybe wrong. If you report a problem regarding the **osc** tool, it may help to include the *osc* output with **additional *--http-debug --traceback** options.



Warning

With **--http-full-debug** all http headers are included, this may include user data and authentication stuff so review and replace such data with **XXXXXXXXX** or so before you post it on the internet.

A GNU Licenses

This appendix contains the GNU General Public License version 2 and the GNU Free Documentation License version 1.2.

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under

copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a). You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b). You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c). If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the

Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a). Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b). Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c). Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it.

However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author
```

```
This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307,
USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.
```

```
signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the [GNU Lesser General Public License \(http://www.fsf.org/licenses/lgpl.html\)](http://www.fsf.org/licenses/lgpl.html) instead of this License.

GNU Free Documentation License

Version 1.2, November 2002

Copyright (C) 2000,2001,2002 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none. The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four

years before the Document itself, or if the original publisher of the version it refers

to gives permission.

- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.