Wacs Installation Guide

Tenth Edition

for WACS 1.0.0

B "Beaky" King

Publication date 29th June 2020

Wacs Installation Guide

by B "Beaky" King

for WACS 1.0.0

Publication date 29th June 2020

Abstract

WACS is a tool for building Adult Web Sites; it is equally suitable for managing a private collection or building a commercial web site. It has many best of breed features including dynamic filtering, model catalogs, automatic download and powerful search engine. It comes with a powerful API (application programming interface) implemented in both Perl and PHP5 languages to allow web developers to leverage it's facilities from their own programs.

This book describes the actions required to install the WACS System onto a suitable host system (typically a server). The intended audience is system administrators and prospective WACS site managers wishing to install WACS on a machine. The guide will try to cover three types of installation: Packaged installation (using RPMs on Fedora/RHEL/CentOS, DEBs on Ubuntu/Debian), installation onto a WebHost environment and manual installation on other Unix/Linux versions.

The WACS source code and other documentation and support tools can all be found at the WACS website at Sourceforge [http://wacsip.sourceforge.net/] and at the Wacs page on launchpad.net [https://launchpad.net/wacs]. A demonstration site built using the WACS tools can be found at PinkMetallic.com [http://www.pinkmetallic.com/] - *CAUTION: contains adult material*. Commercial add-ons and support options can be purchased from Bevtec Communications Ltd, see their website at Bevtec Communications [http://www.bevteccom.co.uk/].

Table of Contents

List of Tables

List of Examples

Chapter 1. Introduction to WACS Installation

WACS Overview

Welcome to WACS, Web-based Adult Content Server, a free software package for the management of material of an "Adult Nature" (or basically whatever euphermism for porn you prefer). It is web-based and can be used for the management of an existing collection, as a download manager, or as a back-end system for running a commercial adult web site. It is dramatically different from most other image gallery systems in that it understands photo sets and video clips as basic concepts, instead of single photographs. It also includes far more specialised tagging, source, relationship and attribute marking concepts than other more generalised systems.

WACS is extremely configurable, making extensive use of configuration files written in eXtensible Markup Language (XML). This book is one of a collection of manuals we have created to help you through the various aspects of using a system as complex as WACS. This guide is solely targeted to installation - additional guides exist for Users, Configuration, Administration and Programming.

About This Book

This book is an installation guide for WACS site managers or system administrators seeking to install the WACS environment on their systems. It does assume a certain amount of familiarity with the normal processes of installing software packages on your systems; the sections on manual installation in particular also assume a basic knowledge of using the Unix operating system (or any other future supported OS platform). It also provides an overview to installing the WACS environment on a web hosting site using an environment like cPanel.

To get the best from this book, you should ideally be familiar with the basic user interface of the WACS applications themselves - the WACS User Guide would be an ideal primer for this and should introduce you to many of the concepts and tools being used here. There is also no substitute for using a real WACS site to get a general feel for how things work and are laid out. Our demonstration site is available at PinkMetallic.com [http://www.pinkmetallic.com/] and provides an opportunity to try WACS hands-on.

Goals

The task of installing WACS onto a new server system consists of a number of distinct steps; these are:

- Preparing the host system
- Installing the pre-requisite software
- Installing the WACS applications and modules
- Getting a working configuration
- Installing some initial data

Some of these topics will be mentioned briefly here and will be covered in more depth in other guides in the WACS documentation set.

About The Examples

For copyright/licensing reasons, the example images feature sets from photoshoots by the main developer of WACS (Beaky) and a friend of his. These sets are available at our demonstration site, PinkMetallic.com [http://www.pinkmetallic.com/] where you can experience WACS in action. Currently access to PinkMetallic.com [http://www.pinkmetallic.com/] is free, but we may at some point in the future make a small charge for access if it doesn't receive the revenue we hope for from referrals.

Chapter 2. Preparation For Installation

Preparation Tasks

Before we even start to install the WACS package, it is very important that we make sure the host candidate system is prepared for the task in hand. To do this, we need to ensure a number of things have been prepared beforehand:

- ensure adequate system resources
- assign and configure for static host name
- review security and access policies

The first of these steps, ensuring adequate system resources, basically involves looking at the sort of material you're intending to store in the WACS system and approximately what the storage requirements will be. If you are looking at holding sets for maybe fifty models who come from a site that specialises in high-resolution images and HD video clips, you may find that an average image set is upwards of 100MB, and an average video clip maybe 500MB-1GB (if you are planning to go for high resolution 4K UHD video clips, expect more like 5GB-10GB per clip). If each model has an average of four video clips and 10 sets, then you're looking at probably 8GB per model, and would need to allocate around 250GB of storage, which with margins for future expansions means about 350-500GB to start off with.

Do remember that on most Linux systems you can use tools like the Logical Volume Manager (LVM) to ease the process of disc space allocation and in particular future expansion when live data is present. It is also perfectly possible to use Network Attached Storage (NAS) devices as the primary storage location for WACS collections.

You also need to make sure you assign a static IP address and hostname to the server system; more details on this and the use of NAS servers is given in the configuration guide. There are also a number of resources on the net to help you through this process - you want one targeted at setting up a web server; configuration management services like cPanel are not a pre-requisite for running WACS so you can leave out those steps if you don't need them for other reasons.

Warning

WACS is not currently compatible with the SELinux enhanced security system - this needs to be reduced to either **permissive** or switched off entirely (**disabled**) for WACS to work. This will affect Fedora and other RedHat-based distributions. It was our intention to resolve this issue in the *1.x* release series of WACS but it now looks likely that we will not fully address this issue until someone chooses to fund that work.

If you're running Fedora (or any other distribution) with SELinux enabled, you will run into problems. WACS does not currently work well with SELinux and you have a choice of either setting it to permissive mode (where it logs problems but does not block things from working) or disabling it entirely. If you disable it entirely, it is much harder to go back to running it later as software updates and the like to not get their SELinux attributes updated. On the other hand, permissive mode will fill up your log file areas and may slow down system operation somewhat.

Linux Operating Systems

If you are using either the RPM or DEB packages of the WACS environment and are using the default applications (MySQL for the standard packages, PostgreSQL for the `wacs-for-psql` packages), the prerequisite applications will be automatically installed if they are not already present. If not, or you are using a different database (Oracle for instance), you will need to install these applications first as detailed in the table below and then follow the manual install steps:

Table 2.1. Software Pre-Requisites For WACS On Linux

Service	Application	Version	Description
Web Server	Apache	> 2.0	main route of access
Database	MySQL	> 5.0	backend database engine
	Oracle	> 10g	alternative database engine
	PostgreSQL	> 10	alternative database engine
Perl	Langauge	> 5.8.0	Langauge interpreter (required)
Php	Language	> 5	Language interpreter (optional)
Perl::DBI	Library	any recent	Database interface library
Perl::DBD	Driver	for Database	Database driver routine for MySQL or Oracle
XML::Simple	Library	any recent	Parsers for eXtensible Markup Langauge (XML) files
Data::Dumper	Library	any recent	Essential debugging tool
File::Basename	Library	any recent	Filename manipulation routines
MIME::Base64	Library	any recent	Binary data encoder used with XML files
netpbm	Converter	any recent	Image format conversion tools
Image::Exiftool	Library	any recent	Image/Video identification
ffmpegthumbnailer	Converter	any recent	Video thumbnailer

Note

Since ffmpegthumbnailer uses various plugin codecs, not all of which are always enabled by default, you may need to include additional non-free repositories and install additional codecs to make sure it works for all the video types you intend to manage on the WACS system. In essence, if you find thumbnails are not being generated correctly for a given video file, use a standard player (mplayer, xine, etc) to determine which codecs are missing and install them.

Chapter 3. Prerequisites

Web Server

WACS is primarily designed to work with the Apache 2 web server as this is the industry leading web server for Linux and Unix platforms. It's also available for the Mac OSX platform from various sources, and even for Microsoft Windows under the name WAMP Server. While other web servers may work fine, we would not recommend using them at this time and stage of WACS development. The RPM/DEB packages automatically drop in configuration file snippits to ensure correct operation of apache 2 on Linux releases with supported packages.

Warning

There are also important differences in the way that access control is configured between the older apache 2.2 release versions, and the newer apache 2.4 and greater versions. WACS includes code to support both versions and the correct version should be selected by the package during installation.

If you are using a web site hosting provider, most will be using Linux for their standard service and will be using a suitable version of Apache. Most will confirm this in their detailed features card before sign-up - if not, you can always ask their pre-sales support contacts. Do be aware that there are some limitations on using WACS on a web site hosting provider and the installation currently requires some manual intervention and editing. The use of the alternative database based user authentication method in WACS is much more suitable for using on a web site hosting provider than the host based authentication used by default in the standard packages.

Relational Database

You do need to be aware that the MySQL network layer appears to be extremely sensitive to what the host is called. It needs to have a permanent, static name which is correctly mapped in the hosts or DNS so that *hostname* maps to *ip address* and the *ip address* maps back to THE SAME *hostname*. If this isn't the case, the final part of the installation - creating the database schemas and populating them - may well not work.

On a web site hosting provider, you will need to work through whatever access configuration facilities they provide. The common **cpanel** administrative interface used by many provides a way to configure remote MySQL access although if you have SSH access, this may not be strictly necessary.

Other database engines may not be so sensitive to the IP address issues - Oracle 10 and 11 versions are not (by default anyway). We currently support MariaDB 10.2.x, PostgreSQL 10 and Oracle 10i/11i/12g in addition to MySQL > 5.x.

Content Storage

The normal location for content storage is the home directory of the WACS user account which is created when you do an install from packages. If you do a manual install, or use the obsolete easyinstall script, you need to check whether this has been created and if not, create it. Obviously putting a large amount of multi-media material into the home directory area of the server may not be desirable so you may wish to consider where it should be placed. As mentioned elsewhere this could be a seperate volume or group of volumes on an LVM partition, an external disc drive or even another remote server or NAS server supporting NFS protocols. These locations are configured in the `wacs.cfg` file and can easily be altered.

The important things are to ensure that the location of the storage is specified in the Wacs configuration file (usually `/etc/wacs.d/wacs.cfg` ahead of adding the first content and that it remains accessible at the same file system location throughout. This means either arranging for it to be mounted, etc as necessary at boot time or deploying an automounter such as **autofs** or **am-utils** to do that for you.

In the case of a WACS system on a web hosting provider, you are unlikely to be able to access the filespace via local network style file sharing mechanisms such as NFS and CIFS. You will have to upload content instead using **ftp**, **sftp** or web browser uploads and import it using WACS's extensive XML import facilities or the collection management tools.

Chapter 4. Linux: Options

Methods Available

With the Linux Operating System, there are three basic options available to you for installing WACS onto your system:

- Linux Package Installation
- Linux Manual Installation
- Linux easyinstall script (Depricated)

Each option above is progressively more complex than the previous one, but in the process affords more flexibility and configurability. The choice is yours.... but if you are not an experienced Linux system admin, we would strongly commend that you use the packages appropriate for your distribution. On other platforms, you probably want to follow the manual install proceedure which is documented in some depth. The older easyinstall script is now depricated and is likely to be removed in a future release. If you're on a supported distribution, use the packages - if not, follow the manual installation steps - it's honestly easier than trying to use a script that may easily make wrong decisions on your particular platform and leave you having to figure out where this went wrong.

If you are using a web hosting provider, unless they're offering pre-installed WACS, you will need to follow the manual installation steps. In order to do so, you will need to have SSH access to the server hosting your site, which may have an additional charge and setup delay associated with it. We've added additional commentary in the manual install section of the installation proceedure covering the steps you need to take when working on a web hosting provider.

Linux Package Install

Tip

This is the prefered way to install WACS. It is currently available for Fedora 27 or 29 based systems using the RPM package manager and for Ubuntu 16.04 LTS and 18.04 LTS systems using the DEB packaging system. It is our hope to extend the packaged software approach to include other platforms in a future release, CentOS 8.x being our next anticipated platform to be added.

Note

While we do not prepare packaged versions for older releases of Fedora and Ubuntu, we do not remove support from them from the packaging build files (`.spec` for Fedora, `debian/` for Ubuntu), so you should be able to recreate packages suitable for use with earlier releases using the package build tools for your platform.

Warning

We attempt to support the current and previous releases of both Fedora and Ubuntu when we make each new WACS release. Packaged support for previous releases is then dropped.

Where available for a given distribution and release, there are a number of WACS RPM or .deb packages you can make use of to install the WACS system. If you are using one of the more sophisticated package managers (dnf, gdebi etc), you need only ask it to install the main wacs package and that tells the package manager what other components it needs to complete the install. This will bring in both the system packages needed - web server, database, perl libraries, etc - and the other parts of the WACS system needed for a working installation. If you are using one of the simpler package managers (rpm, dpkg etc), it will complain about absense of the required packages until all the dependencies have been installed manually.

Since sourceforge.net doesn't yet seem to support rpm or debian repos properly you will have to download the requisite WACS packages manually in order for the install to proceed. We are working on a Personal Package Archive (PPA) area on launchpad.net for Ubuntu .DEB packages which we hope to launch shortly. Please watch the WACS web sites for more details.

Note

In order to conform to the the Fedora packaging guidelines, quite a few of the file locations are different on the packaged version of WACS, from that created by the easyinstall script or manual process. We are gradually migrating to the layout preferred by the packaging and it shouldn't cause problems, but you do need to be aware of it, particularly if moving a configuration file between releases.

Linux easyinstall (Depricated)

Warning

It is our intention to discontinue support for the **easyinstall** script in a release in the near future. You are strongly encouraged to use the other two methods, package install or manual install in preference.

The **easyinstall** script was our pre-packaging approach to installing WACS and we would advise against using it on Fedora and Ubuntu distributions, although it might prove useful for installing on older versions of Linux that are not supported directly by the packages we create. If attempting an install on RHEL or CentOS, try the Fedora packages first. For Debian, try the Ubuntu pacakages. As of WACS 1.0.0, the packages are quite mature and should be the best option. The **easyinstall** command may or may not still work. Additionally installations done with the **easyinstall** will prove harder to upgrade than those done with the packaged solutions.

All versions of WACS include the **wacssetup** configuration tool to provide an easy web-based setup interface replacing what **easyinstall** used to do.

Linux Manual Install

This the only option available for any kind of unsupported operating system platform and on a web site hosting provider where you have no *root* system administrator access and no control over the filesystem layout. The instructions later in this guide take you through all of the tasks needed step by step. This does assume some basic familiarity with command line operation of the Linux/Unix environment and a reasonable knowledge of the software installation policies of the operating system in question. In the case of a web hosting provider, you will need some familiarity with their setup tools and utilities.

Chapter 5. Installing WACS Using Packages

Package List

In order to install WACS using the packages, you need to download a number of separate packages from sourceforge and have them available for your choosen package manager to find. Make sure you pick the right one for your Linux distribution. The list below details what these packages are:

Table 5.1. List of Packages

Name	**Req'd**	**Description**
wacs	Either/Or	The "Master" package which includes the others, MySQL version
wacs-for-psql		The "Master" package which includes the others, PostgreSQL version
wacs-core	Yes	The core files and user interface apps
wacs-tools	Yes	The collection management tools
wacs-download	Opt	The tools used for automatic download from subscription sites - optional
wacs-hostauth	Yes	Tool used to authenticate users using the operating system of the server. This is not needed if you are using database authentication.
wacs-samples	Yes	Some sample data files in XML format and perl API programming examples
wacs-php	Opt	Implementation of the WACS API in the PHP5 language favoured by web developers - also includes Web 2.0 demo program.
wacs-php-skins-simple	Opt	A sample simple adult web site implemented using the WACS API for PHP5. Ready-to-use, customisable and a good source of example code.
wacs-doc-pdf	No	Documentation in PDF format
wacs-doc-html	No	Documentation in HTML format (both single and multi page)

As you can see from the above list, you might wish to download the core packages, plus the download tools, plus whichever format of documentation you prefer to use. Unless you're already familiar with WACS, we'd strongly recommend using the master wrapper package (`wacs` or `wacs-for-psql`) for the installation as it does a number of configuration steps for you. For the PostgreSQL version things are likely to go very wrong if you don't! [1] For the examples ahead, we'll assume PDF is the prefered format - your mileage may vary.

[1] This is due to the postinstall script for this package creating a password based DBA account needed by **wacssetup**.

RPM Installation Steps

Note

For Ubuntu/Debian DEB package installs, please see the section called “DEB Installation Steps” instead.

Important

Before you start on an installation, please make sure that you have a statically allocated IP address, sensible hostname with a fully qualified domain name and that the machine is fully aware of these settings. For more information on these aspects, please consult the configuration guide.

Downloading The RPMs

The first step obviously is to download the appropriate packages for the operating system release, version and processor platform that you intend to run it on. Where a package contains *noarch* that means that it is suitable for any processor architecture running that distribution of Linux. Currently RPM packaged versions are available for Fedora 27 (labeled fc27), and Fedora 29 (labeled fc29) and DEB packaged versions for Ubuntu 16.04 LTS and 18.04 LTS. For more information on using the Ubuntu DEB packaged versions, please see the section called “DEB Installation Steps”

For an initial WACS installation (in this example for release 1.0.0 on an x86_64 machine running Fedora 29), you will probably want the following packages substituting **wacs-for-psql** instead of the main **wacs** if you intend to use PostgreSQL as your backend database instead of MySQL:

- **wacs-1.0.0-1.noarch.fc29.rpm** OR **wacs-for-psql-1.0.0-1.noarch.fc29.rpm**
- **wacs-core-1.0.0-1.noarch.fc29.rpm**
- **wacs-tools-1.0.0-1.noarch.fc29.rpm**
- **wacs-samples-1.0.0-1.noarch.fc29.rpm**
- **wacs-hostauth-1.0.0-1.x86_64.fc29.rpm**

If you also wish to make use of the Wacs-PHP API, The AJAX enabled dynamic search tools, the Web 2.0 demo or the Simple Skin sample web site, you will also want the following files:

- **wacs-php-1.0.0-1.noarch.fc29.rpm**
- **wacs-php-skins-simple-1.0.0-1.noarch.fc29.rpm**

If you plan on making use of the download toolset to connect to subscription sites for automatic downloads (although do be aware that only a very few sites are still supported), you will also want to get the package called **wacs-download-1.0.0-1.noarch.fc29.rpm**. You may also wish to download one of the two versions of the documentation package: **wacs-doc-pdf-1.0.0-1.noarch.fc29.rpm** or **wacs-doc-html-1.0.0-1.noarch.fc29.rpm** - you can always access the same documentation direct from our sourceforge web site.

RPM Installation

Note

The packages **wacs** and **wacs-for-psql** are mutually exclusive - pick which one you want and download *ONLY* that version. They contain the same files and just have differences in the other packages they depend on and how the postinstall script that is part of them acts.

Once you've downloaded the right packages, you need to gain the appropriate privileges and install the packages. There are any number of ways to do this, and you can pretty much use any of them; the example below uses the command-line based yum package manager:

```
# dnf install wacs*.rpm
[...]
#
```

It is also possible to do this with the file manager, right clicking on each package file and choosing `Install Package`. The order on this is a bit tricky, but if you start with **wacs-core** and **wacs-hostauth**, then do the other packages and finally do the main wacs package, this should work out OK.

Other System Issues

Once the packages, and their dependencies, have been installed please confirm that both the Apache 2 Web Server (httpd) and the Database Server are enabled and running. These should be respectively **mysqld** for MySQL, **mariadb** or **mysqld** for MariaDB and **postgresql** for PostgreSQL. In the GNOME desktop, the System -> Administration -> Services menu will take you to the Service Configuration screen where you need to both enable and start **httpd** and **mysqld** (etc) if these are not shown as currently running. If you prefer using the command line, the following steps will do the same task:

```
# systemctl start httpd
Enter SSL pass phrase for nemisis.example.com:443 (RSA) : ********
# systemctl start mariadb
# systemctl enable httpd
# systemctl enable mariadb
Created symlink from /etc/systemd/system/multi-user.target.wants/mariadb.service t
#
```

On older systems (no longer supported by the packaged versions), you may need to use the following commands:

```
# /sbin/service httpd start
Starting httpd:                                      [ OK ]
# /sbin/service mysqld start
Starting MySQL:                                      [ OK ]
# /sbin/chkconfig --levels 345 httpd on
# /sbin/chkconfig --levels 345 mysqld on
#
```

The final system configuration step before starting work on getting WACS configured is to ensure that SELinux is running in a reduced mode that will not block the WACS components from working. This is only an issue on Fedora and other Red Hat based releases at present. You can determine the current mode of SELinux using the sestatus command:

```
% /usr/sbin/sestatus
SELinux status:                 disabled
%
```

To change the normal operational mode, you need to edit the file called `/etc/sysconfig/selinux` and change the line which reads `SELINUX=enabled` to either `SELINUX=permissive` (generates big log files and slows machine but allows for SELinux to be turned back on later more easily) or `SELINUX=disabled` (which disables it completely but can cause problems in the future if you want to switch it back on). You will also probably want to disable it immediately rather than doing a reboot before you can continue working on WACS - to do this, become root and run the following:

```
# /usr/sbin/setenforce 0
setenforce: SELinux is disabled
#
```

You can check this change has taken effect by using the **sestatus** command again.

DEB Installation Steps

Note

For Fedora/CentOS RPM package installs, please see the section called “RPM Installation Steps” instead. Although targetted for Ubuntu versions, these DEB packages should also be suitable for use on standard Debian distributions.

Important

Before you start on an installation, please make sure that you have a statically allocated IP address, sensible hostname with a fully qualified domain name and that the machine is fully aware of these settings. For more information on these aspects, please consult the configuration guide.

Downloading The DEBs

The first step obviously is to download the appropriate packages for the operating system release, version and processor platform that you intend to run it on. Where a package contains *all* that means that it is suitable for any processor architecture running that distribution of Linux. Currently DEB packaged versions are available for Ubuntu 16.04 LTS and 18.04 LTS, while RPM packaged versions are available for Fedora 27 and 29. For more information on using the Fedora RPM packaged versions, please see the section called “RPM Installation Steps”

For an initial WACS installation (in this example for release 1.0.0 on an x86_64 machine running Ubuntu 18.04 LTS) you will probably want the following packages remembering to choose whether you want the **wacs-for-psql** package for PostgreSQL database support instead of the default MySQL for the **wacs**:

- **wacs_1.0.0-1.all.deb** OR **wacs-for-psql_1.0.0-1.all.deb**

- **wacs-core_1.0.0-1.all.deb**
- **wacs-tools_1.0.0-1.all.deb**
- **wacs-samples_1.0.0-1.all.deb**
- **wacs-hostauth_1.0.0-1.amd64.deb**

If you also wish to make use of the Wacs-PHP API, The AJAX enabled dynamic search tools, the Web 2.0 demo or the Simple Skin sample web site, you will also want the following files:

- **wacs-php_1.0.0-1.all.deb**
- **wacs-php-skins-simple_1.0.0-1.all.deb**

If you plan on making use of the download toolset to connect to subscription sites for automatic downloads (although do be aware that only a very few sites are still supported), you will also want to get the package called **wacs-download_1.0.0-1.all.deb**. You may also wish to download one of the two versions of the documentation package: **wacs-doc-pdf_1.0.0-1.all.deb** or **wacs-doc-html_1.0.0-1.all.deb** - you can always access the same documentation direct from our sourceforge web site.

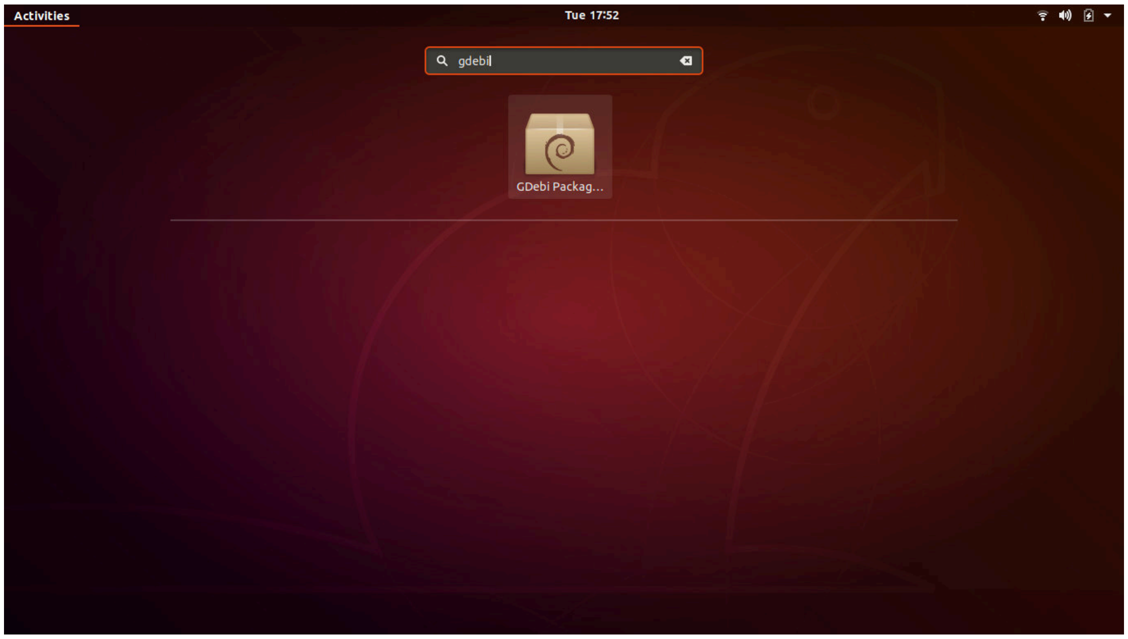
DEB Installation

Once you've downloaded the right packages, you need to do the package installations. There are any number of ways to do this, and you can pretty much use any of them; we recommend use of the **gdebi** package manager which is an optional package that you will almost certainly have to install first. This can be achieved using a number of the standard Ubuntu tools including **apt-get install**, **aptitude** or by using the Ubuntu Software Center. We have not used the Ubuntu software center, aptitude or synaptic package manager in these instructions because those only really cope with software provided by the main Debian repositories. The **gdebi** mixes it up allowing the installation of a locally downloaded DEB file, combining it any packages it depends on through the normal package repositories mechanism. Unfortunately sourceforge.net does not, as we are writing this manual, support the creation of debian style package repositories.

Note

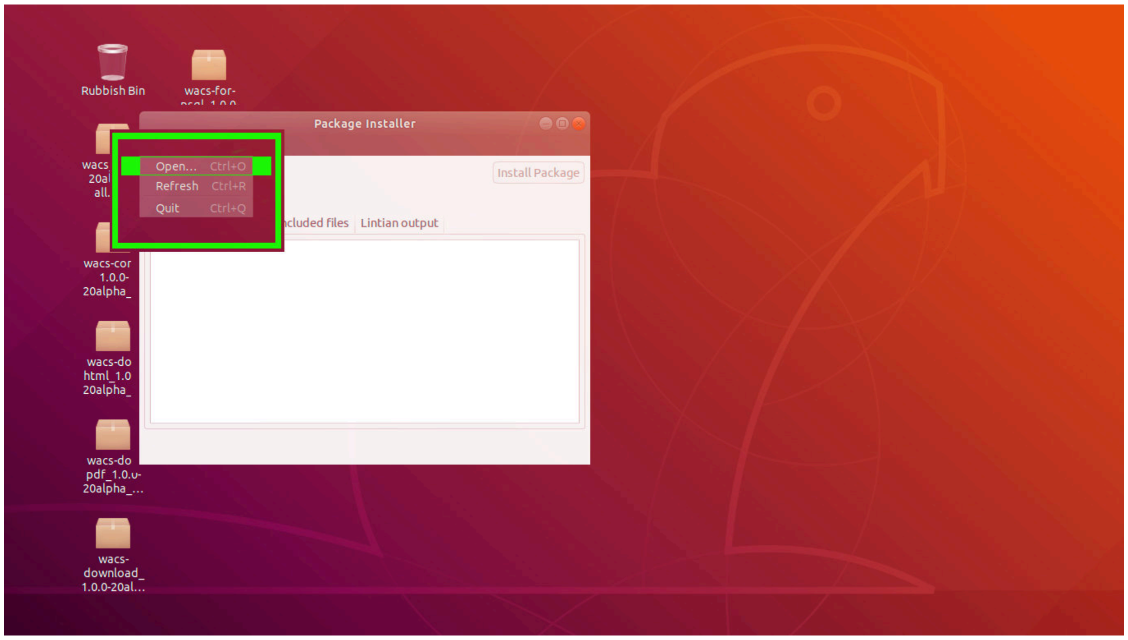
Additionally it is usual for **GDebi** to be configured as an action listed on the actions (right click) menu in the GNOME file manager as installed on Ubuntu for debian packages. You can use this method instead if you prefer.

Once you have installed gdebi, the easiest way to start it up is to click on the Meta (Windows) button and search for it as shown below. Using the file manager to browse to your downloaded files, and then right clicking on the files works too. We're going to show you the pure **gdebi** way so double click on the icon to start it running:

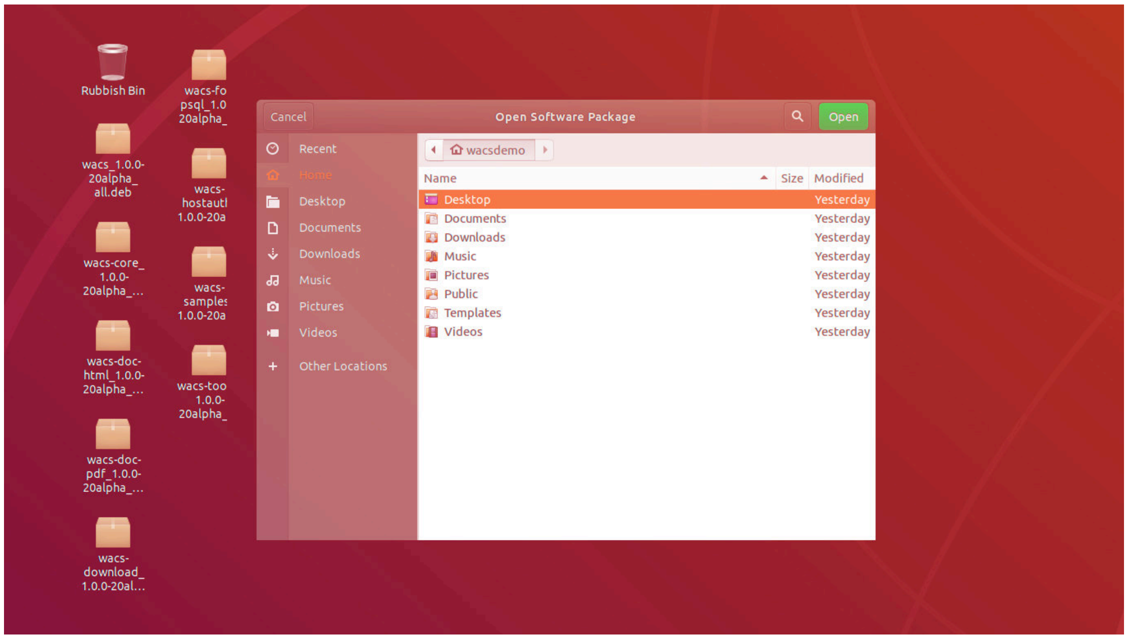


Once **gdebi** is running you use the *File* menu **Open** option to open each DEB package file in turn. The order on this is a bit tricky, but if you start with **wacs-core** and **wacs-hostauth**, then do the other packages and finally do the main **wacs** (or **wacs-for-psql**) package, this should work out OK.

Here we have all of the Wacs packages for Ubuntu/Debian downloaded onto our desktop and we start work with `wacs-core_1.0.0-1_all.deb` . We choose **File -> Open** in gdebi as highlighted in green below:

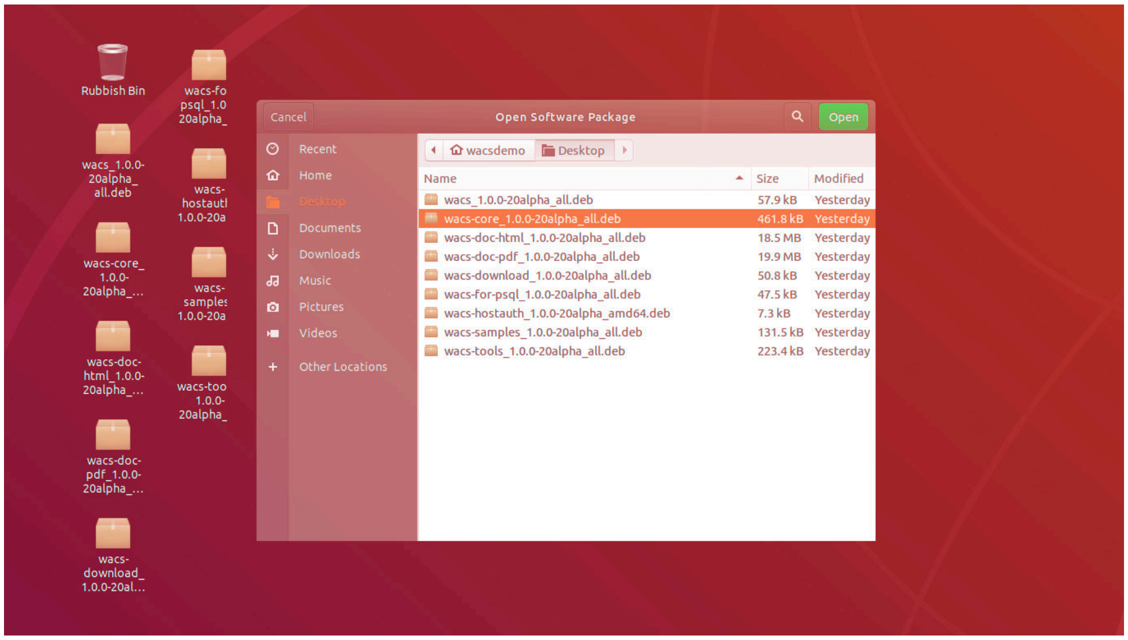


At this point the Open File browser will appear and offer you a number of the most common locations to find the files you are looking for. In the illustration below, we're looking in the top option `Desktop` for the files as we can see them on the desktop behind the *Open File* Dialog box of **gdebi**. Other likely locations are the `Downloads` directory or your home directory. The screenshot below shows this step:



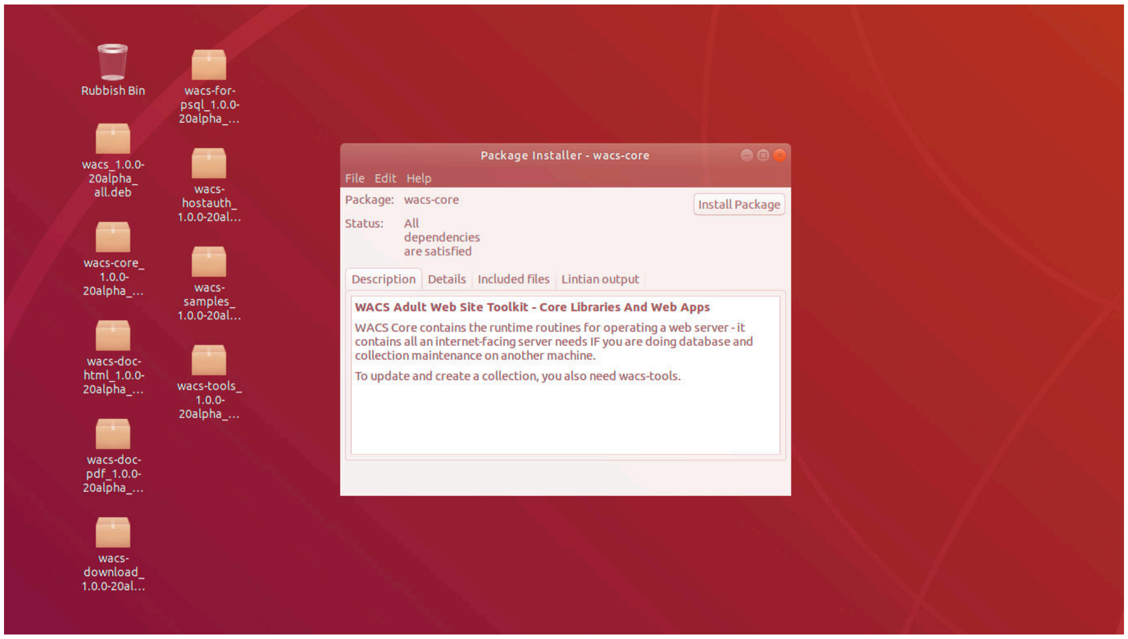
Name	Size	Modified
Desktop		Yesterday
Documents		Yesterday
Downloads		Yesterday
Music		Yesterday
Pictures		Yesterday
Public		Yesterday
Templates		Yesterday
Videos		Yesterday

The next step is to select the first of the packages for installation - this should always be **wacs-core** as this contains the basic bones of the WACS installation that are needed in all cases. The screenshot below shows this being selected in **gdebi**:



Name	Size	Modified
wacs_1.0.0-20alpha_all.deb	57.9 kB	Yesterday
wacs-core_1.0.0-20alpha_all.deb	461.8 kB	Yesterday
wacs-doc-html_1.0.0-20alpha_all.deb	18.5 MB	Yesterday
wacs-doc-pdf_1.0.0-20alpha_all.deb	19.9 MB	Yesterday
wacs-download_1.0.0-20alpha_all.deb	50.8 kB	Yesterday
wacs-for-psql_1.0.0-20alpha_all.deb	47.5 kB	Yesterday
wacs-hostauth_1.0.0-20alpha_amd64.deb	7.3 kB	Yesterday
wacs-samples_1.0.0-20alpha_all.deb	131.5 kB	Yesterday
wacs-tools_1.0.0-20alpha_all.deb	223.4 kB	Yesterday

Clicking on the **Open** button will cause a summary screen to appear, detailing the basic information about the package you are about to install. This is shown below - a key item to look out for in this screen is the bit where it says `Status:` - sometimes this will say as it does in this screen that `All dependencies are satisfied` but on some occasions it will say how many additional packages need to be installed. This can be a quite large, sometimes as high as twenty or thirty packages:


Rubbish Bin
wacs-for-psql_1.0.0-20alpha_...
wacs_1.0.0-20alpha_all.deb
wacs-hostauth_1.0.0-20al...
wacs-core_1.0.0-20alpha_...
wacs-samples_1.0.0-20al...
wacs-doc-html_1.0.0-20alpha_...
wacs-tools_1.0.0-20alpha_...
wacs-doc-pdf_1.0.0-20alpha_...
wacs-download_1.0.0-20al...
Package Installer - wacs-core
File Edit Help
Install Package
Description Details Included files Lintian output

Package: wacs-core

Status: All dependencies are satisfied

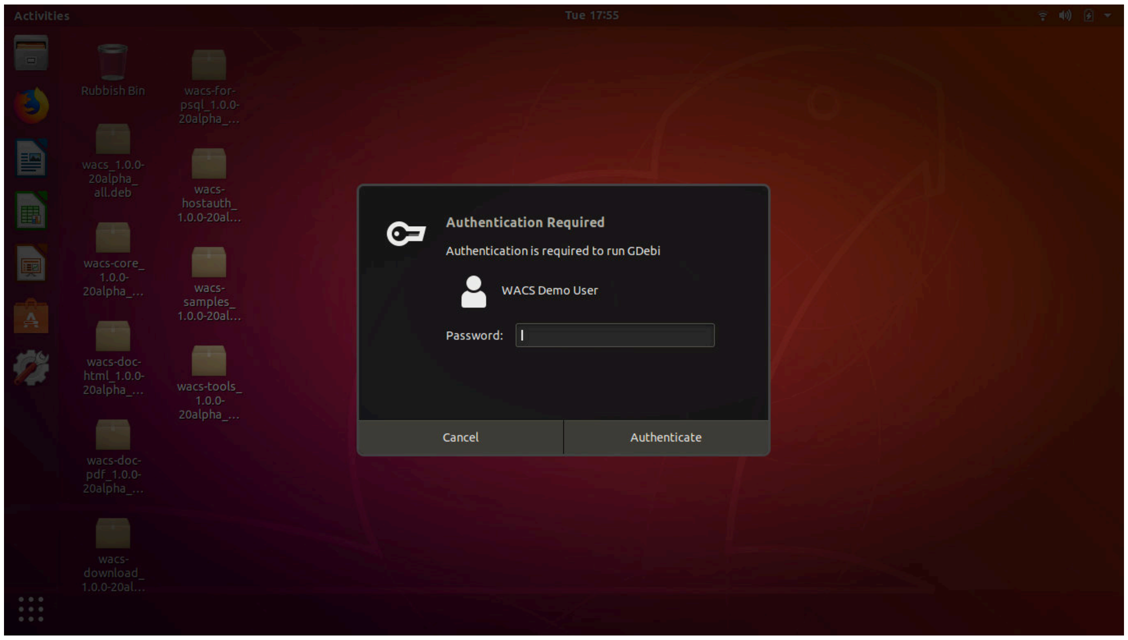
WACS Adult Web Site Toolkit - Core Libraries And Web Apps

WACS Core contains the runtime routines for operating a web server - it contains all an internet-facing server needs IF you are doing database and collection maintenance on another machine.

To update and create a collection, you also need wacs-tools.

While this 28 packages figure may sound high, it's what is actually needed in terms of software infrastructure (libraries and helpers) for all the Wacs packages and these packages will be requested by whichever Wacs package is installed first. Similarly, if you've already installed any of these prerequisite packages on behalf of another application, the number needing to be installed now will be greatly reduced. What we are installing here are things like the **apache 2** web server, the **php7** programming language, and a range of special libraries and tools that we use when accessing media files like photos and videos. These will also include all the various Perl and PHP modules we use for database access and the like.

We click on the **Install Package** button and an additional dialog box may appear requiring Authenication as illustrated below:


Activities
Tue 17:55
Rubbish Bin
wacs-for-psql_1.0.0-20alpha_...
wacs_1.0.0-20alpha_all.deb
wacs-hostauth_1.0.0-20al...
wacs-core_1.0.0-20alpha_...
wacs-samples_1.0.0-20al...
wacs-doc-html_1.0.0-20alpha_...
wacs-tools_1.0.0-20alpha_...
wacs-doc-pdf_1.0.0-20alpha_...
wacs-download_1.0.0-20al...
Cancel
Authenticate

Authentication Required

Authentication is required to run GDebi

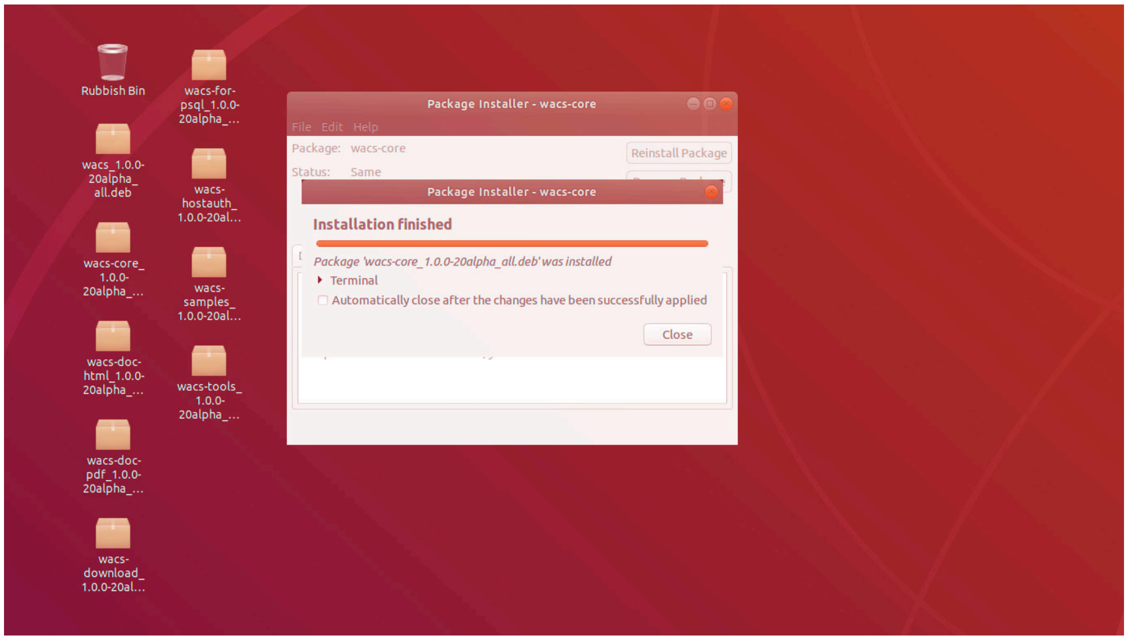
WACS Demo User

Password:

Initially a pop-up box appears saying `Installing dependencies....` This part of the installation procedure can actually take quite a while because on a normal Debian desktop install many of the requested packages will not be available locally. It will therefore make a connection to the internet software repositories and download the likes of **MySQL 5/PostgreSQL 10, apache 2, Php 7** and the perl modules required.

However under some circumstances you may notice that it doesn't show any progress through this process. Although this was mainly a problem with earlier releases, it can still happen. Sometimes the cause is issues with your internet connection or the configuration of your package manager, it can also be that an older package is requesting some user intervention. You can click on the down arrow by the progress bar to expand the Terminal window and see the actual problem. Once you have attended to this, the install process should continue normally.

Once the dependencies have all be installed, the **wacs-core** package itself will be installed last. Finally we should see the message `Installation finished` but of course this is merely for **wacs-core** - the installation of the Wacs environment as a whole is not yet complete:


Rubbish Bin
wacs-for-psql_1.0.0-20alpha_...
wacs_1.0.0-20alpha_all.deb
wacs-hostauth_1.0.0-20al...
wacs-core_1.0.0-20alpha_...
wacs-samples_1.0.0-20al...
wacs-doc-html_1.0.0-20alpha_...
wacs-tools_1.0.0-20alpha_...
wacs-doc-pdf_1.0.0-20alpha_...
wacs-download_1.0.0-20al...
Package Installer - wacs-core
File Edit Help
Package: wacs-core
Status: Same
Reinstall Package
Terminal
Close

Installation finished

Package 'wacs-core_1.0.0-20alpha_all.deb' was installed

Automatically close after the changes have been successfully applied

You now need to repeat the same steps for the other packages; we would suggest installing **wacs-hostauth** next unless you are only planning to use the internal database for user authentication. Even then it can be useful to have the **wacs-hostauth** package installed for initial setup and testing. Unless you are doing all the media ingest operations on a separate workstation with connections to the database and the storage, you will also need **wacs-tools**. You almost certainly also want the documentation set in one form or another so choose one of **wacs-doc-pdf** or **wacs-doc-html**; both is fine too - they don't actually take up that much space. We would also suggest that you do install the **wacs-samples** package as well as several of the samples (*`attrib.xml`* and *`keywords.xml`*) are pretty much essential to getting the initial system up and running. You can choose not to install our sample models and vendors at the setup stage if you don't want those.

When we get to the **wacs** package itself, we use either the MySQL 5 version (default) or the **wacs-for-psql** for PostgreSQL support. There is no difference in the actual version of WACS provided, just in the configuration files and actions it takes for the respective database types. If you intend to use a different database like Oracle 12, you can select either of the **wacs** packages - the code is the same.

The main wacs package comes last; the reason for this is that the wacs package is the “glue” that holds all the parts of Wacs together. In due course we'd hope you could merely install the wacs package itself and have all the pre-requisites automatically installed. Unfortunately the current package managers that do the dependency resolution all steadfastly refuse to include other manually downloaded packages in that dependency resolution process, and so it fails. As soon as sourceforge.net starts offering the ability to create package repositories, or we can get Wacs accepted into the repositories of the major distributions, these problems should all disappear.

Once they're all done, you should be ready to proceed to the next step. We have worked hard to get the packages to do as much of the initial setup as we can and you should at this stage be able to simply move on to running the **wacssetup** which we will cover in the next chapter Chapter 6, *Creating The Initial WACS Databases*.

RPM/DEB Initial Setup

We have put a lot of effort into making the package installs *Just Work*, so we hope that you can skip this section and move straight on to the web-based install and setup process described in the next chapter, Chapter 6, *Creating The Initial WACS Databases*. However if you have an unusual installation environment, the following section details the various things you need to do to get the installer working. You may need this if you are in an environment using external authentication (like LDAP or Active Directory), or other oddities that cause the package scripts not to work.

In addition to appearing there, the manual initial database creation process documentation can be found in the text file `README.database` in the WACS installation tree. This installation tree is usually `/usr/share/wacs`.

However, before we proceed to the initialisation step, it is important to ensure that your own user account has the requisite access rights to make use of the WACS software installation itself. You do not need to do this if you plan on accessing the software solely via the web interfaces. As a user that's probably OK, as an administrator maybe slightly less so as there are some problem resolution and similar tasks that still need to be carried out via a command prompt.

Permissions Issues

The normal action of the RPM/DEB packages is to create a user account to hold all the datafiles, typically called simply **wacs**. Unless you choose to do otherwise, the images and video clips loaded into the WACS system are normally stored in the home directory of this account. For obvious reasons, the security on all the Wacs directories are locked down pretty hard, so you will need to pay attention to it. A new group also called `wacs` is created and initially the wacs and web server owner account are added to this. If you wish to read the documentation, samples and configuration files without always having to become the superuser first, you can simply add your own username to the wacs group.

There are a number of ways to do this including using the **System->Administration->Users and Groups** option or you can do it in the shell as root with the usermod command. Since the interface on Users and Groups GUI is very different between different distributions and versions, we're going to stick with the command line method as that is portable. To do this you use the usermod command if you're using a superuser shell, substituting the *`your_username`* with your user name:

```
# usermod -a -G wacs your_username
#
```

If you're using Ubuntu you can also do it directly using the **sudo** command:

```
# sudo usermod -a -G wacs your_username
```

```
[sudo] password for your_name:
#
```

Warning

After you've added yourself to the wacs group, the change will almost certainly not take place within the current session. You will have to log out and log back in again for your membership to be recognised. The **groups** command lists the groups you are currently in; when this list includes wacs, things should be working - when it does not, they won't be!

Chapter 6. Creating The Initial WACS Databases

Overview: WACS Database Creation

Once you've installed the WACS packages on your computer, the next step is to actually get WACS itself up and running. Since WACS is expected to have a huge number of images and video clips to look after and a whole lot more information about how they relate to one another and what they contain, it uses a database to store all the information it needs. WACS can in fact use pretty much any relational database package for that although we currently support four options: MySQL 5.x, MariaDB 10.x, PostgreSQL 10/12 and Oracle 10/11/12. However before WACS can use that relational database, it needs to have it's own database account, workspace area, data structures and configuration data created. This can be fairly complex task unless you're an experienced DBA (DataBase Administrator) and this chapter will guide you through this process.

When it comes to creating the initial databases needed by WACS in order to function, you have up to three possible options:

- Use the web application **wacssetup** (see the section called “Creating Initial Databases With **wacssetup**”)
- Do the necessary steps manually (see the section called “Manual Database Creation Steps”)
- Do the entire installation and configuration using **easyinstall** (No longer recommended, see Chapter 14, *Installing With EasyInstall*)

Tip

If you have installed WACS using either the Ubuntu `.deb` or Fedora `.rpm` packages, you'll almost certainly find that using the new **wacssetup** web application will be the easiest option. Make sure that you did choose the correct version of the main wacs package for your choice of database - either the standard `wacs` for MySQL, MariaDB or Oracle; or the `wacs-for-psql` for PostgreSQL. There is no difference between what the packages contain, only what they tell the package manager that they also need.

Note

If you are doing the installation with **easyinstall** the necessary steps are taken as part of that process and you don't need to replicate them. This is no longer a recommended way to install WACS.

Creating Initial Databases With wacssetup

Warning

wacssetup is now our standard method of installation and is pretty thoroughly tested before each new release of WACS ships. Please do report any problems you find with it. The **wacssetup** command will tell you which step it is doing and if it fails, you should be able to continue from that step in the manual creation proceedures (see the section called “Manual Database Creation Steps”.

If you've just installed WACS using the packaged version, then you will need to open up a web browser and point it at either:

```
http://localhost/wacs-cgi/wacssetup
```

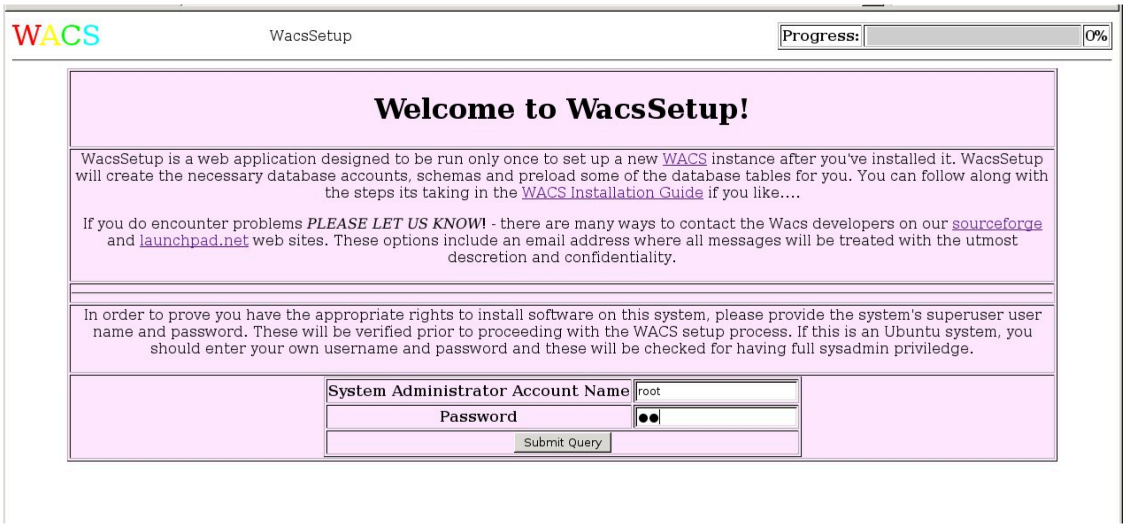
... or ...

```
http://yourhost.example.com/wacs-cgi/wacssetup
```

You can also run the installer remotely if you either don't have a graphics console on the server, or it's at a remote location. So long as you have administrator rights and the server has been correctly configured, it should all work just fine. In this case you just substitute `localhost` for the fully qualified internet domain name of the server.

Alternatively you can just bring up the normal Wacs introduction web page, all of the wacs applications will fail with an error message if you try to use them. Either on this introduction page, or on the connection failed error message screen, you'll find a link to *New Installation? click here to perform initial setup (Administrators Only)*. Click on this link and you'll be sent to the **wacssetup**.

wacssetup itself


WACS WacsSetup Progress: 0%

Welcome to WacsSetup!

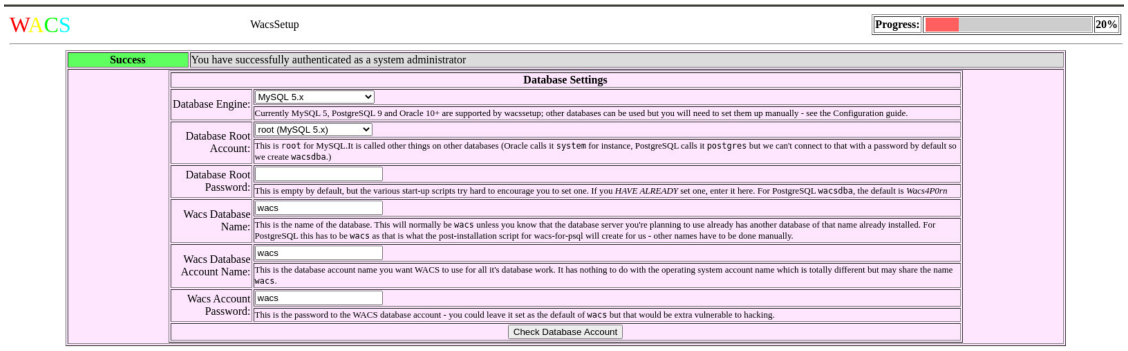
WacsSetup is a web application designed to be run only once to set up a new WACS instance after you've installed it. WacsSetup will create the necessary database accounts, schemas and preload some of the database tables for you. You can follow along with the steps its taking in the WACS Installation Guide if you like....

If you do encounter problems *PLEASE LET US KNOW!* - there are many ways to contact the Wacs developers on our sourceforge and launchpad.net web sites. These options include an email address where all messages will be treated with the utmost descretion and confidentiality.

In order to prove you have the appropriate rights to install software on this system, please provide the system's superuser user name and password. These will be verified prior to proceeding with the WACS setup process. If this is an Ubuntu system, you should enter your own username and password and these will be checked for having full sysadmin priviledge.

System Administrator Account Name	root
Password	●●

The first step of the process is where **wacssetup** makes sure you actually have the rights to do the things you're asking it to do. So it asks you for the root password to the system. If you're on an Ubuntu box and haven't set a root password, you can use your own account name and password providing you have sudo privilege to manage the system.



WACS WacsSetup Progress: 20%

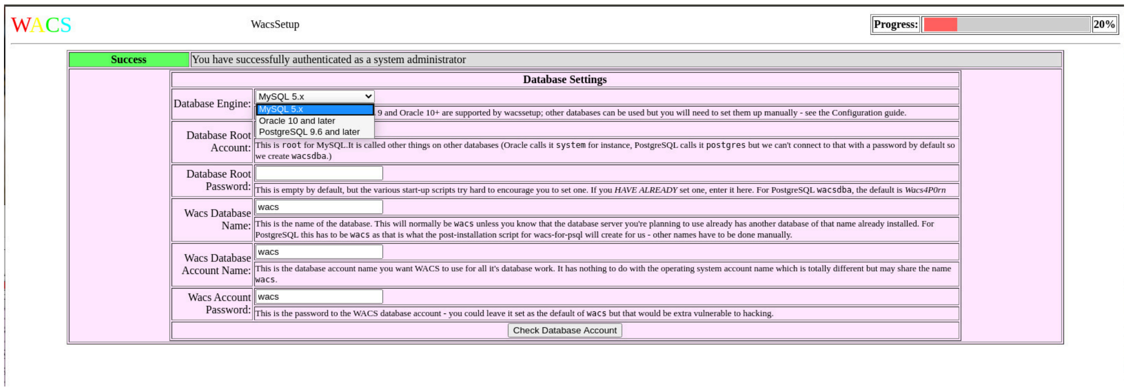
Success	You have successfully authenticated as a system administrator

Database Settings

Database Engine:	MySQL 5.x
	Currently MySQL 5, PostgreSQL 9 and Oracle 10+ are supported by wacssetup; other databases can be used but you will need to set them up manually - see the Configuration guide.
Database Root Account:	root (MySQL 5.x)
	This is `root` for MySQL.It is called other things on other databases (Oracle calls it `system` for instance, PostgreSQL calls it `postgres` but we can't connect to that with a password by default so we create `wacsdba`.)
Database Root Password:	
	This is empty by default, but the various start-up scripts try hard to encourage you to set one. If you *HAVE ALREADY* set one, enter it here. For PostgreSQL `wacsdba`, the default is *Wacs4P0rn*
Wacs Database Name:	wacs
	This is the name of the database. This will normally be `wacs` unless you know that the database server you're planning to use already has another database of that name already installed. For PostgreSQL this has to be `wacs` as that is what the post-installation script for wacs-for-psql will create for us - other names have to be done manually.
Wacs Database Account Name:	wacs
	This is the database account name you want WACS to use for all it's database work. It has nothing to do with the operating system account name which is totally different but may share the name `wacs`.
Wacs Account Password:	wacs
	This is the password to the WACS database account - you could leave it set as the default of `wacs` but that would be extra vulnerable to hacking.

Check Database Account

The second screen asks you to confirm the details of the database wacs is going to use. In almost all cases, apart from giving the database root password (if set), you should probably just accept the defaults. The choices are here primarily to highlight to you what is actually going to be used. Here you can choose which Database you plan to use: MySQL/MariaDB, Oracle 10+ or PostgreSQL 9.6+ as shown below:



WACS WacsSetup Progress: 20%

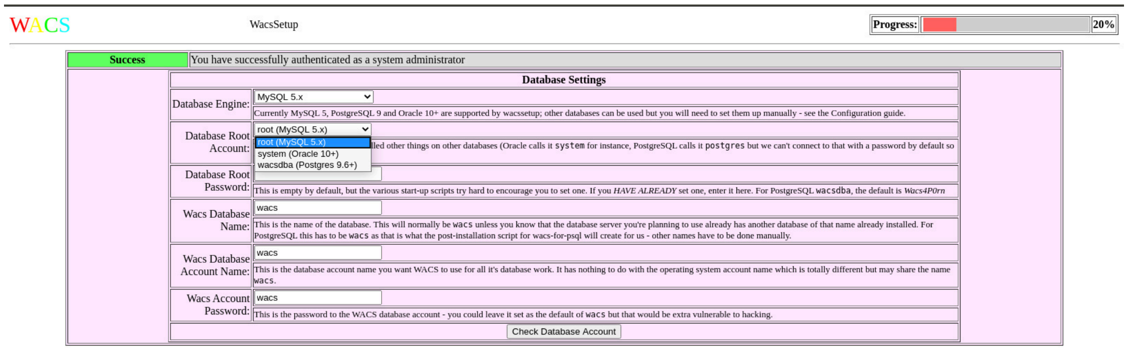
Success	You have successfully authenticated as a system administrator

Database Settings

Database Engine:	MySQL 5.x MySQL 5.x Oracle 10 and later PostgreSQL 9.6 and later
	9 and Oracle 10+ are supported by wacssetup; other databases can be used but you will need to set them up manually - see the Configuration guide.
Database Root Account:	
	This is `root` for MySQL.It is called other things on other databases (Oracle calls it `system` for instance, PostgreSQL calls it `postgres` but we can't connect to that with a password by default so we create `wacsdba`.)
Database Root Password:	
	This is empty by default, but the various start-up scripts try hard to encourage you to set one. If you *HAVE ALREADY* set one, enter it here. For PostgreSQL `wacsdba`, the default is *Wacs4P0rn*
Wacs Database Name:	wacs
	This is the name of the database. This will normally be `wacs` unless you know that the database server you're planning to use already has another database of that name already installed. For PostgreSQL this has to be `wacs` as that is what the post-installation script for wacs-for-psql will create for us - other names have to be done manually.
Wacs Database Account Name:	wacs
	This is the database account name you want WACS to use for all it's database work. It has nothing to do with the operating system account name which is totally different but may share the name `wacs`.
Wacs Account Password:	wacs
	This is the password to the WACS database account - you could leave it set as the default of `wacs` but that would be extra vulnerable to hacking.

Check Database Account

Once you have selected the Database type, you need to select the name of the database system administrator to use - we don't *automatically* select the one normally for the specific database as it is very possible to use the other names on different databases. If you are running on a large Oracle system for instance, your DBA might choose to create you an admin account for just your database activities - `wacsdba` as used in PostgreSQL would be a good name for them to use. You might even want to suggest they use that!



WACS WacsSetup Progress: 20%

Success	You have successfully authenticated as a system administrator

Database Settings

Database Engine:	MySQL 5.x
	Currently MySQL 5, PostgreSQL 9 and Oracle 10+ are supported by wacssetup; other databases can be used but you will need to set them up manually - see the Configuration guide.
Database Root Account:	root (MySQL 5.x) root (MySQL 5.x) system (Oracle 10+) wacsdba (Postgres 9.6+)
	lled other things on other databases (Oracle calls it `system` for instance, PostgreSQL calls it `postgres` but we can't connect to that with a password by default so
Database Root Password:	
	This is empty by default, but the various start-up scripts try hard to encourage you to set one. If you *HAVE ALREADY* set one, enter it here. For PostgreSQL `wacsdba`, the default is *Wacs4P0rn*
Wacs Database Name:	wacs
	This is the name of the database. This will normally be `wacs` unless you know that the database server you're planning to use already has another database of that name already installed. For PostgreSQL this has to be `wacs` as that is what the post-installation script for wacs-for-psql will create for us - other names have to be done manually.
Wacs Database Account Name:	wacs
	This is the database account name you want WACS to use for all it's database work. It has nothing to do with the operating system account name which is totally different but may share the name `wacs`.
Wacs Account Password:	wacs
	This is the password to the WACS database account - you could leave it set as the default of `wacs` but that would be extra vulnerable to hacking.

Check Database Account

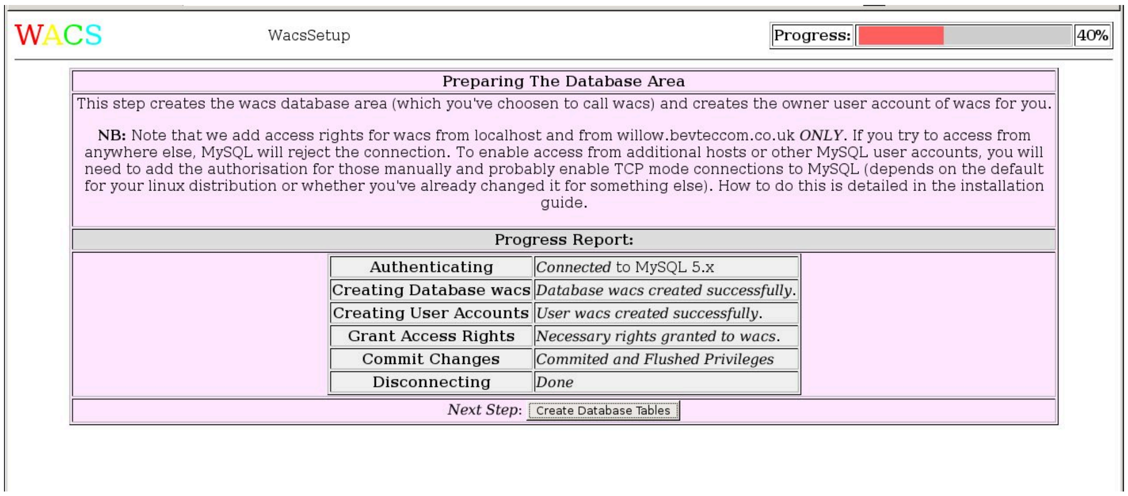
Important

The **wacs-for-psql** version works a little differently from other versions - the postgres DBA is not by default accessible to webapps and to this end we create an additional

DBA account called `wacsdba` as part of the installation process. By default this has the password `Wacs4P0rn` which you should change as soon as possible after the installation has completed. See the section later in this chapter about this.

Note

Merely changing the password here from the default will not prove completely successful (yet), as the configuration file `wacs.cfg` has to be updated to match it. If you do change it here, you should log in on a terminal window *before progressing to the next screen* and change the entries in the `database` section of the `wacs.cfg` config file as well. Choosing a new password in **wacssetup** will ensure the database is setup with the new password from the start, but the last step of **wacssetup** will fail if the password **wacsetup** is using differs from that given in `wacs.cfg` (which is usually to be found in `/etc/wacs.d/wacs.cfg`).



Preparing The Database Area

This step creates the wacs database area (which you've choosen to call wacs) and creates the owner user account of wacs for you.

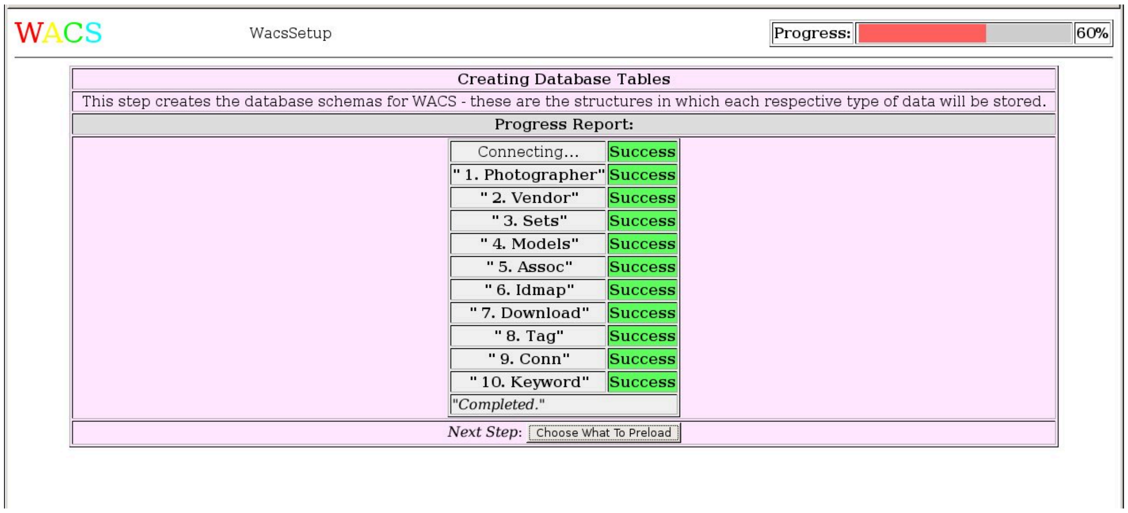
NB: Note that we add access rights for wacs from localhost and from willow.bevteccom.co.uk *ONLY*. If you try to access from anywhere else, MySQL will reject the connection. To enable access from additional hosts or other MySQL user accounts, you will need to add the authorisation for those manually and probably enable TCP mode connections to MySQL (depends on the default for your linux distribution or whether you've already changed it for something else). How to do this is detailed in the installation guide.

Progress Report:

Authenticating	*Connected* to MySQL 5.x
Creating Database wacs	*Database wacs created successfully.*
Creating User Accounts	*User wacs created successfully.*
Grant Access Rights	*Necessary rights granted to wacs.*
Commit Changes	*Commited and Flushed Privileges*
Disconnecting	*Done*

Next Step: Create Database Tables

This step creates the basic database structures and user account. There's not really much to say about this apart from a mention that if this *fails* for any reason, take a look at the section called “Installation Troubleshooting” and Chapter 12, *Troubleshooting* for help and guidence on what to do next.



Creating Database Tables

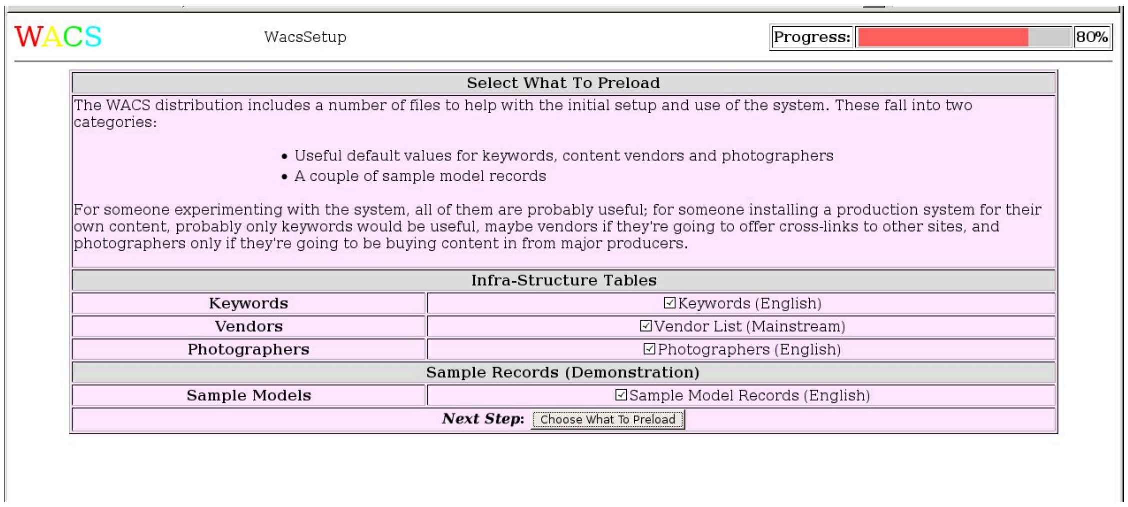
This step creates the database schemas for WACS - these are the structures in which each respective type of data will be stored.

Progress Report:

Connecting...	Success
" 1. Photographer"	Success
" 2. Vendor"	Success
" 3. Sets"	Success
" 4. Models"	Success
" 5. Assoc"	Success
" 6. Idmap"	Success
" 7. Download"	Success
" 8. Tag"	Success
" 9. Conn"	Success
" 10. Keyword"	Success
"Completed."	

Next Step: Choose What To Preload

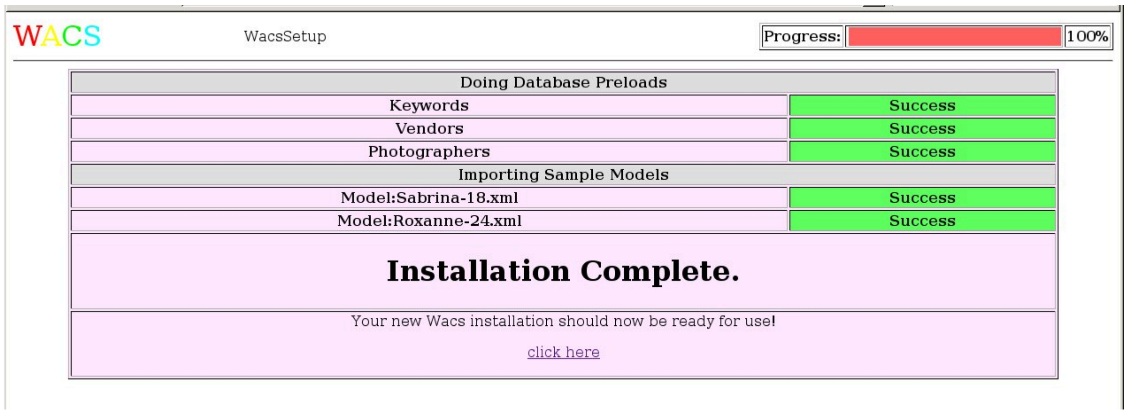
This step creates the WACS specific database structures, known as *schemas* for each of the things it mentions. At this level, even if you're not planning on using facilities like vendors and photographers, at least the data structures need to be present, even if there's no data in them.



Select What To Preload	
The WACS distribution includes a number of files to help with the initial setup and use of the system. These fall into two categories: • Useful default values for keywords, content vendors and photographers • A couple of sample model records For someone experimenting with the system, all of them are probably useful; for someone installing a production system for their own content, probably only keywords would be useful, maybe vendors if they're going to offer cross-links to other sites, and photographers only if they're going to be buying content in from major producers.	
Infra-Structure Tables	
Keywords	☑Keywords (English)
Vendors	☑Vendor List (Mainstream)
Photographers	☑Photographers (English)
Sample Records (Demonstration)	
Sample Models	☑Sample Model Records (English)
Next Step: Choose What To Preload	

At this step you have to decide what sample data you want pre-loaded into the Wacs system. Our advice would be that unless you have an alternative set of keywords that you've developed yourself, you're almost certain to want to preload the Keywords schema under almost all circumstances. The vendors list is definitely useful for private collectors and is probably also of some use to website operators who might wish to earn additional revenue through cross-referal commissions. It also offers some examples of how to configure the download system. The photographers database is probably of the least use unless you're either a collector using sources that provide that information, or a website owner planning to offer that search feature at some point. That said it's small and relatively harmless, so including it isn't a big problem.

The sample model records are a rather different issue as these really are *NOT* suitable for inclusion on any kind of publicly accessible system. The sets and videos mentioned may be licensable for commercial use - contact Wacs developer Beaky for more information. These are however very useful introductions to how model records work and will significantly aid you in getting used to using and managing the Wacs system. We hope to shortly be able to provide some of the sets mentioned for download so you can set up a server with an initial data set.



Doing Database Preloads	
Keywords	Success
Vendors	Success
Photographers	Success
Importing Sample Models	
Model:Sabrina-18.xml	Success
Model:Roxanne-24.xml	Success

Installation Complete.

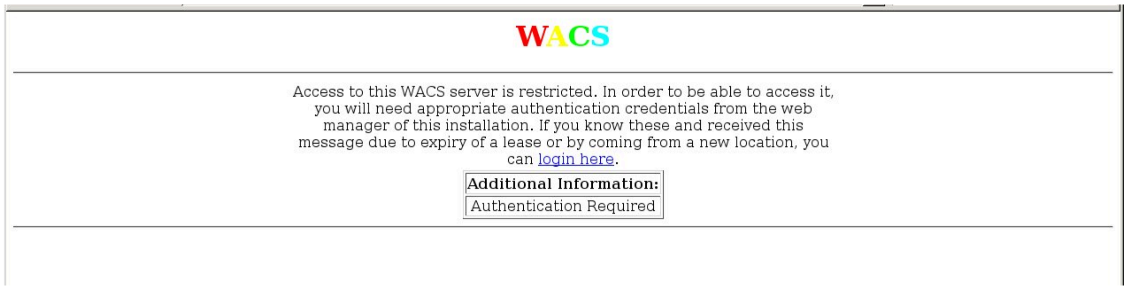
Your new Wacs installation should now be ready for use!

click here

... and that basicially is it. Wacs is installed and the initial system up and running. Although we've covered it else where (most notably in the User Guide, the next section will cover what happens when you click on the *click here* link that first time.

After wacssetup

So... you've installed the packages, you've setup the database and your Wacs server is ready for use. But what exactly should you expect?... this section just guides you through what that first connection to your newly install system should look like.

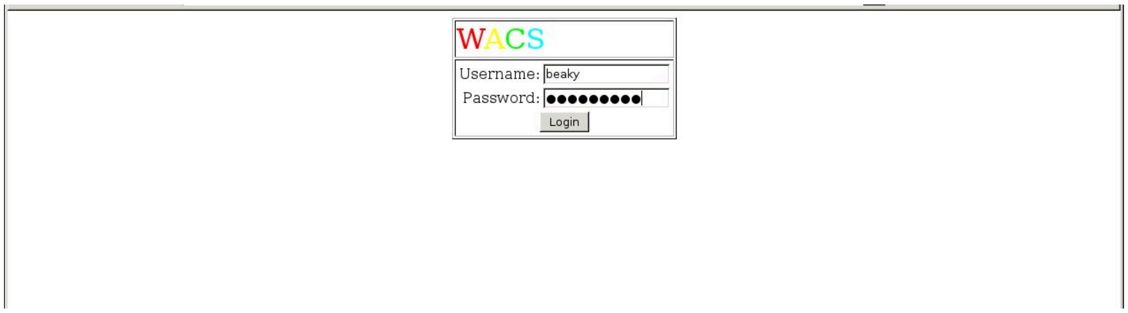


WACS

Access to this WACS server is restricted. In order to be able to access it, you will need appropriate authentication credentials from the web manager of this installation. If you know these and received this message due to expiry of a lease or by coming from a new location, you can login here.

Additional Information:
Authentication Required

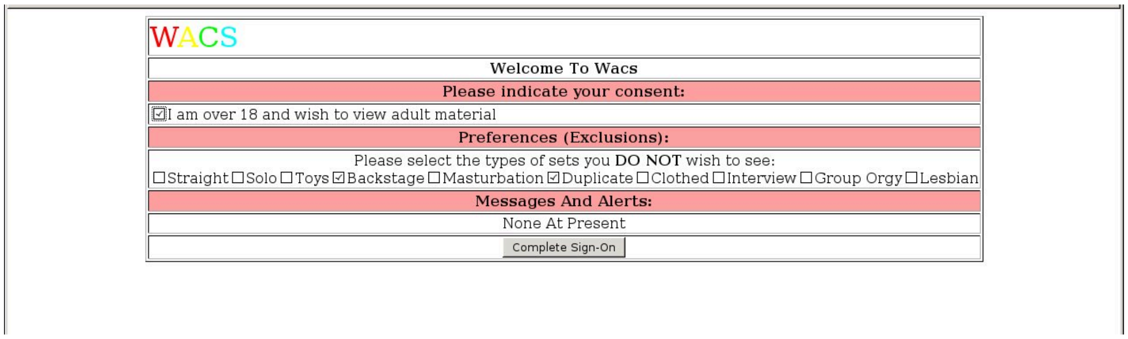
At this point you're probably thinking *Hey, I thought you said the server was ready? Why's it asking me more questions?*. Well, remember Wacs is a sophisticated system that tries very hard to be as secure as possible because of the nature of the material it's designed to hold. Although you may not be familiar with it yet, this is actually the start of just about each and every new session with Wacs. This message is intentionally cryptic so that it does not immediately indicate what the system is. You just need to click on the `login here` link to proceed.



WACS

Username: beaky
Password: ●●●●●●●●●

Here it's asking for your username and password - normally these are the same ones as you'd use when connecting to the server itself - nothing special.



WACS

Welcome To Wacs

Please indicate your consent:

☑ I am over 18 and wish to view adult material

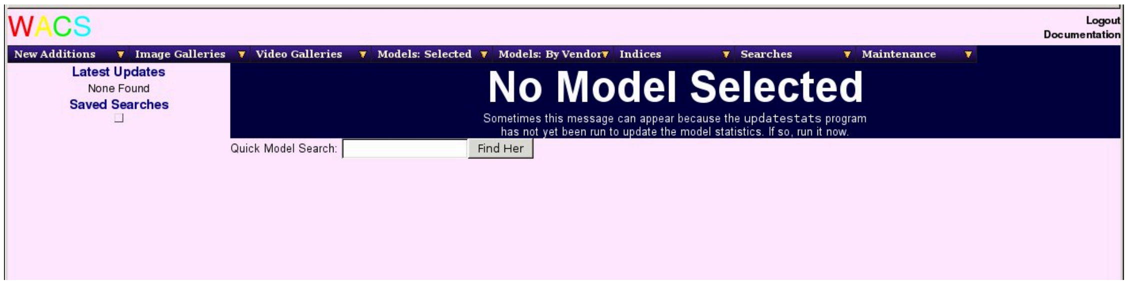
Preferences (Exclusions):

Please select the types of sets you DO NOT wish to see:
☐ Straight ☐ Solo ☐ Toys ☑ Backstage ☐ Masturbation ☑ Duplicate ☐ Clothed ☐ Interview ☐ Group Orgy ☐ Lesbian

Messages And Alerts:

None At Present

At this screen all you really need to worry about is ticking the box consenting to viewing adult material (although there probably isn't any actually there yet). Then just click *Complete Sign-On* and you're there!



No Model Selected

Sometimes this message can appear because the updatestats program has not yet been run to update the model statistics. If so, run it now.

Yes, this is actually a perfectly working Wacs system! It just looks very spartan without any actual data to present. If you look through the model indexes, you should find empty pages for Kaz B, Roxanne and Sabrina if you installed the sample model records.

If you run into problems during the installation, there's a whole chapter (Chapter 12, *Troubleshooting*) which takes you through many of the common pitfalls and problems. This covers both installation issues (the section called “Installation Troubleshooting”) and general issues (the section called “General Troubleshooting Tips”).

The next step is to add some data but that is such a big topic that there's a whole separate guide about that! Go take a look at the Administration Guide....

Manual Database Creation Steps

1. Create the WACS database account (MySQL)

Connect to the database as the root user, giving the password as appropriate; if you've not set one the default is blank so just press return when prompted. You then create the database and the user account (once for each place you might be coming from), give access to that user account to the database, flush the contents and then quit. Here's a sample conversation - you obviously need to replace the 'myserver.example.com' with whatever your real fully qualified domain name is. You might also wish to choose a more secure password, but do remember you need to change it in /etc/wacs.d/wacs.cfg (dbpass and phpdbconnect variables) as well or it just won't work.

Here goes:

```
# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 80
Server version: 5.0.45 Source distribution

Type 'help;' or \h for help. Type '\c' to clear the buffer.

mysql> CREATE DATABASE wacs;
Query OK, 1 row affected (0.03 sec)

mysql> CREATE USER 'wacs'@'myserver.example.com'
    -> IDENTIFIED BY 'wacs';
Query OK, 0 rows affected (0.08 sec)

mysql> CREATE USER 'wacs'@'myserver'
    -> IDENTIFIED BY 'wacs';
```

```
Query OK, 0 rows affected (0.00 sec)

mysql> CREATE USER 'wacs'@'localhost'
    -> IDENTIFIED BY 'wacs';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL ON wacs.* TO wacs;
Query OK, 0 rows affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)

mysql> QUIT;
Bye
#
```

2. Create the necessary database schemas

The next step is to log in as the wacs user account you just created and run the SQL scripts that create the various database tables. There are scripts provided for both MySQL 5 and Oracle 10, but this example is based upon using the MySQL 5 version. These should be found in `/usr/share/wacs/creation/MySQL5`.

```
# cd /usr/share/wacs/creation/MySQL5
# mysql -u wacs -p wacs
Enter password:
Welcome to the MySQL monitor.  Commands end withh ; or \g.
Your MySQL connection id is 82
Server version: 5.0.45 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> source create_mysql.sql
WACS Database Table Creation Script for MySQL
Commencing Table Creation:
  1. Photographer
Query OK, 0 rows affected (0.23 sec)

  2. Vendor
Query OK, 0 rows affected (0.01 sec)

  3. Sets
Query OK, 0 rows affected (0.01 sec)

  4. Models
Query OK, 0 rows affected (0.02 sec)

  5. Assoc
Query OK, 0 rows affected (0.01 sec)
```

```
  6. Idmap
Query OK, 0 rows affected (0.01 sec)

  7. Download
Query OK, 0 rows affected (0.00 sec)

  8. Tag
Query OK, 0 rows affected (0.01 sec)

  9. Conn
Query OK, 0 rows affected (0.02 sec)

 10. Keyword
Query OK, 0 rows affected (0.01 sec)

 11. Wacsuser
Query OK, 0 rows affected (0.01 sec)

 12. Attrib
Query OK, 0 rows affected (0.02 sec)

 13. Notes
Query OK, 0 rows affected (0.01 sec)

Tables Created - Committing Changes
Query OK, 0 rows affected (0.00 sec)

Completed.
mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
Bye
#
```

3. Create default database contents (optional)

Note

this step is *RECOMMENDED* unless you know precisely what you are doing. Some commercial sites may not wish to preload this data, but should substitute their own alternatives if they want certain features to work.

Warning

If you changed the password in step 1. above from the default value, you *NEED* to have made the matching change to the configuration file before doing this step.

Tip

The **wacspop** replaced the previous seperate commands in Wacs release 0.8.6. If you're using an earlier release these instructions will not work. Check the earlier version of this document instead!

There are four database tables that contain standard values, plus whatever you add to them over time; these are called keywords, photographers, attrib and vendors. In this step you will be loading some initial values into these database tables. To do this you need to go to /usr/share/wacs/samples/database and run the **wacspop** populate command in the utils directory on each of the XML data files there in turn. There is also the users.xml file which you should propably also load - you will need it if using the database as the authentication source. This is typically necessary for WebHosted sites and extremely useful on any pay sites.

```
# cd /usr/share/wacs/samples/database
# /usr/share/wacs/utils/wacspop keywords.xml
Inserting Entry For Keyword: seethru
Inserting Entry For Keyword: nopanties
Inserting Entry For Keyword: teen
[...]
Inserting Entry For Keyword: cyc
Inserting Entry For Keyword: country
Inserting Entry For Keyword: alley
# /usr/share/wacs/utils/wacspop photographers.xml
Inserting Entries For Photographer: DFR
Inserting Entries For Photographer: SWE
Inserting Entries For Photographer: MAX
[...]
Inserting Entries For Photographer: JAN
Inserting Entries For Photographer: TOB
Inserting Entries For Photographer: HBM
# /usr/share/wacs/utils/wacspop vendors.xml
Inserting Entries For Site: SE
Inserting Entries For Site: WACSD
Inserting Entries For Site: FJ
[...]
Inserting Entries For Site: AMK
Inserting Entries For Site: KPC
Inserting Entries For Site: KHA
# /usr/share/wacs/utils/wacspop attrib.xml
Inserting Entries For Attribute: shaven
Inserting Entries For Attribute: tinytit
Inserting Entries For Attribute: nopanties
[...]
#
```

4. Import sample model records (optional)

Note

This is a very optional step but will help you if you're setting up a WACS web site for the first time.

This step loads three sample model records into the database to provide an example of how a typical model record might look. There are three such files provided in `/usr/share/wacs/samples/models` containing details of three models: Kaz B, Sabrina and Roxanne.

```
# cd /usr/share/wacs/samples/models
# wacsimport Sabrina-18.xml
Keyless ID map for JAFN
# wacsimport Roxanne-24.xml
# wacsimport KazB-30.xml
```

5. Import sample set records (optional)

Note

For some sample sets for your new WACS web site, we invite you to visit our demonstration site at PinkMetallic.com [http://www.pinkmetallic.com/] - *[CAUTION - contains adult material]*. Access to this site is currently free but there may be a small charge at some point in the future if referal revenues don't cover costs.

You will first need to download the sets that appeal to you, so if you select set number 14 for instance, you will need the `set14.zip` file and the `set14.xml` file. These can be found via the link titled *WACS Resources* from the main menu on PinkMetallic.com [http://www.pinkmetallic.com/]. Once you have these downloaded, place them both in the same directory and run the **wacsxmlin** program to load the data from the XML file. The **wacsxmlin** program requires the name of the .xml file it is to read as an argument, eg `wacsxmlin set4.xml`. The zip file will be automatically unpacked and it's contents placed in the images area. In this example, we're going to use the default layout, which if you haven't edited the configuration file will be gallery style (please see the discussion on Site Layout in the administration guide for more details).

```
# cd ~/Download
# wacsxmlin --default set14.xml
Unpacking archive:
  Roxanne07001.jpg
  Roxanne07002.jpg
  Roxanne07003.jpg
[...]
#
```

Chapter 7. Installation On A Web Hosting Site

Caution

This is a complex task and some level of familiarity with the Unix/Linux command line will probably be needed to be successful. This chapter is intended to guide you through the process of installing WACS onto a web site hosting provider's systems, so you may wish to take some time to familiarise yourself with the tools available at their control panel before starting on this process. You will need shell access to your web site server and this is an added cost option with most web hosting providers - the SSH protocol used to access the shell access services is widely available in Linux as openssh-client and we've successfully used the **PuTTY** free software application on Windows.

Please make sure that all of the packages/services described in the prequisites chapter (Chapter 3, *Prerequisites*) have been installed and are running correctly. If they don't appear to be there, do take the time to check the control panel for options to install additional software onto your web hosting environment. In most cases MySQL doesn't appear to be enabled by default on most web hosting companies.

WebHost Installation: Steps

Preparation On Web Host

1. The first step is to create a new folder under the web document tree (conventionally `public_html`) called `wacs`. Into this directory you need to copy all the components in the *`unpack_location/`*`htmlbones` directory and it's sub-directories, retaining the directory structure as found below `htmlbones`. There are a number of ways in which to achieve this and they vary enormously depending on your hosting provider, level of access at your hosting provider and equipment available to you locally.

   If you are unsure of how to proceed, one of the easiest ways would be to create a zip file of these files (eg the `htmlbones` folder) on the PC where you unpacked the WACS software and upload that file. Hopefully your hosting provider provides the **unzip** command within your shell environment. If not, that's pretty much fair game for a support call, it's a pretty normal thing to need at a shell prompt.

   Other techniques would include using a tar archive, the **rsync** command or placing all your configuration and enviroment files into a version control system like **git** or **Subversion** and using a text based client on the hosting provider to download them from your repository.

2. The second step is to check with the Perl Modules option in the control panel and make sure that DBI, DBD::MySQL and XML::Simple perl modules are available. If not, you may need to do one of three things: go through the process of importing them from CPAN, raise a support call with your hosting provider asking for them to be added or unpack them manually from the packages for the Linux distribution the web host provider is using (see notes in the section called “Final Notes For Web Hosting”).

The Wacs Code Itself

Due to limitations on what you can install, the ability to authenticate user accounts using the operating system tools cannot be implemented when using a Web Hosting Site. You will therefore have to use one of the other authentication techniques - storing user accounts in the database itself, using permanent access lists or enable all access at the WACS level and then use Apache's .htaccess files to demand a password before accessing those commands.

Note

Wacs includes support for authenticating using user account information stored in the main WACS database. Either this or wide open access are the normal ways of operating in a Web Hosting environment. The user account database includes lots of fields useful in managing subscriptions via this method. Bevtec Communications Limited [http://www.bevteccom.co.uk/] are working on a customer management extension using these fields, please contact them for more details. Alternatively you can develop your own utilising the hooks we have put there for this purpose. You can visit our demostration site, PinkMetallic.com [http://www.pinkmetallic.com/] to see the open access system in operation.

3. The first step is to copy the Wacs Perl Modules into an appropriate place - our web hosting provider had already created a perl subdirectory of our account's home directory, so we used that. That probably makes sense even if it hasn't already been done for you as it wasn't on one of the other major web hosting providers we did an install for a consultancy customer on.

```
# cd unpack_location
# cp modules/wacs.pm ~/perl/Wacs.pm
# mkdir ~/perl/Wacs
# cp modules/wacsui.pm ~/perl/Wacs/WacsUI.pm
# cp modules/wacsstd.pm ~/perl/Wacs/WacsStd.pm
# cp modules/wacsid.pm ~/perl/Wacs/WacsId.pm
#
```

Tip

Note the change of case of the names; most command line ftp/sftp tools will allow you to specify a second name on a put command for the name of the file at the destination. Thus you can do: `put wacs.pm Wacs.pm` to do the name change as part of the transfer itself.

Additionally if you want to include support for the PHP dialect of the Wacs API, you will also need to copy some additional files. Before that however you will have to determine which version of PHP your hosting provider is using. The transition from Php 5.x to Php 7.x is quite recent - simply typing: `php -v` at the command line should give you the answer. These instructions are now targeted for the default Php which is version 7 on our hosting provider - if yours is still using Php version 5, substitute `php5` for `php` in these instructions. You will need to do the following:

```
# cd wacs-php_unpack_location
# mkdir ~/php
# cp modules/wacs.php ~/php/wacs.php
# cp modules/wacsui.php ~/php/wacsui.php
# cp modules/XMLSimple.php ~/php/XMLSimple.php
```

```
#
```

You can skip all of the bits about security and the pam modules as we'll be unable to use those aspects of the Wacs system on a web hosting service. We will instead either use a global allow on the `wacs.acl` file to allow everyone access or we will be using the database authentication method.

4. The next step is setting up the cgi-bin directory - the first thing to do is to find out whether the hosting provider allows you to use one and if they have where the hosting provider has put it within your account's space. One major hosting company did not provide for a **cgi-bin** directory, instead requiring `.pl` extensions for each perl program. We built extensive support for doing this into the WACS 0.9.1 release and it now works reasonably effectively.

   In the case of the provider we're using for PinkMetallic.com [http://www.pinkmetallic.com/] this is a sub-directory called `cgi-bin` under the `public_html` directory.

   While the packaged versions of WACS have moved to creating their own `wacs-cgi` directory at the top level of the web server tree, we can only do that when we have control over the Apache (or other webserver) configuration. We can't assume that we will have this on a hosting provider, so we use a dedicated `wacs` sub-directory of the `cgi-bin` directory instead. We recommend putting the wacs scripts into a sub-directory of the `cgi-bin` directory when using a web hosting service to make it easier to manage upgrades and the like. Here's what to do with `cgi-bin` hosting providers:

```
# cd unpack_location
# cp index/wacs* ~/public_html/cgi-bin/wacs/
# cp models/wacs* ~/public_html/cgi-bin/wacs/
# cp presentation/wacs* ~/public_html/cgi-bin/wacs/
# cp retrieval/wacs* ~/public_html/cgi-bin/wacs/
# cp search/wacs* ~/public_html/cgi-bin/wacs/
# cp tag/wacs* ~/public_html/cgi-bin/wacs/
# cp security/wacslogin ~/public_html/cgi-bin/wacs/
# cp security/wacslogout ~/public_html/cgi-bin/wacs/
# cp security/wacspref ~/public_html/cgi-bin/wacs/
# cp manage/wacs* ~/public_html/cgi-bin/wacs/
# chmod 755 ~/public_html/cgi-bin/wacs/wacs*
#
```

   For those who use `.pl` extensions, do the following:

```
# cd unpack_location
# mkdir ~/html/apps
# cp index/wacs* ~/html/apps/
# cp models/wacs* ~/html/apps/
# cp presentation/wacs* ~/html/apps/
# cp retrieval/wacs* ~/html/apps/
# cp search/wacs* ~/html/apps/
# cp tag/wacs* ~/html/apps/
# cp security/wacslogin ~/html/apps/
# cp security/wacslogout ~/html/apps/
# cp security/wacspref ~/html/apps/
# mkdir ~/html/admin
# cp manage/wacs* ~/html/admin/
# chmod 755 ~/html/apps/wacs*
# chmod 755 ~/html/admin/wacs*
```

```
#
```

You will then need to rename each command to having a .pl extension - as in: `mv wacsindex wacsindex.pl` in both the apps and admin directories

5. As described above, the next step is to make copies of those wacs applications that are merely versions of existing applications with alternative default values. In most cases, this will be changing variables called either *`thumbsmode`* or *`vidmode`* as appropriate.

```
# cd ~/public_html/cgi-bin/wacs
# cp wacsmodelpage wacsmpthumbs
# editor wacsmpthumbs
# cp wacsmodelpage wacsmpmini
# editor wacsmpmini
# cp wacsmodelpage wacsmpfull
# editor wacsmpfull
# cp wacsimgcats wacsvidcats
# editor wacsvidcats
# cp wacsimgcats wacsphotcats
# editor wacsphotcats
# cp wacsimglist wacsvidlist
# editor wacsvidlist
# cp wacsimglist wacssetlist
# editor wacssetlist
# cp wacsnewsets wacsnewvideo
# editor wacsnewvideo
# cp wacsshow wacsvidshow
# editor wacsvidshow
# cp wacsindex wacspage
# editor wacspage
# cp wacsimgselect wacsvidselect
# editor wacsvidselect
#
```

6. One additional step we have to take within the web hosting environment is to add symbolic links within the `cgi-bin` directory back out to the `~/perl` directory in order that our web applications can pick up the Wacs perl modules. This is done as follows:

```
# cd /home/yoursite/public_html/cgi-bin/wacs
# ln -s /home/yoursite/perl/Wacs.pm
# ln -s /home/yoursite/perl/WacsUI.pm
# ln -s /home/yoursite/perl/WacsStd.pm
# ln -s /home/yoursite/perl/WacsId.pm
#
```

Once again if you also want to support the WACS PHP API, you'll also need to make the appropriate links so that the php applications can find the Wacs PHP modules. Here the location depends on what skin or code you're using and will need to be present in *each* directory in which you have Wacs-PHP API based applications. Assuming you're going to be installing the Simple Skin, this will be in a directory called `simple` within your `public_html` web document tree. This can be done as follows:

```
# cd /home/yoursite/public_html/simple
# ln -s /home/yoursite/php/wacs.php
# ln -s /home/yoursite/php/wacsui.php
# ln -s /home/yoursite/php/XMLSimple.php
#
```

7. The next step is to install the Php applications and the simple skin if you intend to use them. This step is optional. There are two main sets of components and you can install either one or both of them. The first group are the apps - currently the dynamic model selector (`modelsel.php`) and matching set selector (`setsel.php`). The second group is the sample website - the Simple skin - mentioned earlier. The apps come from the `apps` directory within the Wacs-PHP archive. You may need to make the necessary `wacs` directory within `public_html` if it does not already exist.

   ```
   # cd /home/yoursite/public_html/wacs
   # cp wacs-php_unpack_location/apps/modelsel* .
   # cp wacs-php_unpack_location/apps/setsel* .
   # cp wacs-php_unpack_location/skins/simple/*.php .
   # cd styles
   # cp wacs-php_unpack_location/skins/simple/styles/* .
   #
   ```

   The final aspect of file installation that needs to be covered is to provide the necessary jQuery library used by the two dynamic php apps, **modelsel** and **setsel**. There are a number of approaches to this but basically a suitable version of the file `jquery/jquery.min.js` needs to be available. The URL that the apps will ask for is defined in the **server** section of the Wacs configuration file using the variable **jqueryurl**. Basically you can set this to what you like, using the publicly hosted versions at Google, installing this through your administration panel on your hosting provider or simply by downloading it direct from jquery.org [http://www.jquery.org] and placing it within the document tree of your site (ie in `public_html`). You will need to adjust the entry in the `wacs.d/wacs.cfg` file, server section accordingly. We are currently using JQuery major version 3 with Wacs-PHP but we're not very demanding on what we need; if your hosting provider is offering an older or newer version, it's probably not an issue.

   ### Note

   Please be aware that earlier versions of Wacs prior to 1.x used an alternative library for dynamic exchanges (aka *AJAX*) called XAJAX. We do still include a version of this in the Wacs distributions for anyone who developed code using it for use with Wacs. None of the code we supply needs it anymore and it will be removed at some point in the future. If you are bringing across existing custom applications that use it, you will also need to unpack it from the wacs-php archive.

Configuration

8. This is one area where the procedure for a Web Hosting Site is significantly more complex than that for a conventional Wacs install as practically all of the entries related to database and file system locations (`fsloc`) will need tuning based upon actual layout of the account on the web hosting site.

   The first step is to create a suitable sub-directory for the wacs configuration files, ideally in the top level of your file space. If at all possible *DO NOT* put it in the web space directory as it contains passwords and other configuration items which you do not want everyone to be able to access. Next you want to find out the full path name of your top level directory, for which you use the **pwd** to the Linux/ Unix shell.

```
# pwd
/home/yoursite
#
```

Your wacs configuration files will therefore live in `/home/yoursite/wacs.d` instead of the normal location of `/etc/wacs.d`. To make this change we have to go into the `Wacs.pm` perl module in the `perl` sub-directory and make the appropriate change. We could avoid this *if* the web hosting provider would allow us to establish the `WACS_CONFIG` environment variable in the appropriate virtual server configuration section in the **apache** web server. Our hosting provider would not do this for us so we have to work around that by modifying the default value in the `Wacs.pm` perl module itself. This is a little unfortunate in that it'll mean we have to modify this module each and every time there is a Wacs code update that affects it.

Here's the appropriate change you need to make - what we've done here is copy the line setting the existing value called `default_location1` and commented out the original version with a hash (`#`) symbol at the start of the line. We've then edited the copy to have the new location on our web hosting provider's site there as the first default location:

```
# Assumptions
my $fssep = '/';

# Where to find the WACS configuration
#my $default_location1="/etc/wacs.d";
my $default_location1="/home/yoursite/wacs.d";
my $default_location2="/usr/local/etc/wacs.d";
my $default_location3="/opt/wacs/etc/wacs.d";
my $default_specifier="WACS_CONFIG";
```

Note

If you're also planning to use the PHP version of the Wacs API you will need to make identical changes to the wacs.php file provided with that.

The next step is to copy into this new `wacs.d` directory a sample `wacs.cfg` and `wacs.acl` file. We've provided a sample pair that will hopefully be a good starting point in the `conf/WebHost` sub-directory of the wacs source distribution. Please do make sure that you manually create at least each toplevel directory under your account's home area - ie `run`, `cache`, etc.

Database Setup

This is an area within the process where unfortunately we can't give you much help as it will vary between the different web hosting providers. With our provider for the PinkMetallic.com [http://www.pinkmetallic.com] domain, there was a control panel for creating MySQL Databases which consisted of three steps: `Create New Database`, then `MySQL Users: Add New User` and finally `Add User To Database`. There was an additional option of `Modify Databases` which we had no cause to need at this point.

On another web host we worked on, when we created the MySQL database in the control panel, it gave us a database connect string to use to connect to the database server on which our instance is hosted. This required the use of the specified host URL with the `-h` option to mysql. We also had to include the server host specification in the connect strings in `wacs.cfg`.

With these steps done in the appropriate order, we got a MySQL database account we could log into using the following:

```
# mysql -u yoursite_wacs -p yoursite_wacs
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 206917
Server version: 5.0.81-community MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Database Schema Creation

9. login as the database user just created and run the table create SQL script from the creation directory of the wacs distribution. These scripts are called by a single creation script, the one for MySQL is called `create_mysql.sql` and that is sometimes the only option available on web hosting sites. Those sites that do support PostgreSQL should find that the `create_postgres.sql` script will work as it should in these environments. At this point we haven't tested this option so please let us know if you try it what issues you encounter. One hosting provider still had the older MySQL 5.0 version as their offering and we had only one minor problem with it (see the section called “Final Notes For Web Hosting”).

   To run this on MySQL 5.1 using the account created in the step above, you would do the following (the only difference for a web hosting service is that you probably have a *`yoursite_`* prefix on the user and database names):

   ```
   % cd unpack_location/creation
   % mysql --user=wacs --password=wacs wacs < create_mysql.sql
   WACS Database Table Creation Script for MySQL
   Commencing Table Creation:
     1. Photographer
     2. Vendor
     3. Sets
     4. Models
     5. Assoc
     6. Idmap
     7. Download
     8. Tag
     9. Conn
    10. Keyword
    11. User
    12. Attrib
    13. Notes
   Tables Created - Committing Changes
   Completed.
   %
   ```

Support Scripts

The procedure here is almost exactly the same as above except of course that we cannot add things to `/usr/local/bin` and so we have to place them within our account's home directory.

```
# cd unpack_location
# cp -p tools/* ~/bin
# cp -p download/* ~/bin
# cp -p migrate/* ~/bin
#
```

Additionally you ideally want to ensure that the *PERL5LIB* environment variable is established within your shell environment in order to use the command line tools. The alternative is to put symbolic links in each directory to where the WACS perl module files are to be found. Setting the PERL5LIB environment variable can be done initially with:

```
# PERL5LIB=/home/yoursite/perl
# export PERL5LIB
#
```

To add it to your shell configuration so it is always established, you'll need to edit your shell start-up file (usually `.bashrc` in your home directory) and add the following line at the bottom of it:

```
export PERL5LIB=/home/yoursite/perl
```

Populate The Initial Database

Note

Wacs now doesn't give a running commentry on these steps as they were playing havoc with the formatting on **wacssetup**. If you want to see them, simply edit the `wacs.cfg` debug section, look for the **util_wacspop** entry and set it to 1 or higher.

10. The next step is to populate the vendor database with the sample records, which can be done with:

    ```
    # cd unpack_location/populate
    # ./wacspop vendors.xml
    Inserting Entries For Site: ATKP
    Inserting Entries For Site: AMK
    Inserting Entries For Site: ATE
    Inserting Entries For Site: SE
    #
    ```

    Please contribute back vendor descriptions you create to be included in the next release.

11. Next we need to preload the keywords database table so that the automatic tagging will occur correctly. We do this with:

    ```
    # cd unpack_location/populate
    # ./wacspop keywords.xml
    [...]
    #
    ```

12. Next we need to load the photographers database with some initial example records, which can be done with:

```
# cd unpack_location/populate
# ./wacspop photographers.xml
[...]
#
```

13. Finally we need to load the attributes database with the basic attributes we will be using for searches and markup:

```
# cd unpack_location/populate
# ./wacspop attrib.xml
[...]
#
```

Initial User Creation for Database Authentication

If you're planning to use database authentication on your site, there's a catch 22 situation to get around first when you're doing a manual install. That is that although you can edit and create users using the **wacsusermgr** web application, you can't log in to the system in order to use the application until an initial account has been created. This is not so much of a problem with a normal root account install as this will initially be set up for host authentication which can be used to login that first time, create a user account and then switch over to the database authentication method. For a web site hosting provider where the host authentication method doesn't work, it's a lot more problematic. We now offer two solutions to this problem.

In WACS 0.9.1 we introduced a new users.xml file that sets up three accounts for you - `root, support` and `guest` . These give you an example of each of the possible types of user account - **admin, power** and **viewer**. This is ideal for testing and familiarisation activities but you really *MUST* remember to change these accounts from their default passwords before opening the server up to the internet. Their initial passwords are `Svjck981` for the **root** account, `uKiFt126` for the **support** and `freebie` for the **guest** account. To add these accounts, run the following:

```
# cd unpack_location/populate
# ./wacspop users.xml
[...]
#
```

The other solution is to manually create the necessary user account using the SQL command line tool. Here is an example of how to do this:

```
# mysql -u yoursite_wacs -p yoursite_wacs
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 206917
Server version: 5.0.81-community MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> insert into wacsuser
     > (userid,username,upassword,ustatus,utype,uclass,uadded)
     > values(1,'wacs','badpasswd','A','A','admin','2012-11-11');
```

```
Query OK, 1 row affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
Bye
#
```

Once you've created this account you will be able to log in to a database authenticating WACS system and create other users through the Web GUI.

Final Notes For Web Hosting

Before bowing out on this particular topic, we thought we would just collect together a few random notes and observations which may help you with completing a successful webhost installation.

Webhost History

The first details for how to setup WACS for use in a Web Hosting Provider's environment were introduced in 0.8.4 and are based on our own experience setting up PinkMetallic.com [http://www.pinkmetallic.com] at justhost.com [http://www.justhost.com]. We've since set WACS up for a consultancy client on godaddy.com [http://www.godaddy.com] - you need to use Wacs 0.9.1 or higher for it to work on godaddy because of the need to use .pl extensions for the command names. Please note this is not a specific endorsement of these two providers; merely a point of reference on what has been successfully achieved in the past.

Warning

Facilities and configurations do differ between web hosting providers and there is no guarantee that any specific provider's offering is suitable for running WACS. We do try to help were we can, and www.bevteccom.co.uk [http://www.bevteccom.co.uk] do offer WACS installation consultancy services. Please report any problems you find to us using the facilities on sourceforge or our email addresses.

Known Issues

There's a known issue with the security settings when you use the **wacsimport** and (probably) **wacsxmlin**. Once these have created the icons, they change the permissions of the owning directory to be accessible only by the user and group. This is not the correct strategy on a web hosting provider as their apache web server is not a member of any unix group in common with the shell user account. The quick solution to this is to run the following on the web document tree (usually under `~/public_html/` after each importation has taken place:

```
# cd ~/public_html
# chmod -R o+rX bigicons cache modicons images wacs
#
```

Another issue, with one web hosting provider we worked on, we found that the wacspop command wouldn't work due their using MySQL 5.0 and an earlier version of the interface routine. This problem manifested itself in an error from the `column_info` - changing the empty string specifiction from `''` to a wildcard `'%'` in the final parameter worked around this problem.

Adding Missing Helper Programs

Unfortunately there can be further issues when trying to use the WACS system in that many web hosting providers do not install all of the infra-structure that it needs, including packages like the netpbm tools which provide the image scaling and thumbnailing facilities we use extensively within WACS. It is quite possible to copy both the binary programs and the necessary shared libraries from another Linux host of similar architecture onto your space on the web hosting provider. The alternative approach is to compile the netpbm suite from source code and then statically link the resulting binaries. This is what we had to do to make the PinkMetallic.com [http://www.pinkmetallic.com/] site work. The commands below show what we did - the first command being to determine the CPU/runtime architecture being used - in this case standard 64-bit AMD/Intel EMT architecture, and the second to determine the distribution being used:

```
# uname -p
x86_64
# uname -r
2.6.18-128.1.6.el5
# mkdir ~/lib
# export LD_LIBRARY_PATH=/home/yoursite/lib
#
```

Tip

We understand this step is non-trivial and confusing and there's very little we can do about this other than to invite you to post on the wacs-users mailing list for help and advice. Each web hosting provider probably has a different combination and there is little else we can do other than offer to help as best we can.

This tells us (from experience) that we're dealing with a RedHat Enterprise Linux distribution running on a 64-bit AMD-style architecture processor. We might also see i686 if the hosting company are using 32-bit Intel architecture servers. You can also try the following for more details if it looks like it might be either a redhat-based (CentOS,RHEL,Oracle Unbreakable,etc) or debian-based (Ubuntu,Debian,etc) Linux distribution they're using:

```
# cat /etc/redhat-release
CentOS release 5.5 (Final)
# cat /etc/debian_version
cat: /etc/debian_version: No such file or directory
#
```

For CentOS or RHEL, **centos.org** is the best place to find suitable packages for download - we selected two RPMs from one of their mirror sites: `netpbm-10.35.58-8.el5.x86_64.rpm` for the libraries and `netpbm-progs-10.35.58-8.el5.x86_64.rpm` for the tool binaries. These can usual be found starting at the release number, say `5.5` , then choosing `os` then choosing the architecture either `x86_64` or `i386` (an alias for i686) as appropriate. The package files themselves are then usual found in the `CentOS` directory. We then used the rpm2cpio command to covert these rpm archives into more standard archive formats and then the cpio command itself to unpack the result.

Tip

If the webhosting company is using an older version of CentOS and you can't find the appropriate binaries on the mirror site, try `vault.centos.org`

```
# wget http://www.mirrorservice.org/sites/mirror.centos.org/5.4/o
s/x86_64/CentOS/netpbm-10.35.58-8.el5.x86_64.rpm
[...]
# rpm2cpio netpbm-10.35.58-8.el5.x86_64.rpm > netpbm-10.35.cpio
# cpio -ivudB < netpbm-10.35.cpio
[...]
# wget http://www.mirrorservice.org/sites/mirror.centos.org/5.4/o
s/x86_64/CentOS/netpbm-progs-10.35.58-8.el5.x86_64.rpm
[...]
# rpm2cpio netpbm-progs-10.35.58-8.el5.x86_64.rpm > netpbm-progs-
10.35.cpio
[...]
# cpio -iuvdB < netpbm-progs-10.35.cpio
[...]
#
```

Having done this we have the entire netpbm tree contained in a directory called `usr` under the current directory. The first command the wacs program we were trying to run (actually it was **wacsgenimg**) complained about was **pnmscale** so we fetch the **pnmscale** binary from `usr/bin/pnmscale` and transfer that up to the web hosting provider and place it in our `~/bin` directory. Once we've done that we run the `ldd` command on it which says:

```
# cd ~/bin
# ldd pnmscale
 libm.so.6 => /lib64/libm.so.6 (0x000000365ee00000)
 libnetpbm.so.10 => not found
 libc.so.6 => /lib64/libc.so.6 (0x000000365e200000)
 /lib64/ld-linux-x86-64.so.2 (0x000000365de00000)
#
```

The pertinent thing here is that libnetpbm.so.10 is the only library we need for **pnmscale** to work that isn't there. If we check back in our unpacked package, we should have that file as `usr/lib64/libnetpbm.so.10.35`. Since it's **libnetpbm.so.10** that it's asking for, we transfer our local `usr/lib64/libnetpbm.so.10.35` up to the hosting provider server as **libnetpbm.so.10** using the ftp/sftp put command as follows:

```
sftp> put usr/lib64/libnetpbm.so.10.35 libnetpbm.so.10
Uploading usr/lib64/libnetpbm.so.10.35 to /home/yoursite/lib/libnetpbm.so.10
usr/lib64/libnetpbm.so.10.35                  100%  187KB  93.3KB/s   00:02
sftp>
```

If we now re-run the ldd command on **pnmscale**, we should now see that all the dependencies are resolved. The final test is to run the command itself, and it this case it's default action is to complain that you didn't ask it to do anything. All that remains now is to do the same procedure of copying up the other files from `usr/bin` that Wacs is asking for. We found we needed the following: **pbmtext, pnmcat, pnmscale, pnmtojpeg, pnmtopng, ppmtogif, pngtopnm, giftopnm** and **jpegtopnm**.

The first of the two final things we have to do is to check that everything under the `~/public_html/cache` tree is publicly writeable (gulp!) because we share no groups in common with the web server. The second is to add the `LD_LIBRARY_PATH` variable to the environment used by the web server so that the netpbm commands actually work when invoked by the wacs commands themselves. Fortunately the *`dbienvvar`* and *`dbienvvalue`* variables originally added to enable Oracle to be supported can be used for this purpose. In the `wacs.cfg` file, set *`dbienvvar`* to `LD_LIBRARY_PATH` and *`dbienvvalue`* to the path to your library directory where you placed the `libnetpbm.so.10` file; it's probably something along the lines of `/home/`*`yoursite`*`/lib`. If you're using our sample WebHosting configuration file as a basis, we may have already set this one up for you. You will also need to add a suitable entry in your `.bashrc` or `.cshrc` to set LD_LIBRARY_PATH for your shell.

Chapter 8. Manual Installation

Caution

This is a complex task and some level of familiarity with the Unix/Linux command line will probably be needed to be successful. Please make sure that all of the packages/services described in the prequisites chapter (Chapter 3, *Prerequisites*) have been installed and are running correctly.

Manual Installation: Steps

Preparation

1. Create the wacs user account and group and place your own username in the group file (`/etc/group` or distributed name service equivalent). This is easily done with the useradd command on Redhat-based distributions:

   ```
   # groupadd -r wacs
   # useradd -m -g wacs -r -c "WACS Files Owner" \
     -s /bin/bash wacs
   #
   ```

2. First of all check the web server is working by pointing your web browser to the address you expect it to be - you should see either the standard "It Works!" message or your existing web site. You have two basic choices as to how to install the WACS web tree - one is to place it within the existing web document tree, the other is to explicitly add the wacs directory into a custom configuration for the web server. The package installs of WACS place the document tree in `/usr/share/wacs/html` as part of placing all of the wacs system within `/usr/share/wacs`. If you're undecided as to which to use, we would recommend using the dedicated WACS directory tree either in `/usr/share/wacs` or `/opt/WACS` based on the usual convention on the Operating System you're using.

   You need to install the barebones WACS html index pages from the `htmlbones` directory of the distribution into your web tree and check you can see it. On a conventional Linux installation, this would be done with:

   ```
   # cd unpack_location
   # mkdir /usr/share/wacs/html
   # cp -rp htmlbones/* /usr/share/wacs/html
   #
   ```

   You then need to add the configuration file adjunct into the appropriate web server configuration directory - on Ubuntu this is to be found in `/etc/apache2/conf.d/` and in RedHat flavours of Linux in `/etc/httpd/conf.d` - it's likely to be something similar on all *nix systems. Typically the Apache web server picks up any file ending with `.conf` in this directory so the logical name is `wacs.conf`. This file should contain the following, adjusted appropriately for the location you have choosen:

   Example 8.1. Sample Apache `wacs.conf` File

```
# Bring in the wacs icons, glyphs and stylesheets from the /usr/share area.
Alias /wacs /usr/share/wacs/html

<Directory /usr/share/wacs/html>
  AllowOverride Options FileInfo
</Directory>

# add in the scripts in the cgi-bin directory
ScriptAlias /cgi-bin/wacs /usr/share/wacs/cgi-bin
```

Tip

If this proves not to work (Ubuntu had a problem with it), you can add a symbolic link into `/usr/lib/cgi-bin` pointing to `/usr/share/wacs/cgi-bin`.

3. For the perl modules, first check whether your operating system distribution includes them - Redhat Flavours usually has packages called `perl-DBI`, `perl-DBD-MySQL` and `perl-XML-Simple` - so these could be simply installed with **yum install perl-DBI**, **yum install perl-DBD-MySQL** and **yum install perl-XML-Simple**. On Ubuntu these are called `libdbi-perl`, `libdbd-mysql-perl` and `libxml-simple-perl` respectively.

   The next easiest way to install the necessary perl modules, if they are not already present, is to use the cpan command. On some recent releases, the cpan command has become optional - you will have to do a **yum install cpan** on Redhat systems - Ubuntu is currently still installing it as part of perl. Once you have cpan, the necessary perl module installs can typically be done with:

```
# cpan XML::Simple
# cpan Data::Dumper
# cpan File::Basename
# cpan MIME::Base64
#
```

The Wacs Code Itself

4. The next step is to install the WACS perl modules into an appropriate perl library directory on your system. If you run **perl -V**, the perl interpreter will give you a list of the directories it will look in for perl modules. Generally the best idea is to use that doesn't include a specific version number, and ideally one that is obviously seperate from where the Operating System places it's versions of perl modules. On both Ubuntu and Redhat flavours of Linux, this would be `/usr/share/perl5`.

   First of all install the `WacsId.pm` perl module into the `/usr/share/perl5` directory (or the appropriate one) of your system. [NB: note the change of case of the first letter of the perl module name from wacs.pm to Wacs.pm]. Then create a directory within there called `Wacs` and copy into there the **WacsUI.pm, WacsStd.pm, WacsId.pm** and **WacsInstUtil.pm** modules:

```
# cd unpack_location
# cp modules/wacs.pm /usr/share/perl5/Wacs.pm
# mkdir /usr/share/perl5/Wacs
# cp modules/wacsui.pm /usr/share/perl5/Wacs/WacsUI.pm
# cp modules/wacsstd.pm /usr/share/perl5/Wacs/WacsStd.pm
# cp modules/wacsid.pm /usr/share/perl5/Wacs/WacsId.pm
# cp install/WacsInstUtil.pm /usr/share/perl5/WacsInstUtil.pm
```

```
#
```

5. The next step is to install the PAM modules and **pam_auth** program but you only need to do this if you want the host operating system to authenticate users for you. If you're going to be using user accounts in the database itself, you can skip this step completely. First install the wacs PAM (Plugable Authentication Modules) configuration into the `/etc/pam.d` directory. You will also need to compile the pam_auth program using the provided make file and then install the binary created into whereever your tooldirs configuration variable is set to (a common value is `/usr/bin` but `/usr/local/bin` or even `/opt/WACS/bin` would probably be better). If this compilation fails, the most likely cause is that the libpam development package is not installed. Then you need to create the `/var/run/wacs` directory where the dynamic leases files are stored and change it's ownership to apache (or whatever your web server user is).

   ```
   # cp unpack_location/security/wacs.pam /etc/pam.d/wacs
   # chown root.root /etc/pam.d/wacs
   # chmod 644 /etc/pam.d/wacs
   # cd unpack_location/security
   # make -f Makefile all
   Building pam_auth.x86_64-Fedora8 ...
   cc -o pam_auth.`arch`-`lsb_release -si | sed 's/\ /_/g'``lsb_release -sr`
    pam_auth.c -lpam
   # ls pam_auth*
   pam_auth    pam_auth.c   pam_auth.x86_64-Fedora8
   # cp pam_auth.x86_64-Fedora8 /usr/local/bin/pam_auth
   # chown root.wacs /usr/local/bin/pam_auth
   # chmod u+s /usr/local/bin/pam_auth
   # mkdir /var/run/wacs
   # chown apache.apache /var/run/wacs
   #
   ```

   ### Note

   if you run selinux (Security Enhanced Linux) on Fedora Core or Redhat (or another future distro that includes it), you will need to give apache privilege to read the /var/run/wacs directory - this can be done by changing the context of the directories and files. The commands to do this are:

   ```
   # chcon system_u:object_r:httpd_sys_content_t /var/run/wacs
   # chcon -R system_u:object_r:httpd_sys_content_t /var/run/wacs/*
   #
   ```

   If the leases file does not exist when you first do this and you encounter problems, try using the second of these two commands again.

6. install the wacs application programs into the cgi-bin tree:

   ```
   # cd unpack_location
   # cp index/wacs* models/wacs* presentation/wacs* /usr/share/wacs/cgi-bin/
   # cp retrieval/wacs* search/wacs* tag/wacs* /usr/share/wacs/cgi-bin/
   # cp security/wacslogin /usr/share/wacs/cgi-bin/
   # cp security/wacslogout /usr/share/wacs/cgi-bin/
   # cp security/wacspref /usr/share/wacs/cgi-bin/
   ```

```
# cp manage/wacs* /usr/share/wacs/cgi-bin/
# chmod 755 /usr/share/wacs/cgi-bin/wacs*
#
```

7. install the wacs application programs into the cgi-bin tree:

```
# cd unpack_location
# cp index/wacs* models/wacs* presentation/wacs* /usr/share/wacs/cgi-bin/
# cp retrieval/wacs* search/wacs* tag/wacs* /usr/share/wacs/cgi-bin/
# cp security/wacs* manage/wacs* /usr/share/wacs/cgi-bin/
# chmod 755 /usr/share/wacs/cgi-bin/wacs*
#
```

8. copy the applications that are just duplicate versions of existing commands and change the appropriate mode variables:

```
# cd /usr/share/wacs/cgi-bin
# cp wacsmodelpage wacsmpthumbs
# editor wacsmpthumbs
# cp wacsmodelpage wacsmpmini
# editor wacsmpmini
# cp wacsmodelpage wacsmpfull
# editor wacsmpfull
# cp wacsimgcats wacsvidcats
# editor wacsvidcats
# cp wacsimgcats wacsphotcats
# editor wacsphotcats
# cp wacsimglist wacsvidlist
# editor wacsvidlist
# cp wacsnewsets wacsnewvideo
# editor wacsnewvideo
# cp wacsshow wacsvidshow
# editor wacsvidshow
# cp wacsindex wacspage
# editor wacspage
# cp wacsimgselect wacsvidselect
# editor wacsvidselect
#
```

edit the file and change the mode variable (thumbsmode in this case). Repeat this process for wacsimgcats becomes wacsvidcats and wacsphotcats, and so on. At the end, make sure all of the copies are executable:

```
# cd /usr/share/wacs/cgi-bin
# chmod 755 wacs*
```

Configuration

9. install the configuration file, `wacs.cfg` into a suitable location such as `/etc/wacs.d` or `/usr/local/etc/wacs.d`. Edit this file and make sure the key settings are right for your server, specifically the location of the image files, the location of the video files and the server name in the URLs. You will also need settings for the database user name and password you intend to use, and the

environment and path locations needed for the database system you are using. For more information, see the Configuration Guide.

10. create a suitable permanent access control list in the configuration directory choosen above, the supplied `wacs.acl` should provide a suitable template. This step can be skipped if you're only ever going to use lease-based access with logins. For more information on the format of the access control lists, please see the section on security in the Configuration guide.

Database

11. create a suitable owner account for the wacs data tables in your database system. The instructions here cover doing this for both MySQL and Oracle 10g, in that order. With MySQL 5.x, this would be done with:

```
% mysql --user=root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17 to server version: 5.0.22

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database wacs;

mysql> create user 'wacs'@'myserver' identified by 'wacs';

mysql> create user 'wacs'@'localhost' identified by 'wacs';

mysql> grant all on wacs.* to wacs;

mysql> commit;

mysql> flush privileges;

mysql> quit
```

With Oracle 10g, this would be something like:

```
% sqlplus
SQL*Plus: Release 10.1.0.3.0 - Production on Fri Sep 29 14:53:56 2006
Copyright (c) 1982, 2004, Oracle.  All rights reserved.

Enter user-name: system
Password: *******

Connected to:
Oracle Database 10g Release 10.1.0.3.0 - 64bit Production

SQL> create user wacs identified by wacs;

User created.

SQL> grant connect, resource to wacs;
```

```
Grant succeeded.

SQL> alter user wacs default tablespace main
   quota unlimited on main;

User altered.

SQL> commit;

Commit complete.

SQL> quit
Disconnected from Oracle Database 10g Release 10.1.0.3.0 - 64bit Production
```

Of course there is a huge amount of variance in how any given database is installed, so you will need some knowledge about your installation. For instance, your main tablespace may not be called "main" as it is in the example.

Database Schema Creation

12. login as the database user just created and run the table create SQL script from the creation directory of the WACS distribution. These scripts are called by a single creation script, the one for oracle is called `create_oracle.sql`, the one for MySQL is called `create_mysql.sql`. To run this on MySQL 5.1 using the account created in the step above, you would do the following (the only difference for a web hosting service is that you probably have a *`yoursite_`* prefix on the user and database names):

```
% cd unpack_location/creation
% mysql --user=wacs --password=wacs wacs < create_mysql
[...]
%
```

To run this on Oracle 10g using the account created in the step above, you would do the following:

```
% cd unpack_location/creation
% sqlplus wacs/wacs @create_oracle
SQL*Plus: Release 10.1.0.2.0 - Production on Fri Oct 6 19:11:41 2006
Copyright (c) 1982, 2004, Oracle.  All rights reserved.

Connected to:
Oracle Database 10g Release 10.1.0.3.0 - 64bit Production

WACS Database Table Creation Script for Oracle

Commencing Table Creation:

  1. Photographer


Table created.

  2. Vendor
```

```
Table created.

  3. Sets


Table created.

  4. Models


Table created.

  5. Assoc


Table created.

  6. Idmap


Table created.

  7. Download


Table created.

  8. Tag


Table created.

  9. Conn


Table created.

 10. Keyword


Table created.

 11. User


Table created.

 12. Attrib


Table created.

 13. Notes
```

```
Table created.

Tables Created - Committing Changes


Commit complete.

Completed.

Disconnected from Oracle Database 10g Release 10.1.0.3.0 - 64bit Production
%
```

Support Scripts

13. The penultimate major activity is to install the tools scripts, and if required the download and migrate tools, into a suitable directory, normally this would be `/usr/local/bin`, but it could be put within the wacs tree if desired. `/usr/local/bin` is usually in the default path for all the shells and thus available to user accounts without further work. To install, do:

```
# cd unpack_location
# cp -p tools/* /usr/local/bin
# cp -p download/chkmodel /usr/local/bin
# cp -p download/getarc /usr/local/bin
# cp -p download/refresh /usr/local/bin
# cp -p migrate/* /usr/local/bin
#
```

If you want to put it somewhere else, within the wacs home area would be fine, somewhere like `/home/wacs/bin`, but you will then need to add that directory to the path of your shell. For the C-shell, you would add `set path=(/home/wacs/bin $path)` into the .cshrc file in the home directory of your own account and those of other people who might be adding contents to the wacs server. For the Bourne style shells (sh,bash,etc), you would need to add `PATH=/home/wacs/bin:$PATH` and `export PATH` to the .profile or .bashrc files in the home directories. Once added, depending on the shell, you may need to type `rehash` to rescan the path for the new commands.

Populate The Initial Database

Note

In recent releases of Wacs we have suppressed the confirmatory `Inserting Entries For...` messages as they were playing havoc with the formatting on **wacssetup**. If you want to see them, simply edit the `wacs.cfg` debug section, look for the **util_wacspop** entry and set it to 1 or higher.

14. The next step is to populate the vendor database with the sample records, which can be done with:

```
# cd unpack_location/populate
# ./wacspop vendors.xml
Inserting Entries For Site: ATKP
Inserting Entries For Site: AMK
Inserting Entries For Site: ATE
Inserting Entries For Site: SE
#
```

Please contribute back vendor descriptions you create to be included in the next release.

15. Next we need to preload the keywords database table so that the automatic tagging will occur correctly. We do this with:

```
# cd unpack_location/populate
# ./wacspop keywords.xml
[...]
#
```

16. Next we need to load the photographers database with some initial example records, which can be done with:

```
# cd unpack_location/populate
# ./wacspop photographers.xml
[...]
#
```

17. Finally we need to load the attributes database with the basic attributes we will be using for searches and markup:

```
# cd unpack_location/populate
# ./wacspop attrib.xml
[...]
#
```

Initial User Creation for Database Authentication

If you're planning to use database authentication on your site, there's a catch 22 situation to get around first when you're doing a manual install. That is that although you can edit and create users using the **wacsusermgr** web application, you can't log in to the system in order to use the application until an initial account has been created. This is not so much of a problem with a normal root account install as this will initially be set up for host authentication which can be used to login that first time, create a user account and then switch over to the database authentication method. For a web site hosting provider where the host authentication method doesn't work, it's a lot more problematic. We now offer two solutions to this problem.

In WACS 0.9.1 we introduced a new users.xml file that sets up three accounts for you - `root, support` and `guest` . These give you an example of each of the possible types of user account - **admin, power** and **viewer**. This is ideal for testing and familiarisation activities but you really *MUST* remember to change these accounts from their default passwords before opening the server up to the internet. Their initial passwords are `Svjck981` for the **root** account, `uKiFt126` for the **support** and `freebie` for the **guest** account.

The other solution is to manually create the necessary user account using the SQL command line tool. Here is an example of how to do this:

```
# mysql -u wacs -p wacs
```

```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 206917
Server version: 5.0.81-community MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> insert into wacsuser
     > (userid,username,upassword,ustatus,utype,uclass,uadded)
     > values(1,'wacs','badpasswd','A','A','admin','2012-11-11');
Query OK, 1 row affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
Bye
#
```

Once you've created this account you will be able to log in to a database authenticating WACS system and create other users through the Web GUI.

More On SELinux

Note

Although we have discussed the steps needed to get WACS working under SELinux, we've currently not managed to track down all of the dependencies and in the interests of expediency we have gone ahead with code development without it. At this point we do not consider the WACS 1.0.0 release or any release prior to that to be SELinux compatible.

We therefore recommend that your operating system is configured so as to ensure that SELinux is running in a reduced mode that will not block the WACS components from working. This is only an issue on Fedora and other Red Hat based releases at present. We hope to have this resolved by the next release of WACS. You can determine the current mode of SELinux using the sestatus command:

```
% /usr/sbin/sestatus
SELinux status:                 disabled
%
```

To change the normal operational mode, you need to edit the file called `/etc/sysconfig/selinux` and change the line which reads `SELINUX=enabled` to either `SELINUX=permissive` (generates big log files and slows machine but allows for SELinux to be turned back on later more easily) or `SELINUX=disabled` (which disables it completely but can cause problems in the future if you want to switch it back on). You will also probably want to disable it immediately rather than doing a reboot before you can continue working on WACS - to do this, become root and run the following:

```
# /usr/sbin/setenforce 0
setenforce: SELinux is disabled
#
```

You can check this change has taken effect by using the **sestatus** command again.

Chapter 9. Upgrading An Existing Installation

About Upgrading

Introduction

While it is obviously vital to be able to upgrade an existing WACS installation from one release to another, there are a great number of factors that come into play. It is not always easy to resolve these and with WACS we've taken a modular approach to handling the issues presented. This means that we've provided tools for the necessary changes as separate entities and then integrated them together with the existing **wacssetup** web application to provide a wide palette of options. You can choose to let **wacssetup** walk you through the whole upgrade process, or you can choose to use each tool individually to customise the process to your needs.

The key things that need to be considered are:

Table 9.1. Aspects of an Upgrade

What	Description	Application
Database Schema	This covers the shape of the database, the information it stores and how it does so	**wacsschema**
Application Code	The actual code of the applications themselves	OS Package Manager or **upgrade** command
Predefined Values	This covers the various pre-defined data like keywords, icon associations, photographer info and the like	**wacspop**
New Fields Content	This covers how new fields in existing records will be populated to allow the new applications to make use of them	`T.B.A.`

We will discuss each of the tasks and the tools available to help with it individually before then walking you through letting **wacssetup** preform the entire process for you.

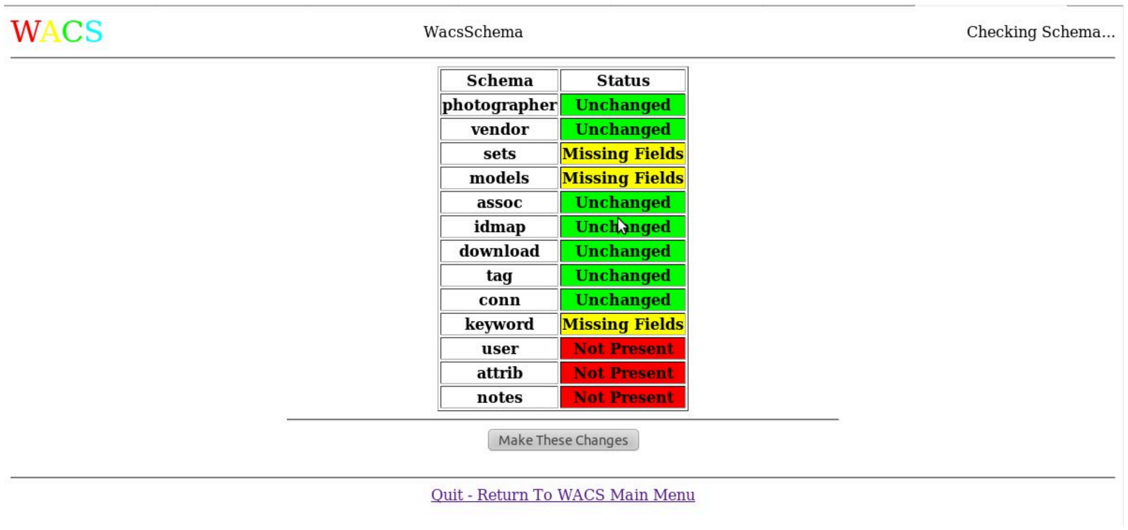
The WacsSchema Command

The new **wacsschema** compares the structure of the database to which it is currently talking with the files in the creation directory distributed with the latest version of the application code. If these differ, it provides a mechanism to update the existing database schemas to the new structure or to create new schemas where they are not present it the current database. This application is very unusual in that it has both a command line and web mode of operation, both of which function essentially in the same way. In general, it will ignore local customisations and leave them untouched during the upgrade process.

Warning

Changing the database structure is by it's very essence a tricky process and you should make absolutely sure you have good backups before attempting it. Seriously, please don't ignore this warning!

The first page shown to you by the **wacsschema** is a simple summary of the current state and takes no actions, so it's not *fundamentally* dangerous to invoke it. Just don't click on the `Make These Changes` button unless you're very sure and have backups! Shown below is typical of what it says if run on an upgraded installation that still has a pre-Wacs 0.8.5 database schema:



Schema	Status
photographer	Unchanged
vendor	Unchanged
sets	Missing Fields
models	Missing Fields
assoc	Unchanged
idmap	Unchanged
download	Unchanged
tag	Unchanged
conn	Unchanged
keyword	Missing Fields
user	Not Present
attrib	Not Present
notes	Not Present

As you will see from this, it has determined that there are missing fields from the `sets, models` and `keyword` database schemas; that `user, attrib` and `notes` schemas are missing entirely. All the other database schemas are found to be complete and up to date. In the web version, it gives you the option to click on the `Make These Changes` button and it will run the appropriate SQL (we *HOPE*) to alter and/or create the database schemas as necessary.

Note

wacsschema does not *remove* any fields it does not recognise, so existing additional custom database fields will be left intact. Additionally it uses the SQL scripts relevant to the SQL Database type being used (ie MySQL5, Oracle, etc) from the `creation` directory within the wacs installation. Were you to add extra fields into those files, **wacsschema** would attempt to add them for you.

For those of you who either prefer command line applications or who are using a web hosting service where **wacssetup** doesn't work, **wacsschema** will automatically work in text-only mode if it doesn't detect that it's been run by a web server. Shown below is what it says when run in text mode:

```
% wacsschema
Schema photographer is unchanged.
Schema vendor is unchanged.
Schema sets is missing fields saltmedia, sfocus, sfps, sinter, snext,
sprev, srank, ssetpos, sskipfr.
Schema models is missing fields magency, maltimage, mbirthdate, mbody
image, mdress, mlabia, monfile, mstarsign.
Schema assoc is unchanged.
Schema idmap is unchanged.
Schema download is unchanged.
Schema tag is unchanged.
Schema conn is unchanged.
Schema keyword is missing fields kiwear, kiwscore.
Schema user is not present in current database.
Schema attrib is not present in current database.
Schema notes is not present in current database.
%
```

As you will see **wacsschema** returns to the command prompt after delivering it's verdict on the database schema. If you want the Text Mode version to actually *make* the changes, you need to run wacsschema with the *`-y`* option. Thus to actually perform the schema update, you would use:

```
% wacsschema -y
[...]
%
```

In the next chapter we will see the use of **wacsschema** as integrated into the **wacssetup** for a simple and straightforward WACS upgrade.

Software & Data Upgrade Tools

The next significant activity in the upgrade process is to ensure that we have the latest version of the program code. Now obviously if we're using a packaged installation, it's a relatively straightforward process to install the packages that relate to the new version. Unfortunately as we have no repositories on sourceforge.net, it can be a little bit more frustrating than it really should be. Still, hopefully you can find a package manager that will let you update the packages as needed.

However if you're making your own changes, or wish to run the very latest developement code straight from the Subversion repository on sourceforge, we've created a command line application called **upgrade**. This conventionally lives in the `install` directory of the source code tree and upgrades the application code to the latest. It is perfectly possible to use the **upgrade** command to place the latest subversion application code over the top of a WACS installation installed by packages and **wacssetup** . That should work just fine.

The Upgrade Command

The application code and related icons and other HTML bits are looked after by the upgrade command; to run this download and unpack the new distribution source code tree, and as the super user (root) run the following commands. Please make sure it is using the correct wacs.cfg file by setting `WACS_CONFIG` in the environment if necessary:

```
# cd unpack_location/install
# ./upgrade
WACS - Upgrade
--------------
```

```
Welcome to the WACS Upgrade script.  This script will
attempt to upgrade an existing WACS installation to the
latest version.  It does not attempt any of the initial
package installation steps, so if you've not installed
WACS before, you should be using easyinstall instead.

It will ask for confirmation before modifying any configuration
files (but as always you should have backups before upgrading
anything.

Do you wish to continue? (y/n): y
[...]
#
```

At the end of it's run, upgrade will print out some key notes about things that will require manual attention to get the new release working. The section below will give you some guidance on how these may be achieved.

Additional Steps

The upgrade command will give you some information on what extra steps you may need to take to migrate to this release. For example, it may tell you that a new database field needs to be added to a particular model schema. If you've used the **wacsschema** command to update your schema, you don't need to worry about these.

If not, you will have to make the schema changes that are needed by hand using your database's SQL command line tools. In the transition from 0.5 to 0.6.x the mrace field was added, and upgrade will tell you about this. First step is to find the specification of the field from the appropriate SQL script in the creation directory, so for Oracle this will be `creation/ora_models.sql`. From this you will see that the field specification for Oracle is:

```
[...]
 mrace            varchar2(15),
[...]
```

You have three options for adding this to the database - you can choose to alter the existing schema (may leave fields in an odd order in describe); you can rename the existing table, create the new one, copy the data across and then repoint any relational constraints to the new table; or you can export your entire database, create a fresh one and import the records back in (the tools do not cover connections, saved searches or customised infra-structure tables (keywords, vendors, photographers) at present but are otherwise quite usuable). The former is quick and easy if the database supports it but leaves the field list in an odd order; the middle one is more work but produces a fully "normal" schema in the end but requires serious black magic if your database understands relational constraints. The final one is **VERY* experimental* at this point but will improve with time.

Here is a worked example that shows how to use the alter table syntax in Oracle's SQL*Plus command interpreter to add one field called mrace:

```
% sqlplus
[...]
Username: wacs
Password: ****
```

```
sqlplus> alter table models
       > add ( mrace      varchar2(15) );

Table altered.
sqlplus> commit;

Commit complete.
sqlplus> desc models
[...]
 MRACE                  VARCHAR2(15)
sqlplus> quit
%
```

Another issue you need to be aware of is that the upgrade script will not over-write any existing files in the wacs web document tree (by default this is `/var/www/html/wacs`) because you may well have tailored them and we wouldn't want to overwrite those. You may well therefore need to look at what is in the htmlbones directory and copy some of the new files across into your web tree, or merge the new html into your modified version of the pages.

Upgrading The Data

Obviously over time the defaults data we provide for the key database tables does get added to, improved and bug fixed. How you choose to handle this very much depends on how much you've customised your keywords, vendor and photographer data relative to the distributed versions. At the simplest level, it might be nice to simply reload everything but of course if your database enforces referential integrity as it should, you can't just drop an entire data set that has relations pointing to it. The current compromise is that the wacs data importer, **wacspop** will only create new entries that are not currently in the existing database. Any entry that is already there will remain unchanged, quite possibly not benefiting as it should from an update.

Chapter 10. Using wacssetup To Upgrade

Wacssetup Does Upgrades Too...

Talented application that it is **wacssetup** can now handle upgrading an existing installation as well as configuring a new one. Well at least since Wacs 0.8.6 it can, with a little help from it's friends **wacsschema** and **wacspop**!

Backing Up The Database

Before attempting anything as fundamental as altering the database schema, it's vitally important to back up the database first. So before we actually run **wacssetup** to do the upgrade, we'll take a dump of the database to disc just in case.

Backing Up A MySQL5 Database

For MySQL5 the simplest way of taking a suitable database dump is to use the **mysqldump** supplied as part of MySQL. Typical usage is as follows:

Tip

Do make sure that you've got space to store the resulting database dump file in the current directory. **mysqldump** can be run in any directory. You might want to make a special directory to store such dumps in.

```
% mysqldump -u wacs -p wacs > wacs-db-dump.mysql
Enter password:
%
```

The parameters are `-u` followed by the username you use to access the wacs database. If you've forgotten what this is, the **dbuser** value in the `/etc/wacs.d/wacs.cfg` is the one you want for this. The next parameter is `-p` which tells it to prompt you for a password. The final parameter is the name of the wacs database; again this can be confirmed from the **dbname** value in the wacs config file. The greater-than (>) sends the output into the file we've named `wacs-db-dump.mysql` (you might want to include the date in the name). The **mysqldump** program will then prompt for a password and silently perform the dump unless there's a problem.

It's also a good idea to check the file has been created. If you run the linux **head** command on the resulting database dump, it should look something like this:

```
% head wacs-db-dump.mysql
-- MySQL dump 10.13  Distrib 5.1.56, for redhat-linux-gnu (x86_64)
--
-- Host: localhost    Database: wacs
-- ------------------------------------------------------
-- Server version 5.1.56

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
[...]
%
```

Backing Up An Oracle10 Database

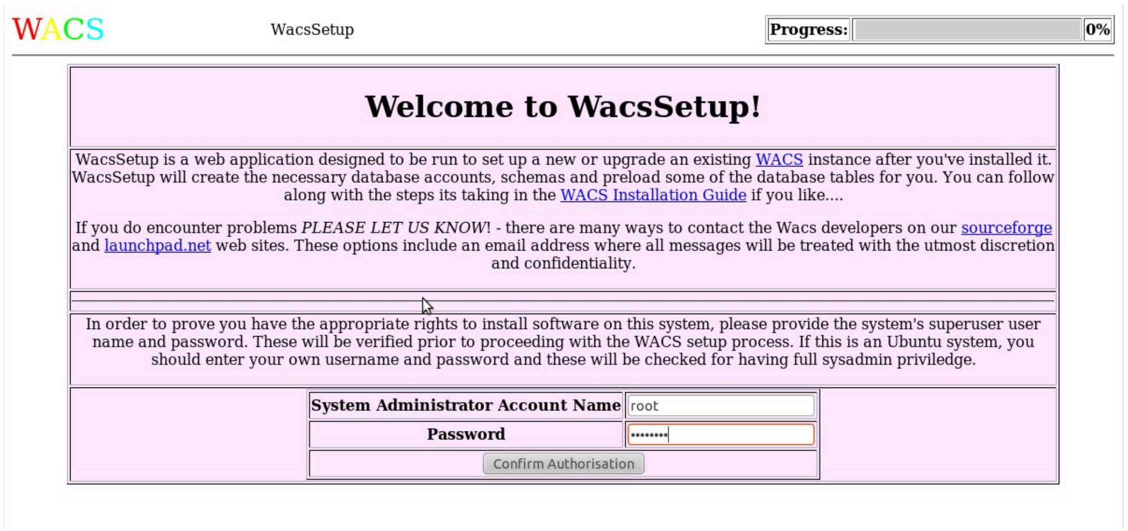
For Oracle 10 databases, it's probably best to use the database export command, **exp**. You invoke **exp** without options and simply follow the prompts and in most cases accept the defaults. Here we've chosen to specify the name of the database dump file as `wacs-dump-31Jul11.dmp` instead of the default of `expdat.dmp`.

```
% exp

Export: Release 11.1.0.6.0 - Production on Sun Jul 31 07:50:46 2011

Copyright (c) 1982, 2007, Oracle.  All rights reserved.


Username: wacs
Password:

Connected to: Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - 64bit Pr
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Enter array fetch buffer size: 4096 >

Export file: expdat.dmp > wacs-dump-31Jul11.dmp

(1)E(ntire database), (2)U(sers), or (3)T(ables): (2)U >

Export grants (yes/no): yes >

Export table data (yes/no): yes >

Compress extents (yes/no): yes >

Export done in US7ASCII character set and AL16UTF16 NCHAR character set
server uses WE8MSWIN1252 character set (possible charset conversion)

About to export specified users ...
User to be exported: (RETURN to quit) > wacs

User to be exported: (RETURN to quit) >

. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user WACS
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user WACS
About to export WACS's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export WACS's tables via Conventional Path ...
. . exporting table                          ASSOC      36065 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                           CONN        429 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                       DOWNLOAD      44141 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                          IDMAP       4457 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                        KEYWORD        171 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                         MODELS       3402 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                   PHOTOGRAPHER         75 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                           SETS      33882 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                            TAG       1026 rows exported
EXP-00091: Exporting questionable statistics.
. . exporting table                         VENDOR         15 rows exported
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.
%
```

With the database now backed up, we can proceed to use **wacssetup** to perform the upgrade. As usual you can invoke **wacssetup** by following the link on either the WACS front page or by typing it directly. On most package WACS installations it's URL will be: http://localhost/cgi-bin/wacs/wacssetup.

Performing The Upgrade

In this section we'll actually go ahead with the database schema upgrade using **wacssetup**. The first screen should be familiar from when we performed the original installation. The need to authenticate our right to make drastic changes to the system remains exactly the same!



WACS WacsSetup Progress: 0%

Welcome to WacsSetup!

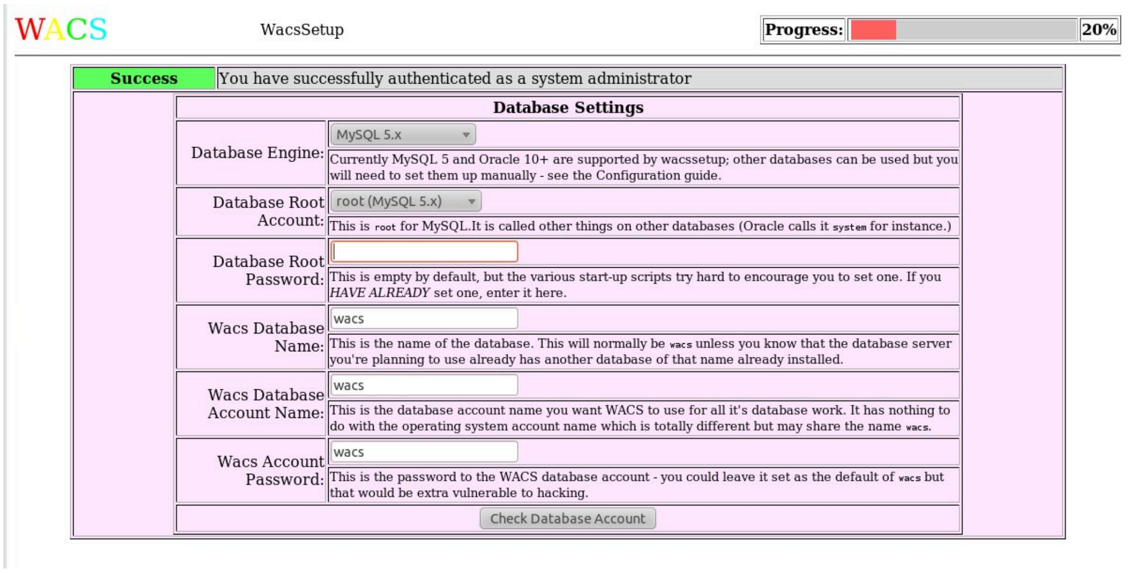
WacsSetup is a web application designed to be run to set up a new or upgrade an existing WACS instance after you've installed it. WacsSetup will create the necessary database accounts, schemas and preload some of the database tables for you. You can follow along with the steps its taking in the WACS Installation Guide if you like....

If you do encounter problems *PLEASE LET US KNOW*! - there are many ways to contact the Wacs developers on our sourceforge and launchpad.net web sites. These options include an email address where all messages will be treated with the utmost discretion and confidentiality.

In order to prove you have the appropriate rights to install software on this system, please provide the system's superuser user name and password. These will be verified prior to proceeding with the WACS setup process. If this is an Ubuntu system, you should enter your own username and password and these will be checked for having full sysadmin priviledge.

System Administrator Account Name	root
Password	••••••••

Once we've authenticated, we now need to specify how we're going to connect to the database. Do remember to enter the database's administrator password at this point if it's been set. The default values are filled in automatically from the `wacs.conf` file so this should be correct in a virtual server environment with multiple wacs instances.



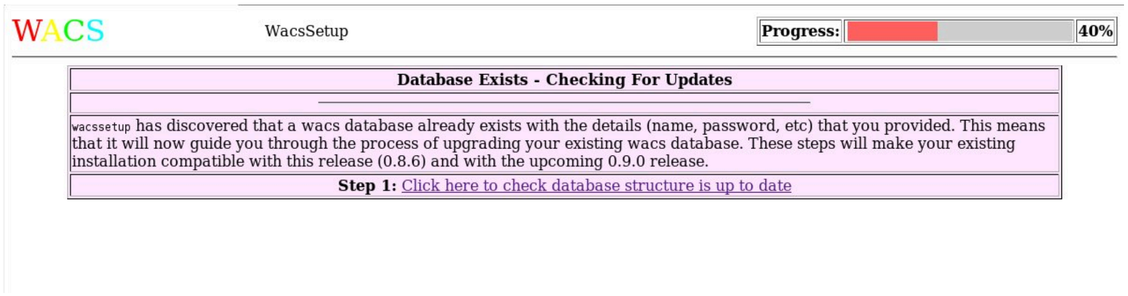
WACS WacsSetup Progress: 20%

Success You have successfully authenticated as a system administrator

Database Settings

Database Engine:	MySQL 5.x Currently MySQL 5 and Oracle 10+ are supported by wacssetup; other databases can be used but you will need to set them up manually - see the Configuration guide.
Database Root Account:	root (MySQL 5.x) This is root for MySQL.It is called other things on other databases (Oracle calls it system for instance.)
Database Root Password:	 This is empty by default, but the various start-up scripts try hard to encourage you to set one. If you *HAVE ALREADY* set one, enter it here.
Wacs Database Name:	wacs This is the name of the database. This will normally be wacs unless you know that the database server you're planning to use already has another database of that name already installed.
Wacs Database Account Name:	wacs This is the database account name you want WACS to use for all it's database work. It has nothing to do with the operating system account name which is totally different but may share the name wacs.
Wacs Account Password:	wacs This is the password to the WACS database account - you could leave it set as the default of wacs but that would be extra vulnerable to hacking.

It's once we've covered the two authentication steps that things suddenly change. As you can see below **wacssetup** has detected an existing installation and is now prompting us with a link to **wacsschema** to check and make the necessary schema changes. **wacsschema** will know that it's being called by **wacssetup** and will change it's responses slightly as a result.

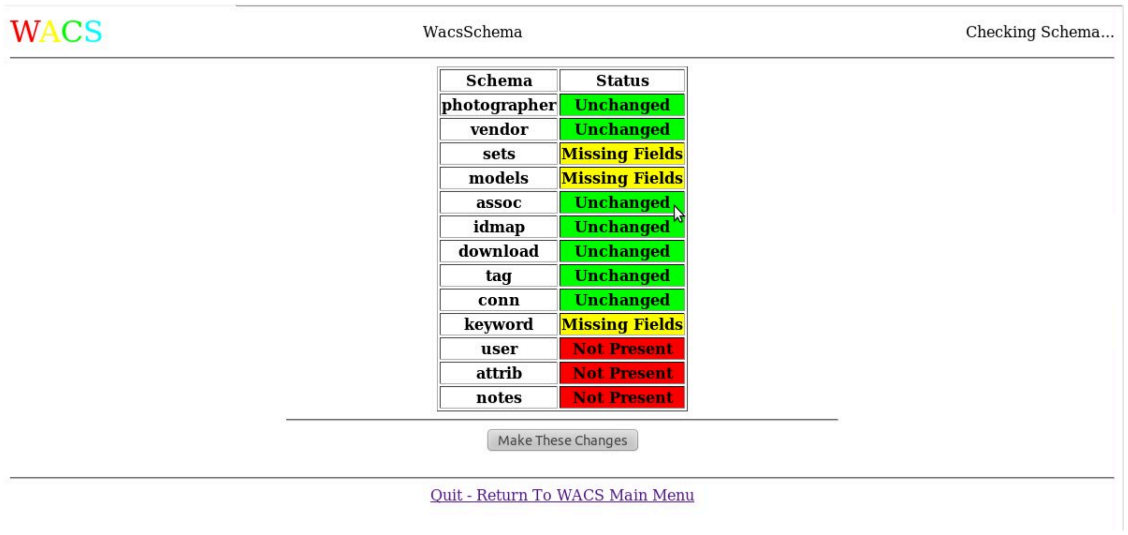


Database Exists - Checking For Updates

wacssetup has discovered that a wacs database already exists with the details (name, password, etc) that you provided. This means that it will now guide you through the process of upgrading your existing wacs database. These steps will make your existing installation compatible with this release (0.8.6) and with the upcoming 0.9.0 release.

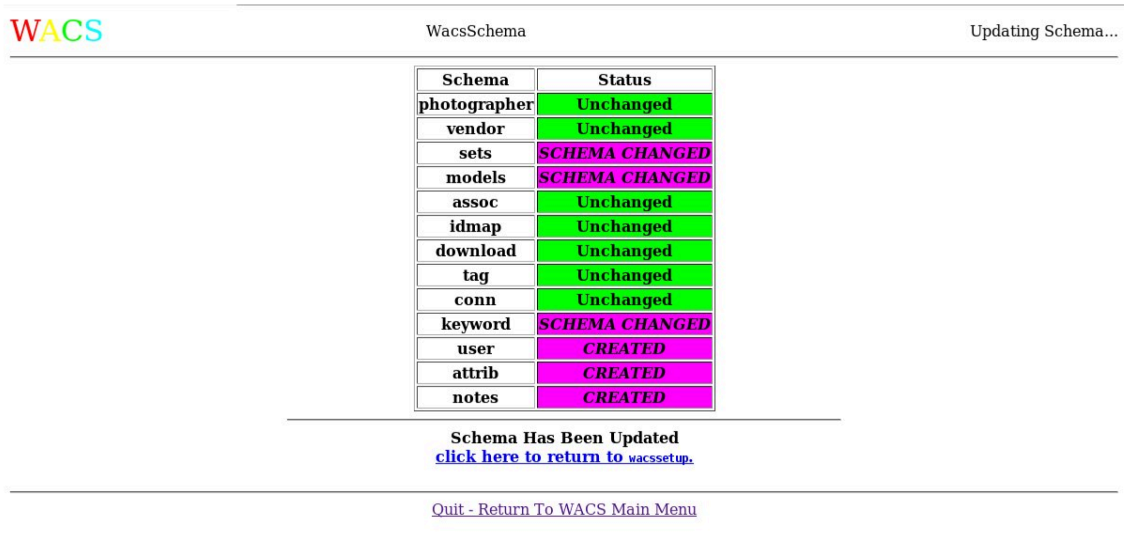
Step 1: Click here to check database structure is up to date

As we saw earlier when discussing **wacsschema** as a free-standing application, it first asseses the situation and then asks us for confirmation that it should take the actions it has determined are necessary.



Schema	Status
photographer	Unchanged
vendor	Unchanged
sets	Missing Fields
models	Missing Fields
assoc	Unchanged
idmap	Unchanged
download	Unchanged
tag	Unchanged
conn	Unchanged
keyword	Missing Fields
user	Not Present
attrib	Not Present
notes	Not Present

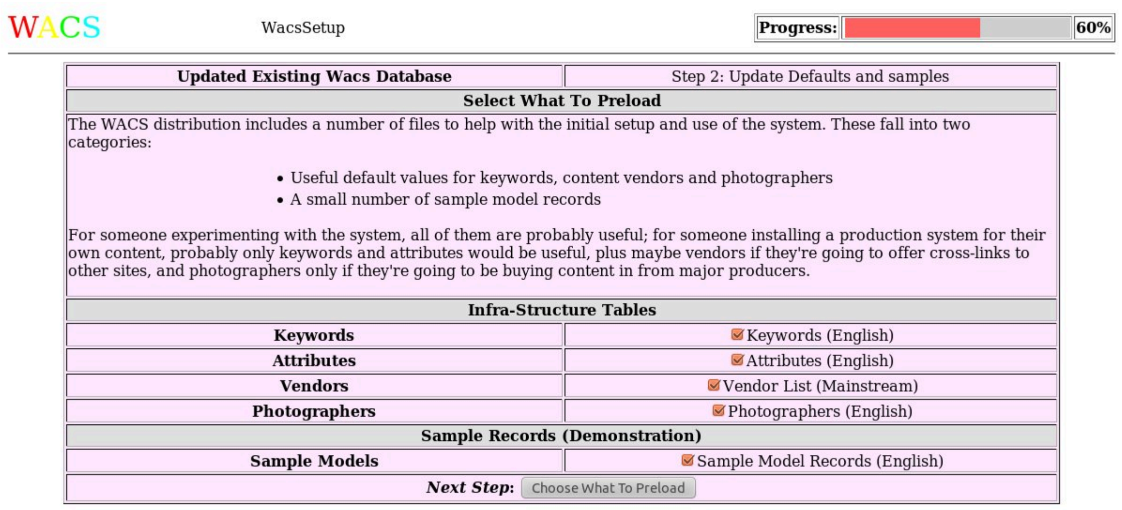
And here is the result when we actually click on the `Make These Changes` button and have wacsschema do it's work.



Schema	Status
photographer	Unchanged
vendor	Unchanged
sets	*SCHEMA CHANGED*
models	*SCHEMA CHANGED*
assoc	Unchanged
idmap	Unchanged
download	Unchanged
tag	Unchanged
conn	Unchanged
keyword	*SCHEMA CHANGED*
user	*CREATED*
attrib	*CREATED*
notes	*CREATED*

Schema Has Been Updated
click here to return to wacssetup.

On completion, it gives us a special link back to **wacssetup** which continues where it left off. In this case it's the preloading of database values. As with the original setup run, we can choose which files we want to load.



Updated Existing Wacs Database	Step 2: Update Defaults and samples
Select What To Preload	

The WACS distribution includes a number of files to help with the initial setup and use of the system. These fall into two categories:

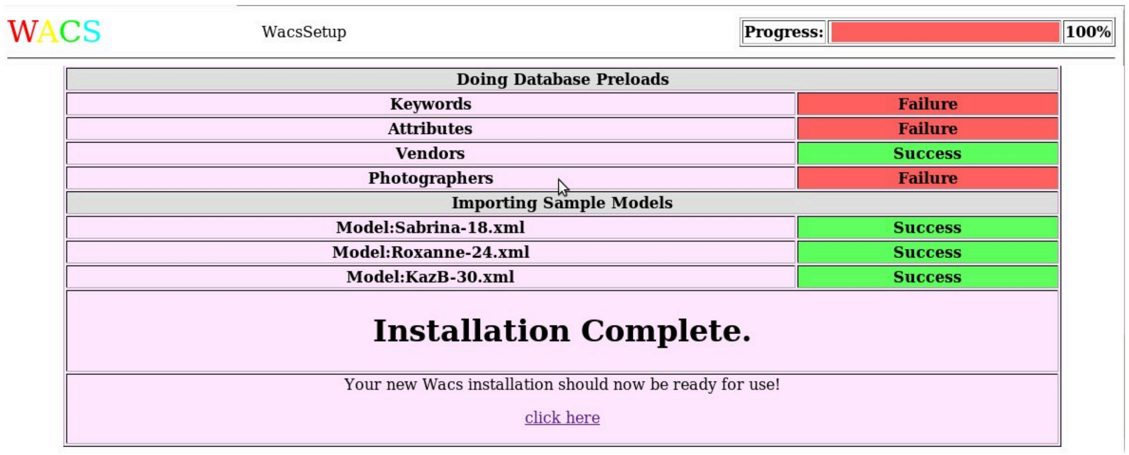
- Useful default values for keywords, content vendors and photographers
- A small number of sample model records

For someone experimenting with the system, all of them are probably useful; for someone installing a production system for their own content, probably only keywords and attributes would be useful, plus maybe vendors if they're going to offer cross-links to other sites, and photographers only if they're going to be buying content in from major producers.

Infra-Structure Tables	
Keywords	✓ Keywords (English)
Attributes	✓ Attributes (English)
Vendors	✓ Vendor List (Mainstream)
Photographers	✓ Photographers (English)
Sample Records (Demonstration)	
Sample Models	✓ Sample Model Records (English)

Next Step: Choose What To Preload

And now we reach the end of the process. Note that some of the database preloads have failed in our example. This should not actually pose a problem and merely indicates that certain records already existed and could not be overwritten. So long as the WACS system is working, this isn't an issue.



Doing Database Preloads	
Keywords	**Failure**
Attributes	**Failure**
Vendors	**Success**
Photographers	**Failure**
Importing Sample Models	
Model:Sabrina-18.xml	**Success**
Model:Roxanne-24.xml	**Success**
Model:KazB-30.xml	**Success**

Installation Complete.

Your new Wacs installation should now be ready for use!

click here

And so we have now upgraded the database schema to the new version and can look forward to making use of the exciting new features coming in Wacs 1.0.x.

Chapter 11. Moving XML Files Between Releases and Installations

Introduction To Moving XML

The WACS system provides a number of ways of moving all kinds of types of data between installations using files created in eXtensible Markup Language - XML for short. This includes both actual data that forms part of the working data set of a server and configuration data for the server. In general the process is fairly simple, but there are always issues between different versions of WACS as things do change between releases and over time. In this chapter, we will discuss some of the issues related to how to move data between either different servers or different versions of the WACS programs. The steps and concepts themselves are pretty much the same.

XML Default Values Files

The WACS distribution currently comes with four XML files containing default values for some of the main database tables. These are:

Table 11.1. XML Defaults files in WACS

Tablename	Defaults File	Purpose
attrib	**attrib.xml**	These are the various attributes that can be assigned to a set or model - introduced in Wacs 0.8.5
vendor	**vendors.xml**	Some sample vendor details for major internet sites
photographer	**photographers.xml**	Some sample photographer details taken from major internet sites
keyword	**keywords.xml**	A good working set of English language keywords with scores and attributes

Porting Older XML Files

In WACS release 0.8.6 the code that previously read the defaults XML files for each database table was merged into a new single data population utility called **wacspop**. This replaced the previous separate populator commands: **vendpop**, **keywordpop** and **photpop** and obviated the need for the creation of a new command for handling the new attributes schema. With this move, it became necessary for each XML file to better identify what it actually was. It was therefore decided to add three extra attributes into each file, very similar to those already used by the data exchange XML files created by **wacsexport**, **wacsxmlout** and **wacsselout**.

Three new attributes were added to the database population XML files:

Table 11.2. Required XML Entities

XML Entity	Value
version	normally `WACS_POPULATE_VERSION_2`
coreschema	The appropriate schema name: **attrib**, **vendor**, **photographer** or **keyword**.
revisiondate	The date this revision was created in DD-MON-YYYY format

If you wish to convert an older file to the new format so that it can be imported by **wacspop**, you merely need to add these three values near the top (typically third line) of your XML file using a normal text editor.

```
[...]
<version>WACS_POPULATE_VERSION_2</version>
<coreschema>vendor</coreschema>
<revisiondate>29-MAY-2011<revisiondate>
[...]
```

Once this has been done, **wacspop** should import your XML file taken from a previous release of Wacs. Do be aware however that moving an older file to a newer release will not populate the new fields added for Wacs 1.0.0 and these fields will remain empty. It is an intended future upgrade for **wacspop** that it should be able to update records as well as just add any that are not currently there.

Chapter 12. Troubleshooting

Introduction

We obviously hope the installation of Wacs will go smoothly but it is a pretty complex system that depends on a significant number of other packages working together with our code. This chapter aims to provide help and assistance if all doesn't go according to plan. There are two main sections - the first (the section called “Installation Troubleshooting”) covers what to do if the installation proceedure itself goes wrong; the second (the section called “General Troubleshooting Tips”) covers more general issues where the system installs but then doesn't work or where it stops working for some reason.

Installation Troubleshooting

What to do if the installation goes wrong...

Tip

DON'T PANIC!

The first thing to do is to try and remember where it started to fail - the **Back** key on your web browser should be able to help here if you were using **wacssetup**. The following steps should help you find out what worked and what didn't...

Checking Database And Accounts

MySQL5

Open up a terminal window on the server and try the following (but using whatever username/password you actually gave **wacssetup**):

```
% mysql -u wacs -p wacs
```

and enter your password at the prompt. If it replies with:

```
ERROR 1044 (42000): Access denied for user 'wacs'@'localhost' to database 'wacs'
```

it is fairly safe to assume that the user account and database did not in fact get created.

Oracle 10 and Oracle 11

Open up a terminal window on the server and try the following (but using whatever username/password you actually gave **wacssetup**):

```
% sqlplus
SQL*Plus: Release 11.1.0.6.0 - Production on Wed Nov 18 14:58:32 2009

Copyright (c) 1982, 2007, Oracle.  All rights reserved.
```

```
Enter user-name: wacs
Enter password:
ERROR:
ORA-01017: invalid username/password; logon denied
```

it is fairly safe to assume that the user account and database did not in fact get created.

Checking Structures

If the first group of actions that **wacssetup** takes seem to have gone OK, and then the next step where it creates the schema starts to show problems, then we need to check if anything got created. This particular step is where problems with the server IP address configuration often show themselves, particularly when using MySQL as it is the first time that we try to use our Wacs system credentials in earnest. Please see the section called “Preparation Tasks” for more information on this. A sure sign of this failure will be an error message something like this:

```
DBI connect('wacs:www.example.com','wacs',...) failed: Access denied for
 user 'wacs'@'myserver.example.com' (using password: YES) at wacssetup
 vendpop line xxx
Can't connect to database
Reason given was Access denied for user 'wacs'@'myserver.example.com'
 (using password: YES)
```

Verifying Schema

Anyway, if you're seeing this kind of message, you need to verify if the database schemas did indeed get created or not. To do this, we log into SQL with the Wacs user account details as follows (change account names and passwords as appropriate):

```
% mysql -u wacs -p wacs
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 316
Server version: 5.0.84 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current
 input statement.

mysql> describe conn;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| centryno  | int(9)       | NO   | PRI | NULL    |       |
| cgroup    | int(6)       | YES  |     | NULL    |       |
| corder    | int(3)       | YES  |     | NULL    |       |
| cflag     | char(1)      | YES  |     | NULL    |       |
| cstatus   | char(1)      | YES  |     | NULL    |       |
| cmodelno  | int(6)       | YES  |     | NULL    |       |
| csetno    | int(9)       | YES  |     | NULL    |       |
| cphotog   | varchar(6)   | YES  |     | NULL    |       |
| ctype     | varchar(20)  | NO   |     | NULL    |       |
| cdesc     | varchar(80)  | YES  |     | NULL    |       |
| ccomments | varchar(240) | YES  |     | NULL    |       |
| cpath     | varchar(160) | YES  |     | NULL    |       |
| cadded    | date         | YES  |     | NULL    |       |
| camended  | date         | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
14 rows in set (0.00 sec)

mysql> quit
%
```

Note

The SQL commands used, `describe` and `quit` are identical for Oracle although the formatting and output will differ slightly.

If you instead get a message like: `ERROR 1146 (42S02): Table 'wacs.conn' doesn't exist` then that is confirmation that the schema create phase had failed. Once you've resolved the cause (most likely these hostname issues), you have two choices:

- remove the existing half built structures and run **wacssetup** again (see below)
- complete the database creation manually picking up the manual install instructions from the appropriate point.

Cleaning Out Half-Created User Accounts

If you choose to do the former option above (restart), you will need to remove the existing user account and database area as follows:

MySQL version - replace `wacs` and `myserver.example.com` as appropriate. The second drop user step may fail - that's not a problem. Be very careful indeed you have the right database name for step four!

```
% mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is xxx
Server version: 5.0.84 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> drop user 'wacs'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> drop user 'wacs'@'myserver';
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> drop user 'wacs'@'myserver.example.com';
Query OK, 0 rows affected (0.01 sec)

mysql> drop database wacs;
Query OK, 10 rows affected (0.01 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye
%
```

You should then be able to re-run the **wacssetup** web application (once of course you've worked on the root cause of the failure - most likely the IP address/DNS name issues).

Beyond That

Once you've got beyond the user account, database area and database schema creation steps, there is really only the initial data population left to be done. For this it's simplest to just follow the last steps of the the section called “Manual Database Creation Steps” process from the step the section called “3. Create default database contents (optional)”.

wacssetup completes but you can't login

If the **wacssetup** completes normally but you fail to to login when you're sure you're using the right username and password, there's a couple of things to try. run `/usr/bin/pam_auth` (packaged installations) or `/usr/local/bin/pam_auth` (easyinstall/manual installations) - give it your username and password, separated by a space, and then press <ENTER>. If it replies with `OK` all is well; if it replies with `ERR` or some other error message (and you know you typed your password right of course) then there's a problem.

Most likely it's in `/etc/pam.d` and to do with the file called `wacs` there. Compare it to other files in the same directory like `sshd` and `squid` (if present) for the correct dialect for this platform. The security sub-directory of the Wacs source tarball contains various versions written over the years, one of which may be in the correct PAM dialect.

If All Else Fails

The next step is probably to follow the manual creation instructions at this point and see how they progress. If those fail, please visit our web sites - sourceforge site [http://wacsip.sourceforge.net/] or launchpad site [https://launchpad.net/wacs/] or use our mailing lists or email address to get help.

Tip

Do take a look through the checklist below the section called “General Troubleshooting Tips” as there may be tips there that help and at the very least having gone through these will help us get to the root cause of your problem quicker.

General Troubleshooting Tips

Obviously we hope the installation script will create a running installation for you, but there will no doubt be occasions when it does not. Before seeking help via the mailing lists and other resources on the sourceforge site [http://wacsip.sourceforge.net/], there are some things you should clarify. The first of these is to confirm what the status of the various subsystems are. Here's a quick check list:

Troubleshooting Checklist

- Is there an error being reported?

  1. If you get the error message `Can't find lsb_release in order to determine distribution` and you are on an older Fedora Core or CentOS version, try running: **yum install redhat-lsb** and then running the installer again. Upgrades often leave out this package although it should be part of the standard operating system.

  2. If you get the error message `DBI connect('wacs:myserver.myisp.com','wacs',...) failed: Can't connect to MySQL server on 'myserver.myisp.com' (110) at ./vendpop line 39` check that you can ping the hostname of your server locally. Often people don't have things set up so that a machine with an "internet name" can see itself by the same name on the local network. The above error is a symptom of this problem.

  3. Check the apache web server log file in `/var/log/httpd/`*`myserver`*`-errorlog`

  4. Check the system messages in `/var/log/messages`

  5. Check the output from the kernel by running **dmesg** (a common cause of trouble is the SELinux security mechanism, an avc_denied message in the output of dmesg is a solid clue to this - see comments below on SELinux)

- Is the web server running?

  1. If you point a web browser at the top level URL of your server, do you get a web page back, be it a distribution-supplied test page or previous pages you placed there?

  2. Can you get the wacs main page? (http://*myserver*/wacs/)

  3. Can you get any response from the WACS cgi-bin programs, even so much as a coloured background to a blank screen? (http://*myserver*/cgi-bin/wacsnewmodels)

  4. Is there an httpd process running? (**ps wax | grep httpd**)

  5. Is the HTTPD set to start automatically? (**systemctl enable httpd**)

- Is the database server running?

  1. See if you can connect using the SQL command line application - called mysql for MySQL, sqlplus for Oracle 10g.

  2. Check for the database processes running - mysqld for MySQL, a whole cluster of oracle* and ora_* processes for Oracle 10g.

- Can you login in to Wacs?

1. Check you can actually log into the server with that username and password!
2. run `/usr/bin/pam_auth` (packaged installations) or `/usr/local/bin/pam_auth` (easyinstall/manual installations) - give it your username and password, separated by a space, and then press <ENTER>. If it replies with `OK` all is well; if it replies with `ERR` or some other error message (and you know you typed your password right of course) then there's a problem. Please see the section called "**wacssetup** completes but you can't login" for more information on this.

Special Notes About SELinux

SELinux is an enhancement to Linux that allows potentially vulnerable services (like an internet-exposed web server) to be operated on a basis where each action the program tries to take needs to be explicitly allowed, rather than the normal allowed unless denied approach of most Unix environments. As such SELinux presents a whole new group of challenges for getting WACS to work, because we have to extend the ruleset as to what is allowed and what is not. It can be done, but it will take work and some experimentation. Whereever we have not used the Operating System supplied packages (Web Server, Database, etc), we're going to have to add those rules. The first thing to check is whether SELinux is enabled - to do this, type:

```
% sestatus
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   permissive
Mode from config file:          permissive
Policy version:                 20
Policy from config file:        targeted
%
```

If it's either disabled, or is enabled but with a current mode of permissive, it's not actualling going to be causing us a problem right now. If it is enabled and enforcing, we've got to work on it. The web server process needs a security context of `httpd_sys_content_t` to be present on any directory it needs to access, so the first step is to add this context to each directory (outside of the normal ones) that it is likely to access. This is done with the **chcon** commands shown above in the manual install chapter - example:

```
# chcon system_u:object_r:httpd_sys_content_t /var/run/wacs
#
```

In addition to the directory gaining the `httpd_sys_content_t` security context, any pre-existing files will also need the same, so this can be done with:

```
# chcon -R system_u:object_r:httpd_sys_content_t /var/run/wacs/*
#
```

To inspect the security context of a file or directory, you use the -Z option to the ls command:

```
# ls -Z /var/run/wacs/
-rw-r--r--  apache apache system_u:object_r:httpd_var_run_t leases.acl
#
```

While the easyinstall script does try to set these for all the areas the web server might go (`/var/run/wacs`, `/etc/wacs.d` and the files area `/home/wacs/*`), any problems which are causing avc_denied messages in the dmesg output are most likely down to this issue.

Chapter 13. Using Other Databases

Position Overview

WACS currently works with four of the leading database backends - MySQL (version 5.x), MariaDB (version 10.x), Oracle (versions 10g, 11i and 12) and PostgreSQL (9.5,10.x). For MariaDB, just install as if it were MySQL. That said, the usage of SQL by the WACS code is generally kept as generic as possible with, outside of the initial table creation steps, no code differences between what is used when talking to MySQL and when talking to Oracle. This should make the code sufficiently generic to work against just about any database that supports a full featured SQL interface via the DBI module for perl and the pear DB and/or PDO module for Php. Supporting a new database platform should be a simple matter of ensuring the appropriate DBI and pearDB modules for that database are available on the system and that the appropriate date types have been added to the conversion functions provided in the core WACS module. We have already started to look at adding support for Microsoft SQL Server and Azure SQL in a future release.

The basic WACS package applications are designed to work with MySQL 5 and in fact in order to try to make WACS work “Out Of The Box”, they include dependencies on the MySQL 5 server code to ensure the necessary support is added. If you want to use PostgreSQL instead, look for a separate wacs-for-psql package instead of the main wacs package - it will pull in all the same other wacs packages but with PostgreSQL dependencies instead of MySQL. This does not mean that the packaged versions *have* to be used with MySQL only however; you can easily install an Oracle server as well and merely change a few entries in the `wacs.cfg` file and have it work just fine.

Oracle 10, 11, 12, 18 and 19

Since Oracle is a proprietary package, most Linux distributions do not include the binary libraries needed to connect to it. Thus while you can use the package installer to install the necessary database agnostic generic drivers, you have to do additional work to get the actual database drivers to work. In this section, we will outline the steps needed to get the necessary parts installed for Oracle to work from both Perl and PHP5.

Oracle Perl Driver

The first step to getting WACS to work with Oracle is to download a suitable set of client side applications and libraries - these are typically provided by *Oracle Instant Client* or a package of a similar name. Of course this does not include the database itself but when using the likes of Oracle, it would be quite normal (but not required) to have separate web servers and database engines. Wacs works just fine on multiple front end web servers with a single Oracle backend. Similarly collection administration and straightforward use can be performed on a separate workstation from hosting the database.

Generally doing a full install of instant client will provide all the libraries you need. If you are downloading these from the Oracle OTN web site, you will need the following packages either as zip files or rpms if that is the package manager on the distribution you are using.

Table 13.1. Required Client Side Oracle Packages

Component	Package Name
Basic Libraries	**oracle-instantclient19.6-basic-19.6.0.0.0-1.x86_64.rpm**

Component	Package Name
Developement (SDK)	**oracle-instantclient19.6-devel-19.6.0.0.0-1.x86_64.rpm**
ODBC (Useful, not reqd)	**oracle-instantclient19.6-odbc-19.6.0.0.0-1.x86_64.rpm**
SqlPlus (Command Line)	**oracle-instantclient19.6-sqlplus-19.6.0.0.0-1.x86_64.rpm**

The next step is to establish the necessary `ORACLE_HOME` environment variables and check that the Oracle stack is running by connecting to the server using the **sqlplus** client (sometimes called **sqlplus64** on the older x86_64 versions) provided as part of the instant client package. If this reports that the necessary shared libraries are not found, this may be because the dynamic library caching configuration files have not been created as needed. For RedHat based distros like Fedora, this is achieved by creating a file in `/etc/ld.so.conf.d/` called something like **oracle-19-installclient.conf** which simply contains a single line pointing to where the files like `libsqlplus.so` are to be found. For the x86_64 version of instantclient installed from the RPMs, this should be `/usr/lib/oracle/19.6/client64/lib` assuming the RPM was installed into it's default locations. Once this file has been created, you just need to run **ldconfig** as root to rebuild the `/etc/ld.so.cache` to include the files in this path.

If you get the message `sqlplus64: error while loading shared libraries: libnnz11.so: cannot enable executable stack as shared object requires: Permission denied` that means that SELinux is still active on your host and has barred access to the Oracle libraries. The quick fix is to disable SELinux by running **setenforce permissive** as root on your machine and editing the file `/etc/selinux/config` to set the mode to permissive on subsequent system boots.

Once you've actually got the sqlplus program to run, that is unfortunately not usually the end of the story as it has to be configured to find the database server, etc. There are two parts to this task - firstly to create a suitable `tnsnames.ora` file to describe how to contact your server - place this into a newly-made `/etc` directory within the oracle instant client directory. Secondly you need to set environment variables that tell instant client what to look up within that tnsnames.ora file. This is done by establishing four environment variables, ideally globally for the system - these are:

Table 13.2. Oracle Environment Variables

Variable	Typical Value
ORACLE_HOME	/usr/lib/oracle/19.6/client64
ORACLE_SID	yourdbname
TWO_TASK	yourdbname
TNS_ADMIN	/usr/lib/oracle/19.6/client64/etc

In many ways the best way to establish these environment variables is to create the appropriate files in the global profiles directory, which is to be found in `/etc/profile.d`. Typically it's best to create two files called something like **oracle11g-instantclient.sh** with commands for sh/bash shells, and **oracle19-instantclient.csh** with commands for csh/tcsh shells. The shell (.sh) file should look something like this:

```
export ORACLE_HOME=/usr/lib/oracle/19.6/client64
export ORACLE_SID=yourdbname
export TWO_TASK=yourdbname
export TNS_ADMIN=/usr/lib/oracle/19.6/client64/etc
```

You may also wish to alias sqlplus64 to sqlplus if you're using one of the old versions of the instant client that used this name.

It should then be a fairly simple matter of using **cpan DBD::Oracle** to download, compile and install the necessary database driver for Perl DBI. Once that's done, you just need to follow the installation instructions for Oracle in the appropriate chapters of this guide.

Oracle Php5 Driver

The conventional install of Php5's pear DB routines actually does include the first line of support for Oracle in the form of the **oci8.php** file in `/usr/share/pear/DB` but this itself needs **oci8.so** which normally lives in `/usr/lib64/php/modules`. To make this module, you will need the full set of SQL*Net libraries as provided by either a full database installation or Oracle's **instant client** product. You will also need the C compiler and the **php-devel** package (this contains the command **phpize** so if trying to invoke **phpize** gives `Command not found` you almost certainly don't have the necessary development package installed.

Since WACS currently uses the now-obsolete **Pear DB** module for database access, it appears to be a bit of a challenge to find the correct source code. We do also support using the **PDO** interface as from Wacs 1.0.0. We have not yet re-written the WACS sample programs to use PDO but may well do in due course.

While we were preparing this document (March 2011, revised May 2018 and June 2020), the source code for the OCI8 driver could be found at the PECL web site [http://pecl.php.net/get/oci8]. If you down load this with something like wget and save it locally, running the following (assuming you have ORACLE_HOME set correctly) as root produced a successful install: `pecl install oci8`. You then need to enable the extension in php itself; in some distros this is done by simply adding **extension=oci8** into the `/etc/php.ini` file, while in more recent distros it is now convention to create a file called **oci8.ini** in the directory `/etc/php.d` containing this line. Once this configuration change has been made, you will need to restart your web server to make the new settings live. All that then remains to be done is to configure the phpdbconnect string in the main wacs configuration file with the correct database specification and things should start to work.

Building support for the PDO - PHP Data Objects - module is rather more complex unfortunately. The good news is that it is a fully integrated part of PHP itself and it's source code is part of the main PHP sources. The bad news is that many Linux distributions turn it off when they are building php (as being proprietory code); others are at least building the hooks and putting it in a package. If you can't find a suitable package, you may well be able to build one quite easily.

Here are the steps to take to do this - you may need to install the necessary packages for building packages such as **php-devel** (Fedora, RedHat, CentOS) or **php7.3-dev** (Debian, Ubuntu).:

```
RPM-based distros:
% cd rpmbuild/SRPMS
% dnf download --source php
% rpm -iv php-7.4.6-1.fc32.src.rpm
% cd ../SOURCES
% tar -xJf php-7.4.6.tar.xz
% cd php-7.4.6/ext/pdo_oci
% mkdir -p ~/src/php/pdo_oci
% cp -r * ~/src/php/pdo_oci
% cd ~/src/php/pdo_oci
% phpize
% ./configure --with-pdo-oci=shared,instantclient,\
/usr/lib/oracle/1.9.6/client64/lib,19.6.0.0.0
```

```
% make
% sudo make install
%
DEB-based distros:
% cd debuild
% aptitude source php7.3
% cd php7.3-7.3.11/ext/pdo_oci
% mkdir -p ~/src/php/pdo_oci
% cp -r * ~/src/php/pdo_oci
% cd ~/src/php/pdo_oci
% phpize
% ./configure --with-pdo-oci=shared,instantclient,\
/usr/lib/oracle/1.9.6/client64/lib,19.6.0.0.0
% make
% sudo make install
%
```

This alternative solution is essentially to download the source packages for PHP for your distribution and rebuild the piece of them that we need by enabling the `--with-pdo-oci` flag. If you are using the Oracle Instant Client distribution, you specify this with `--with-pdo-oci=instantclient,/usr,19.6.0.0.0` as an option to the **configure** script where the `/usr` is where it is installed and `19.6.0.0.0` is the exact Oracle DB version number. Do remember that the php version will be updated from time to time and you may need to update the driver to match the current version.

Oracle PHP7 driver

Note

For PHP7, you will need a minimum of oci8-2.1.8 which should be available from pecl as before. See the sectiona above for details of how to install it.

Some distributions have now moved over to using **php-fpm** to provide PHP functionality from within the Web Server. Generally this is a good idea as it stops the web server having to start up a new PHP interpreter each time. Unfortunately it stops the method we've used previously and in other languages to set the essential `ORACLE_HOME` environment variable needed for the oci8 and PDO oci drivers to work. These instead now have to be set in `/etc/php-fpm.d/www.conf` where you have to add a line of the form `env[ORACLE_HOME] = /usr/lib/oracle/19.6/client64` to it. Of course you need to change the path to the actual install location of your Oracle database software. Once added you need to do a `systemctl restart php-fpm.service` to make your changes active.

PostgreSQL Version 9.5, 10.x and 12

We are in the process of adding support for PostgreSQL to WACS and hope it will be part of the Wacs 1.0.0 release. Please check the release notes for the up-to-date position of this support, but we are hoping it will prove fully functional. Some of the customisation available with other databases may not be available.

Note

We have created a **wacs-for-psql** package for both Fedora and Ubuntu that attempts do to what is necessary to create a working Wacs installation. To this end, we create the basic database and a special *`wacsdba`* database administrator account while in the package's own post-install script. As this is run as root on the machine, it is capable of doing things in the way necessary for those steps in the PostgreSQL way of working.

PostgreSQL Installation Basics

PostgreSQL is rather different from Oracle and MySQL as it tends towards using the host operating system authentication mechanisms for identifying users rather than a more conventional username and password combination. It can be reconfigured to allow username and password login, and we attempt to do this in as minimalist a way as possible to preserve normal operation for other users of the database. By default, the system manager account, `postgres` gains automatic database manager authority when it invokes any of the postgres tools (like the SQL command line, **psql**).

Warning

NB: some Linux distributions, eg Ubuntu, support simultaneous installation of multiple versions of Postgresql. In this case, you may find that the TCP/IP port number on which the database listens has changed from the default of 5432. In one case on an Ubuntu Xenial system which had been upgraded from previous releases; PostgreSQL 9.1 had port 5432, PostgreSQL 9.3 had port 5433 and PostgreSQL 9.5 had 5434. All of these are configured in `/etc/postgres/<release_no>/main/postgresql.conf` so do check the **port =** directive to make sure you are connecting to the version of the database you think you are.

A key point to understand is that the Perl DBD driver and similar interfaces to PostgreSQL use the network domain rather than the named pipe for communication and this is not by default enabled by the installation packages for PostgreSQL. In the next section, we will take you through the setup and configuration steps needed for PostgreSQL to interact with the network properly.

Enabling Network Operation In PostgreSQL

There are three steps you need to take to enable network-based operation in postgres; to do these as described belowe you will need access to either the postgres or root accounts for the server computer. If you do not have access to this, there may be other ways to do what is needed via cPanel or similar but you will have to consult a suitable administrator for the system you are using.

Getting PostgreSQL to Listen

The first thing we need to do is to get the **PostgreSQL** server to listen for connections on the network port. This is done by editing the `postgresql.conf` file which can be found in *`/etc/postgresql/10/main`* for PostgreSQL 10 or in *`/etc/postgresql/9.5/main`* PostgreSQL 9.5.

To use WACS effectively the database is all managed under a dedicated WACS user account and it is thus easier to provide username and password without the normal host-based user authentication taking place. If you wish to use the interactive PostgreSQL command line interpreter, **psql** for fixing up issues or making ad hoc database queries, it is best to use the `-h` option with the host name of the database service. You will also need the `-U` option to specify the username to use, and maybe `-W` to force interactive prompts for the password (optional). A sample invocation of the **psql** SQL command line would therefore be:

```
psql -d wacs -h myserver -U wacs
```

Chapter 14. Installing With EasyInstall

WARNING: easyinstall is depricated

Warning

The **easyinstall** installation method is now depricated and will be removed in a future release of WACS. We do not recommend using it for this release (1.0.0) and are not testing it extensively. The documentation is left here for now to reflect the fact that the code is still supplied and people may have customisations for it that they wish to move from a previous release. New WACS users are strongly discouraged from using it. If you do make use of it to install on a platform or distribution we do not currently support, we would appreciate being sent any changes you make to it.

Easyinstall: Download

WACS is obtainable from sourceforge where it is known as WACSip because of a name clash with a different package. The sourceforge site contains the latest code, documentation, news articles, mailing list details and even some screenshots. The URL is http://wacsip.sourceforge.net [http://wacsip.sourceforge.net/]. You can obtain WACS either by downloading one of the official releases or by pulling the very latest "bleeding edge" version from the Subversion repository. We do not use the newer git version control system as yet; WACS is now well over 10 years old and the project was started under Subversion (svn). The official releases are also mirrored at http://launchpad.net/wacs [http://launchpad.net/wacs/].

To get the latest official release, go to the sourceforge project page [http://sourceforge.net/projects/wacsip/] and click on download. If the latest release is 1.0.0, then download wacs-1.0.0.tar.gz. Once downloaded, save it somewhere appropriate for unpacking - the installation will work without the archive, although there are sample configs and docuentation files which are not installed anywhere. If for instance you've decided to keep wacs in an "src/wacs" directory of your home directory and your web browser has placed the downloaded file on your Desktop, do:

```
% cd
% mkdir -p src/wacs
% cd src/wacs
% tar -xzvf ~/Desktop/wacs-1.0.0.tar.gz
x wacs-0.8.1/README
[...]
% cd wacs-1.0.0
```

Alternatively to get the very latest version from the sourceforge subversion repository, do the following:

```
% cd
% mkdir -p src/wacs
% cd src
% svn co https://wacsip.svn.sourceforge.net/svnroot/wacsip/trunk wacs
[...]
% cd wacs
```

Easyinstall: Running

EasyInstall should be just that, but probably won't be. An installer is a complex thing and depends heavily on the environment around it. If you're running Fedora Core 6, Fedora 7, Fedora 8, Fedora 9 or Ubuntu 7.04 (Feisty Fawn) or 8.04 LTS and accept all the defaults, there is a reasonable chance it'll work. If you're running an older release with either MySQL < 5.0 or Apache < 2.2, you may well have problems... On other RPM-based distributions with the yum updater (SuSE, CENTOS, RHEL) you're in with a chance of it working, or at the very least installing most of the necessary infrastructure for you. Other non-RPM based distributions will almost certainly fail (apart from Ubuntu which should work), but if you feed back the error messages, we'll have a go at fixing it. If using an apt-get based distribution, modifying the places where it checks for "Ubuntu" to whatever your distribution returns when you do an **lsb_release -sir** may well help.

To run easyinstall, become the super user (root) and issue the following commands:

```
# cd unpack_location/install
# ./easyinstall
```

and follow the onscreen prompts. Packages invoked by this script will include your package manager (yum, apt-get, etc) and the perl CPAN installer. At the end of the package configuration questions, you will be shown your answers and asked for comfirmation; if you answer n for no, you'll be asked the questions again. After that, once the installation phase starts, if you make a mistake in answering a question press <CTRL>-C to abort and start again. Between all the package managers and installers, you may well have to answer a couple of dozen questions in all.

The final system configuration step before starting using WACS is only applicable if you're using a version of Linux which includes the security hardening extension, SELinux. This currently is limited to the Red Hat based distributions like Fedora, Red Hat Enterprise Linux and CENTos. Rumour has it OpenSuSE will shortly be including SELinux as an option. Unfortunately this release of WACS is not compatible with SELinux and so it'll have to be configured so as to ensure that SELinux is running in a reduced mode that will not block the WACS components from working. We hope to have this resolved by the next release of WACS. You can determine the current mode of SELinux using the sestatus command:

```
% /usr/sbin/sestatus
SELinux status:                 disabled
%
```

To change the normal operational mode, you need to edit the file called `/etc/sysconfig/selinux` and change the line which reads `SELINUX=enabled` to either `SELINUX=permissive` (generates big log files and slows machine but allows for SELinux to be turned back on later more easily) or `SELINUX=disabled` (which disables it completely but can cause problems in the future if you want to switch it back on). You will also probably want to disable it immediately rather than doing a reboot before you can continue working on WACS - to do this, become root and run the following:

```
# /usr/sbin/setenforce 0
setenforce: SELinux is disabled
#
```

You can check this change has taken effect by using the **sestatus** command again.

At that point the installation should be complete and you'll need to look at the getting started document for how to set up a WACS collection.

Index