Wacs Manpage Collection

Fifth Edition

for WACS 0.9.2

B "Beaky" King

Publication date Monday 9th May 2016

Wacs Manpage Collection

by B "Beaky" King

for WACS 0.9.2

Publication date Monday 9th May 2016

Abstract

WACS is a Web-based Adult Content Server environment built using the LAMP (Linux, Apache, MySQL and Perl/Php) infrastructure. It provides the database, toolchest and API infrastructure to build a sophisticated indexed archive of adult material for either personal or commercial use. It is a Free Software/Open Source package released under the GNU Public License Version 3 (GPLv3) and supports use of Oracle in addition to MySQL.

This document contains a collection of the command line tools provided with WACS and the various functions they perform.

The WACS source code and other documentation and support tools can all be found at the WACS website at Sourceforge [http://wacsip.sourceforge.net/]. Releases are also uploaded to the Launchpad [http://launchpad.net/wacs/] site for additional resilience. There is a demonstration web site for WACS at PinkMetallic.com [http://www.pinkmetallic.com/] where you can see WACS in action and download the sample sets featured in this document - *CAUTION: contains adult material*. The main developers can be contacted at wacs@beaky.name [mailto:wacs@beaky.name] - messages sent to this address will be treated as confidential. Any email addresses, names, URLs or other identification information will be removed before any publication of the contents on bug lists, feature requests, mailing lists etc. 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

Part I. Command Line Tools for WACS Users

The commands listed here cover those tools that are designed to be of use to users of the WACS system rather than those targetted to system administration tasks.

Name

rungq — run an image viewer with a local slide show of Wacs image sets
runmp — run a movie player with a local list of Wacs video clips

Synopsis

```
rungq [modifiers] argument
```

```
runmp [modifiers] argument
```

Arguments

`tag`*`number`*	specifies that the *tag set* (aka Saved Search) of the number given should be used. This can be abreviated to **t** and the number or even just the tag number on it's own.
`conn`*`number`*	specifies that the *connections set* of the number given should be used. This can be abreviated to **c** and the connection set number.
`model`*`number`*	specifies that the complete portfolio of the model specified should be used. This can be abreviated to **mod** or even **m** and the model's number.

Modifiers

`--cat=`	specifies that only a single category of sets should be shown from those that would otherwise match - this uses the standard values for `scatflag`. The values are as follows: F for Straight, G for Group Orgy, M for Masturbation, L for Lesbian, S for Solo and T for Toys. Thus `--cat=T` will show only toy sets.
`--excl=`	specifies what categories of sets should be excluded - the default is Backstage and Duplicate. These are provided as a comma separated list. To show only solo sets you would specify `--excl=F,G,L,B,D`. To cancel the default exclusions, specify a blank exclusion list: `--excl=`.
`--skip=`	specifies how many of the sets that would otherwise have matched should be skipped. This allows you to restart partway through a particular set of matching sets/clips, eg `--skip=15` starts at the 16th matching set.

Description

While the Wacs system is primarily web-based, through tools like **rungq** and **runmp**, it is possible to access both the underlying database and image files/video clips in the WACS archive directly within the Linix/Unix shell environment. This offers some advantages in terms of speed and flexibility of display over accessing it through a web-browser but of course looses both the ubiquity and just-works aspects of using the web interface. Using **rungq** or **runmp** is very much an optional alternative with restricted functionality to using the full web interface.

In order to use **rungq** or **runmp** you will need to have the appropriate access at the operating system level to the database and to the files themselves; this is normally achieved through being a member of the Linux/Unix group `wacs`. It is also possible to make this work on a workstation or laptop computer on the local network through a combination of NFS and network transport of the SQL requests to the database. More information on how to configure these is given in the Configuration Manual for the WACS system.

In essence what **rungq** does is retrieve the full pathnames of the directories containing the image sets specified and invokes an image viewer on those. It opens the first image at which point it is normal to fullscreen the viewer and either page through the images using the space bar or left mouse click, or set a slideshow going with a fixed interval between images. **rungq** can *play* in this way saved searches (aka tag sets), connections or a specific models complete portfolio. Various modifiers described above can make further sub-selections based on set type or skip over a number of the entries if these have been recently viewed. The **runmp** command works in a very similiar way but passes the full path names of the video clip files instead.

rungq uses whatever viewer is specified in the configuration file, or by default **gqview** because this viewer will let the user browse through a slideshow of image directories specified in this way. This behaviour was added to **gqview** at our request by the original author - many if not most other image viewers require explicit specification of the image files themselves or open all the files at once (eog used to do this although by GNOME 2.26 it seems to work correctly). The configuration variable is called `imageviewer` and is located in the `external` section of the file - please see the WACS configuration guide for more details.

runmp uses the **mplayer** but could easily use a different movie player if that is prefered. Again this can be configured through the standard WACS configuration file and is controlled by the variable `movieplayer` in the `external` section. This often includes the `-fs` command line option to cause the player to open in full screen mode.

Examples

- `rungq t23` - show the image sets in saved search number 23
- `rungq --cat=T t23` - show the image sets of category *T*oys from saved search 23
- `rungq  --excl=B,D,F,G  c7` - show the image sets from connection set number 7 skipping Backstage, Duplicate and Straight Sex sets.
- `rungq --skip=15 mod17` - show the portfolio of model number 17, starting from the 16th set.
- `runmp --cat=L m23` - show the Lesbian movies featuring model number 23.

Part II. Command Line Tools for WACS Collection Administration

The commands listed here are various tools that aid in collection administration aspects of managing the WACS system.

Name

wacschk — Perform routine sanity checks on Wacs collection

Synopsis

`wacschk` *`set_number`* `wacschk [--dnl]` *`model_number`*

Argument

`set_number`	the number of the set you wish to have checked.
`model_number`	the number of the model you to check download details for.

Modifiers

`--dnl OR --download`	specifies that download details should be checked rather than set details - following argument is then expected to be a model number.

Description

The **wacschk** command is used to perform various standard checks on elements of the Wacs collection. When passed a set number it performs a number of simple checks to make sure the set files do actually exist in the location specified in the Wacs database.

Examples

`wacschk --dnl 123` would check all the download records for model no *123* making sure that those marked as completed had an associated set number and that those marked as pending did indeed have the necessary archive file in the download area waiting to be unpacked.

See Also

wacsgenimg(1), wacsgenvid(1), wacsupdinfo(1), wacsupdstat(1)

Name

wacsvidcomb — Combine the elements of a multi-part video into a single playable file

Synopsis

```
wacsvidcomb set_number
```

Argument

`set_number`	the number of the first (primary or secondary) video set at the start of a multi-set sequence

Description

The **wacsvidcomb** command is used to combine a group of smaller video files into a playable compilation file that contains all of the content in a single file. It is invoked with the set number of the first video clip in the chain and automatically follows the next links to all sets of *continuation* type and then creates a compilation of them using mplayer's mencoder application. Where the `skip frame` value has been set for the set, this will be used to remove any pre-amble from the intermediate sets.

You do need to have the **mencoder** command installed and in the standard path of the user running the command. You also need to ensure that it can see all the necesary codecs to playback and encoder the types of file you wish to handle. Some of these may not be in the default installation for copyright, licensing or patent encumberence issues - you may need to install video codes from a third party. Once the encoding is complete, the resultant video file will be placed into the `process` directory of the WACS unpack area. The **wacscreatectl** command will collect it from here and place it into the content cache area on the web tree.

Examples

`wacsvidcomb 1234` would check set number *1234* for being a video set with either Primary or Secondary type and having follow-on files. Assuming that they are 1235, 1236, and 1237; `wacsvidcomb` would make a video compilation consisting of sets 1234, 1235, 1236 and 1237 - a four-part video compiled into a single continuous play file.

See Also

wacscachectl(8)

Name

wacscachectl — Controls the content caching mechanism within WACS

Synopsis

```
wacscachectl [options]  set_number
```

Argument

options	the action you want to perform - initially **--create** is the only one supported
set_number	The set number to work on

Description

The **wacscachectl** command is used to configure the links for the content caching system and copy a compilation file created by **wacsvidcomb** into it's correct home from the process area in the Wacs download area. The command probes the set details in the database to determine what needs to be done (if it can). If it fails, the set probably doesn't have modern set status marking and this can be easily fixed using the **wacssetmgr** web app.

See Also

wacsvidcomb(8)

Name

wacsupdinfo — Index and update database details about image sets

Synopsis

```
wacsupdinfo directory_tree
```

Description

The **wacsupdinfo** command is the main tool for managing image sets within the WACS system and combines a number of activities within the single command. It will add any new sets found within the directory tree it's pointed add, it will detect and act on any relocations that are necessary, and it will redo the keyword processing updating set attributes where necessary. It was known by the name **updateinfo** in releases prior to Wacs 0.9.2 when it was renamed to make it's identity as part of the WACS system clearer and to avoid any future name clashes with other applications.

In most cases you will not have cause to run **wacsupdinfo** directly as this is done for you by the **wacsgenimg** which also handles the creation of appropriate thumbnail images for the sets. In fact, you probably won't even run **wacsgenimg** as this is run for you by the Web GUI's **wacsplacemgr** application when uploading a set!

See Also

wacsgenimg(1), wacsgenvid(1)

Part III. Data Migration Tools

The tools covered in this section are related to those tools that allow the migration of data between various WACS installations.

Name

wacsexport — export a WACS model record as an XML file

Synopsis

```
wacsexport [--noimages] model_number
```

Argument

`model_number`	the number of the model you wish to export details of.

Modifiers

`--noimages`	specifies that images should not be included. This is used to exclude copyrighted images when exporting model information for public distribution (ie posting on a forum).

Description

The **wacsexport** command exports all of the information the current WACS installation knows about the specified model as an XML file called by her name and model number. Thus exporting Kaz B's model record from PinkMetallic.com would create an XML file called `KazB-3.xml`. This includes not only the basic model record but also all of her identity map (idmap) records and all related download records. Additionally it also includes any headshot or other icon pictures it has of her (unless told not to via `--noimages`) as Base-64 encoded elements within the XML file.

The resulting XML file is suitable for emailing, etc as it is "ASCII armoured" for the protection of binary data (ie images). It can be imported into another WACS installation using either the **wacsimport** command or the upload function of the **wacsmodelmgr** (WACS Model Manager> web application. The transfer of sets themselves and their associated descriptive data (metadata) is handled seperately by the **wacsxmlout** and **wacsxmlin** commands.

Examples

`wacsexport 3` would create a file called KazB-3.xml if Kaz B is model 3 on the current WACS installation.

See Also

wacsimport(1), wacsxmlin(1), wacsxmlout(1), wacsselout(1)

Name

wacsimport — imports details of a model from a WACS XML export file

Synopsis

```
wacsimport Model_XML_File
```

Description

Data about models can be exchanged between WACS installations by way of a number of XML file formats, one of which is that which describes models. The **wacsimport** command is provided to load a recieved WACS Model XML file into the WACS system; this includes the model's basic details, her known identities, her known sets and both the large and small headshot icons.

The **wacsimport** command tries to determine if the model described by the XML file is already known to the current WACS installation. If she isn't, it will create a new record for her, install any headshot icons enclosed in the XML file, add all her known identities and all known set information. If she is, it will update the existing record with any *new* information included within the XML file. This will include any new identities, known set information (aka download records) and any fields where there is no data already.

For instance, if her *hometown* is blank in our local data about her, but present in the XML file, it will be updated - existing values will not be overwritten. There is one exception to this in that if an identity is marked as Active or Dormant in the database and marked as Gone in the XML file, it will be marked as Gone in the database. This is because it's a data item unlikely to be factually inaccurate and is very likely to change over time.

Some sample XML model files are provided in the WACS distribution and additional ones will be made available through our demonstration site www.pinkmetallic.com as they become available.

See Also

wacsexport(1), wacsxmlin(1), wacsxmlout(1), wacsselout(1)

Name

wacsxmlout — exports set details as a WACS XML export file

Synopsis

```
wacsxmlout [--noimages] set_number
```

Description

The **wacsxmlout** command is a tool to create an XML export file containing information on a set to allow it to be transported between different WACS installations. **wacsxmlout** is invoked with a set number (ie set 123) and will create a file named like `set123.xml`. The XML file **wacsxmlout** creates contains the set information, all associated download records, associations and idmap records for all of the models featured in the set. Additionally, unless `--noimages` was specified, it will contain the thumbnail icon, official icon and any additional icons related to the set.

wacsxmlout *DOES NOT* export the set itself and will need to be recombined with the appropriate zip or movie file when it's loaded into the destination WACS system. These need to be downloaded seperately. Please note that the WACS 0.9.x versions of **wacsxmlout** generate version 2 export files containing database fields not found in earlier schemas. It is not currently possible to automatically backport a version 2 file onto a WACS 0.8.x or earlier installation.

Sample XML files can be downloaded from the resources section on our demonstration site www.pinkmetallic.com as they become available.

See Also

wacsimport(1), wacsexport(1), wacsxmlin(1), wacsselout(1)

Name

wacsxmlin — imports set details from a WACS XML export file

Synopsis

```
wacsxmlin [modifiers] XML_file_name
```

Modifiers

`--default`	specifies that the set should be stored in the calculated default location for this WACS server instead of it's location on the original server.
`--clone`	specifies that this should be an *EXACT* copy of the original - same location, set number, etc. Useful in creating clone web servers for resilience, etc.
`--asnew`	tells **wacsxmlin** to mark this as a newly created set so it appears in the new sets/ videos index.

Description

The **wacsxmlin** command is a tool to import a WACS set export XML file containing information about a set to allow it to be transported between different WACS installations. **wacsxmlin** is invoked with the name of the set XML file created previously by the **wacsxmlout** command on another WACS installation. For a set numbered 123 on the originating site, the inputs to **wacsxmlin** would be `set123.xml` and `set123.zip`. This set will normally recieve a new set number on importation into the new WACS installation.

Sine the XML file **wacsxmlin** reads contains the set information, all associated download records, associations and idmap records for all of the models featured in the set, **wacsxmlin** does it's best efforts to find and associate the set being added to those models who feature in it on the new WACS installation. If the export file contained them, **wacsxmlin** will unpack the thumbnail icon, official icon and any additional icons related to the set.

Sample XML files can be downloaded from the resources section on our demonstration site www.pinkmetallic.com as they become available and can be used to import our sets into your own installation. Since these sets are released under the Creative Commons license, you are free to import them into your WACS server and even include them on a commercial site if you wish to, providing the PinkMetallic.com branding remains intact. We can provide proof of age documentation to US-based sites in order to conform to 18 USC 2257 for a small fee.

wacsxmlin in WACS 0.9.0 and higher understands both version 1 (version 0.8.6 and earlier) and version 2 (version 0.9.0 and later) files. Either type may be used on an installation providing the database schema is up to date.

See Also

wacsimport(1), wacsexport(1), wacsxmlout(1), wacsselout(1)

Name

wacsselout — exports the details of a selection as a WACS XML export file

Synopsis

`wacsselout t`*`tag_number`*

`wacsselout c`*`connection_number`*

Description

The **wacsselout** command is a tool to create an XML export file containing information about a selection of sets to allow them to be transported between different WACS installations. It can work with either saved searches (aka tag sets) or with connections.

To export saved search no 23, you would run **wacsselout t23** and it will create a file called `tag23.xml` in the current directory. To export connection no 10, you would run **wacsselout c10** and it will create a file called `conn10.xml` in the current directory.

Bugs

Currently there is only one very limited way to make use of the selection file thus exported, namely wacsselin . This will be enhanced and upgraded in a future release. The XML file created by **wacsselout** does contain all the necessary information and so should be a valid backup for future use.

See Also

wacsselin(1), wacsimport(1), wacsexport(1), wacsxmlin(1), wacsxmlout(1)

Name

wacsselin — imports a selection from a WACS XML export file as created by **wacsselout**

Synopsis

```
wacsselin tagnumber.xml

wacsselin connnumber.xml
```

Description

The **wacsselin** command is a tool to import a selection, either a saved search or a connection, from an XML file containing a selection of sets. It can work with either saved searches (aka tag sets) or with connections and is the counterpart to **wacsselout**. It's new in WACS 0.8.6 and currently has very limited functionality as it requires all the models and sets to have already been created and just "joins them up". It does however report anything found to be missing, so you can progressively fix it. Future releases will have additional functionality.

Examples

To import a previously created selection file that contains a tag set called originally called 40 on the source server:

```
% wacsselin tag40.xml
%
```

See Also

wacsselout(1), wacsimport(1), wacsexport(1), wacsxmlin(1), wacsxmlout(1)