

**XADMIN(8)**

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>SYNOPSIS</b>	<b>1</b>
<b>2</b>	<b>DESCRIPTION</b>	<b>2</b>
<b>3</b>	<b>COMMANDS</b>	<b>3</b>
<b>4</b>	<b>CONFIGURATION</b>	<b>8</b>
<b>5</b>	<b>SAMPLE CONFIGURATION</b>	<b>9</b>
<b>6</b>	<b>EXIT STATUS</b>	<b>10</b>
<b>7</b>	<b>BUGS</b>	<b>11</b>
<b>8</b>	<b>SEE ALSO</b>	<b>12</b>
<b>9</b>	<b>COPYING</b>	<b>13</b>

## Chapter 1

# SYNOPSIS

**xadmin** [*COMMAND*] [*OPTIONS*]

## Chapter 2

# DESCRIPTION

*xadmin* is command line interface to Enduro/X. It is administration utility. Which is servers as communication interface between Enduro/X ATMI local daemon and administrator. *xadmin* can receive commands as parameter to *xadmin*. It can read command from pipe (stdin). And it can work in interactive mode. In this case *xadmin* is started with no parameters.

*xadmin* is responsible to start idle instance of *ndrxd*. After that *xadmin* sends commands to *ndrxd* and prints the response from daemon to the stdout in human readable form. Commands are sent and response from *ndrxd* are received via POSIX Queues.

*xadmin* can lookup the configuration (currently used for user specific generators). The configuration search order is following:

1. INI file by *NDRX\_XADMIN\_CONFIG* environment variable. This can also be a directory. The common-config routines will load all configs (.ini .cfg, .conf, .config) files.
2. INI file in *\$HOME/.xadmin.config*;
3. INI file in */etc/xadmin.config*;

## Chapter 3

# COMMANDS

**quit**

Quit from command line utility

**q**

Alias for quit

**exit**

Alias for quit

**echo**

Echo text back to terminal

**idle**

Enter daemon process in idle state (if not started)

**help**

Print help (this output)

**stat**

Prints general status information

**poller**

Print polling sub-system name i.e. **epoll** - when build is using Linux e-poll sub-system, this allows true one queue multiple server approach to be used. **kqueue** - the same benefits as for Linux **epoll** only on BSD platform, **SystemV** - for this build System V queues are used (instead for others Posix queues are used). In case of System V still one queue multiple servers are available with configuration of request address. **poll** - this is worst mode for Enduro/X as only Round Robin message distribution is available for dispatching messages to servers. This uses only standard features of Posix queues. The poller is built in the distribution and cannot be changed via configuration, only possible changing is with reinstalling other Enduro/X build.

**ldcf**

Load configuration

**start [-y] [-s <server>] [-i <srvid>]**

Start application domain. If config not loaded, it loads configuration automatically. -y means do not ask for confirmation. -s means ATMI servers binary name to be started, instead of whole application domain. -i means start specific server instance.

**psc**

Print services

**stop [-y] [-s <server>] [-i <srvid>] [-k]**

Stop application domain. -y means do not ask for confirmation. -s means ATMI servers binary name to be stopped, instead of whole application domain. -i means stop specific server instance. -k flag will keep *ndrxd* running, and protected servers will not be shutdown.

---

**sreload [-y] [-s <server>] [-i <srvid>]**

Reload application domain - restart server instance by instance. Configuration be loaded prior. -y means do not ask for confirmation. -s means ATMI servers binary name to be reloaded, instead of whole application domain. -i means restart specific server instance.

**sr**

Alias for *sreload*

**psc**

Print available services.

**down [-y]**

Force appserver shutdown & resources cleanup. RUN ONLY IF YOU KNOW WHAT YOU ARE DOING! Basically this kills all ATMI servers and Enduro/X daemon. This does NOT remove client processes.

**cat**

Attached console to ndrxd user session in progress

**reload**

Load new configuration

**testcfg**

Test new configuration

**unadv -i server\_id -s service\_name**

Un-advertise service. -i is server id, -s is service name to be unadvertised.

**readv -i server\_id -s service\_name**

Re-advertise service. Might be usable if service Q was unlinked. -i is server id, -s is service name to be re-advertised.

**restart [-y] [-s <server>] [-i <srvid>]**

Restart app or service (invokes start & stop with same args!). -y makes to not to ask for confirmation. -s is server/binary name. -i is server ID.

**r**

Alias for *restart*

**-v**

Print version info

**ver**

Alias for -v

**ppm [-2]**

Print process model. The argument -2 show second page of process model attributes.

**psvc**

Shared mem, print services

**psrv**

Shared mem, print servers

**cabort [-y]**

Abort shutdown or startup operation in progress. -y do not ask for confirmation.

**sreload [-y] [-s <server>] [-i <srvid>]**

Restart servers instance by instance

**pq**

Print Queue statistics from ndrxd.

**pqa [-a]**

Print all queues including client and admin Q. -a includes other prefix queues.

**pt**

Print global transactions in progress.

**printtrans**

Alias for *pt*.

**abort -t <transaction\_manager\_reference> -x <XID> [-g <resource\_manager\_id>] [-y]**

Abort transaction. -g does abort single resource manager's transaction. -y is for auto confirmation.

**aborttrans**

Alias for *abort*.

**commit -t <transaction\_manager\_reference> -x <XID> [-y]**

Commit transaction. -y is for auto confirmation.

**committrans**

Alias for *commit*.

**pe**

Print Environment variables of *ndrxd* process.

**printenv**

Alias for *pe*.

**set ENV\_NAME=ENV VALUE**

Set environment value. The value of env variable is parsed as command line arguments. Prior sending to *ndrxd* they are concatenated with spaces in between.

**unset ENV\_NAME**

Unset environment variable

**pc**

Print client processes. This sends command to Client Process Monitor server (*cpmsrv*).

**bc -t <process\_tag> [-s <sub\_section>] [-w <wait\_time>]**

Boot client process. This sends command to Client Process Monitor server (*cpmsrv*). Processes are registered in *ndrx-config.xml* <clients> section. If sub section is not specified, then default value is minus sign (-). The *process\_tag* and *sub\_section* can contain wildcards percent (%) sign. Then boot process will be executed in batch mode and progress will be returned to the xadmin's output. When running in batch mode *wait\_time* is time in milliseconds to sleep after each matched process is marked for start. Note that *wait\_time* shall be less than global timeout specified in *NDRX\_TOUT* env variable (or cconfig [*@global*] section).

**sc -t <process\_tag> [-s <sub\_section>] [-w <wait time in milliseconds>]**

Stop client process. This sends command to Client Process Monitor server (*cpmsrv*). The process is stopped by *process\_tag* and optional *sub\_section*. If sub section is not specified, then default value is minus sign (-). The *process\_tag* and *sub\_section* can contain wild-card percent sign (%), then stopping is executed in batch mode (stop all matched running processes). If *wait\_time* is specified then in batch mode it is sleep in milliseconds after each stopped process. Note that *wait\_time* shall be less than global timeout specified in *NDRX\_TOUT* env variable (or cconfig [*@global*] section). Also time needed for stopping shall be counted in. If the timeout occurs, *cpmsrv* will complete the operation anyway.

**rc -t <process\_tag> [-s <sub\_section>] [-w <wait time in milliseconds>]**

Reload client process. This sends command to Client Process Monitor server (*cpmsrv*). The process is reloaded (stopped/-marked for start) by *process\_tag* and optional *sub\_section*. If sub section is not specified, then default value is minus sign (-). The *process\_tag* and *sub\_section* can contain wild-card percent sign (%), then reloading is executed in batch mode (stop/start running processes one by one). If *wait\_time* is specified then in batch mode it is sleep in milliseconds after each stopped process. Note that *wait\_time* shall be less than global timeout specified in *NDRX\_TOUT* env variable (or cconfig [*@global*] section). Also time needed for stopping shall be counted in. If the timeout occurs, *cpmsrv* will complete the operation anyway. The *wait\_time* can be used in cases when reloading the binaries with out service interruption, in that case *wait\_time* should contain the *cpmsrv*'s interval check time (-i flag) because for start operation binary is only marked for boot and not the booted. Basically this executes *sc/bc* for each of the matched processes.



**mqlc**

List queue configuration. This broadcasts the requests of config listing to all *tmqueue* servers. If flags column contains *D* flag, then it means that queue was dynamically defined and QDEF string contains values from default queue.

**mqlq**

List actual queues allocated on system. Similarly as for *mqlc* this requests the information from all *tmqueue* servers. *#LOCK* column contains the number of active non committed messages in Q. *#SUCC* and *#FAIL* column contains number of processed messages for automatic queues (messages are sent to destination services automatically by *tmqueue* server).

**mqrq**

This command requests all queue servers to reload the configuration file.

**mqlm -s <QSpace> -q <QName>**

List messages in queue. *-s* is queue space name (set by *tmqueue -m* paramemter). The output lists the message ID in modified base64 version (/ changed to \_).

**mqdm -n <Cluster node id> -i <Server ID> -m <Message ID>**

Dump/peek message to stdout. The values from *-n* (node id), *-i* (svid), *-m* (message id) can be taken from *mqlm* command. This command prints to stdout, the *TQCTL* structure in form of UBF buffer and the message it self. If message is UBF, then UBF dump is made, otherwise hexdump of message is printed.

**mqch -n <Cluster node id> -i <Server ID> -q <Q def (conf format)>**

Change/add queue definition to particular *tmqueue* server. The format of the queue definition is the same as used *q.conf(5)* (see the man page). You may miss out some of the bits (except the queue name). Those other bits will be take from default q.

**mqrm -n <Cluster node id> -i <Server ID> -m <Message ID>**

Remove message from queue. You have to identify exact queue space server here by Enduro/X cluster id and server id.

**mqmv -n <Source cluster node id> -i <Source server ID> -m <Source Message ID> -s <Dest qspace> -q <Dest qname>**

Move the message from specific qspace server to destination qspace and qname. The bits from *TPQCTL* which are returned by *tpdequeue()* call are preserved in new *tpenqueue()* call. Note that for this call *xadmin* must be in invalid XA environment, so that distributed transaction can be performed.

**killall <name1> [<name2> ... <nameN>]**

Kill all processes given by *ps -ef*. The command does match the name in the line. If substring is found, then process is killed.

**qrm <qname1> [<qname2> ... <qnameN>]**

Remove specific Posix queue.

**qrmall <substr1> [<substr2> ... <substrN>]**

Remove queue matching the substring.

**provision [-d] [-v<param1>=<value1>] ... [-v<paramN>=<valueN>]**

Prepare initial Enduro/X instance environment, create folder structure, generate configuration files with ability to register all available services.

**gen [-d] [-v<param1>=<value1>] ... [-v<paramN>=<valueN>]**

Generate application sources. See the *xadmin*'s help for more details. Currently it is possible to generate C and Go sources and the UBF buffer headers for both languages. By running the command, wizards will be offered asking for different details. Which later can be reconfigured by *-d* - allowing to default the wizard, while *-v* allows to set wizard values from command line.

**pubfdb**

Print UBF custom fields database contents to the terminal.

**cs <cache\_db\_name>|-d <cache\_db\_name>**

Print cache contents (headers) to the terminal. The database name is specified in *cache\_db\_name* parameter.

**cacheshow**

Alias for *cs*.

**cd -d <dbname> -k <key> [-i interpret\_result]**

Dump specified message to the terminal. With specified *-i* flag, the attempt for data interpretation will be made. For UBF buffers the output will be formatted with command **Bprint()**.

**cachedump**

Alias for *cd*.

**ci -d <dbname> [-k <key>][-r use\_regexp]**

Invalidate cache. In case if only *-d* is specified, whole database will be dropped. Exact record may be dropped with *-k* flag. In case if *-r* is used, then key will be matched as regular expression over the data keys. In case if drop database is used, linked keygroup records are not processed. For other scenarios, linked records are processed according to the configuration.

**cachedump**

Alias for *cd*.

**svmaps [-p] [-s] [-a] [-i] [-w]**

Command is available only for System V messaging sub-system. Command prints the Queue ID (same ids from **ipcs** command) mappings to Posix queues used by Enduro/X. **-p** parameter (which is enabled by default) prints the mapping table from Posix Queue to System V. **-s** parameter uses reverse table by printing System V mappings to Posix. **-a** enables to print all the entries in the shared memory (there could be lots of lines printed. Totally set by **NDRX\_MSGQUEUESMAX** environment variable). **-i** (used by default) prints only the queues which currently are in use. **-w** prints mappings which were in use, but currently are not in use. The **-i**, **-w** arguments can be combined.

**shms**

Shows currently open shared memory segments.

**pmode**

Prints technical information about Enduro/X build mode (basically *ndrx\_config.h*).

## Chapter 4

# CONFIGURATION

The following parameters from section [**@xadmin**] or [**@xadmin/(<\$NDRX\_CCTAG>)**] are used (if config file is present):

**gen scripts=***PATH\_TO\_GENERATOR\_SCRIPTS*

This parameter configures the path where *xadmin* should look for .pscript files. The file names must be in following format: `gen_<lang>_<type>.pscript`. Basically <lang> and <type> will be offered as targets under `$xadmin gen` command. The `$xadmin help` will print these scripts. For script reference look in Enduro/X source code, **xadmin/scripts** folder. It is assumed that these scripts will inherit *WizardBase* class compiled into Enduro/X. This class is driving the wizard. Also note that each parameter which is asked to user enter into wizard, can be overridden from command line with **-v<param1>=<value1>**. The generator can be defaulted by **-d** argument.

## Chapter 5

# SAMPLE CONFIGURATION

For system wide settings the following file is created: **/etc/xadmin.config**:

```
[@xadmin]  
gen scripts=/development/templates
```

## Chapter 6

# EXIT STATUS

**0**      Success

**1**      Failure

## Chapter 7

# BUGS

Report bugs to [support@mavimax.com](mailto:support@mavimax.com)

## Chapter 8

## SEE ALSO

**ndrxd(8), q.conf(5), tmqueue(8) cpmsrv(8)**

## Chapter 9

# COPYING

© Mavimax, Ltd