YAZ proxy User's Guide and Reference

COLLABORATORS			
	TITLE : YAZ proxy User's Guide and Reference		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Mike Taylor and Adam Dickmeiss	December 11, 2019	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

List of Tables

Abstract

YAZ proxy is a powerful general purpose Z39.50/SRU proxy.

This manual covers version 1.3.11.

Chapter 1

Introduction

The YAZ Proxy is highly configurable and can be used in a number of different applications, ranging from debugging Z39.50-based applications and protecting overworked servers, to improving the performance of stateless WWW/Z39.50 gateways. Among other features, it includes:

- SRU GET/POST/SOAP server function, to allow any Z39.50 server to also support the SRU protocols.
- Load balancing across multiple backend servers
- Session-sharing and pre-initialization to improve performance in servers with expensive session initialization
- Configurable request filtering, to keep bad requests from reaching the server
- XML support -- MARC records can be converted to MARCXML, and XSLT-transformations allow the proxy to support arbitrary retrieval schemas in XML
- Load governor function limits requests from aggressive batch-mode clients
- Configurable logging
- Efficient multiplexing software enables small memory footprint and very high performance
- Z39.50 character set negotiation support.

1.1 Licensing

The proxy application and the proxy library is covered by the GPL.

1.2 Support

Configuration and installation assistance and ongoing support is available for the YAZ Proxy. For further information about support or licensing options, please contact David Dorman in the US (dorman at indexdata.com, 860-346-1237 or toll free 866-489-1568) or Sebastian Hammer in Denmark (quinn a indexdata.com, or +45 3341 0100)

Chapter 2

Installation

You need a C++ compiler to compile and use YAZ proxy. The software was implemented using GCC so we know that works well with YAZ proxy. From time to time the software has been compiled on Windows using Visual C++. Other compilers should work too. Let us know of portability problems, etc. with your system.

YAZ proxy is built on top of the YAZ and YAZ++ toolkits. You need to install these first. For some platforms there are binary packages available for YAZ/YAZ++.

We also highly recommend that libxml2 and libXSLT are installed. YAZ must be configured with libxml2 support. If not, SRU is not supported. The YAZ Proxy uses libXSLT for record conversions via XSLT.

YAZ proxy may also use USEMARCON to convert between MARC formats. This is useful if you want the proxy to offer more MARC record types than the backend target supports. Get USEMARCON from: British Library USEMARCON page .

2.1 Building on Unix

On UNIX, the software is compiled as follows:

```
$ ./configure
$ make
$ su
# make install
```

You can supply options for the `configure` script. The most useful ones are:

`--prefix` directory Specifies installation prefix. By default `/usr/local` is used.

`--with-yazpp` directory Specifies the location of `yazpp-config`. The `yazpp-config` program is generated in the source directory of YAZ++ as well as the binaries directory when YAZ++ is installed (via make install).

If you don't supply this option, `configure` will look for `yazpp-config` in directories of the `PATH` environment - which is nearly always what you want.

--with-xslt directory Specifies prefix for libXSLT (and libxml2). configure must be able to locate **xslt-config** in PREFIX/bin. If this option is omitted, configure looks for **xslt-config** in the current PATH.

--with-usemarcon directory Specifies USEMARCON installation prefix. configure must be able to locate **usemarcon-config** in PREFIX/bin. If this option is omitted, configure looks for **usemarcon-config** in the current PATH.

For the whole list of `configure` options, refer to the help: `./configure --help`.

Configure uses GCC's C/C++ compiler if available. To specify another compiler, set `CXX`. To use other compiler flags, specify `CXXFLAGS`. For example, to use `CC` with debugging do:

```
CXXFLAGS="-g" CXX=CC ./configure
```

This is what you have after successful compilation:

src/yazproxy The YAZ Proxy program. It gets installed in your binaries directory (*prefix*`/bin`).

src/libyazproxy.la The YAZ proxy library. This library gets installed in the libraries directory (*prefix*`/lib`).

include/yazproxy/*.h C++ header files, which you'll need for YAZ proxy development. All these are installed in the header files area (*prefix*`/include/yazproxy`).

etc Various files such as configuration files, XSLT files, CQL to RPN conversion files, a sample start/stop control script `yazproxy.ctl.sh` that can be used as template for an `/etc/init.d` script. These files are installed in the YAZ proxy's data area (*prefix*`/share/yazproxy`).

2.2 Building on Windows

YAZ++ is shipped with "makefiles" for the NMAKE tool that comes with Microsoft Visual Studio. Version 6 and .NET has been tested. We expect that YAZ++ compiles with version 5 as well.

Note
The YAZ proxy has never been used in production on Windows. Although it compiles and runs, doesn't mean it scale on that platform. Furthermore the YAZ proxy currently doesn't run as a Service - only as a Console application.

Start a command prompt and switch the sub directory `WIN` where the file `makefile` is located. Customize the installation by editing the `makefile` file (for example by using notepad). The following summarizes the most important settings in that file:

DEBUG If set to 1, the software is compiled with debugging libraries (code generation is multi-threaded debug DLL). If set to 0, the software is compiled with release libraries (code generation is multi-threaded DLL).

YAZ_DIR This must be set to the home of the YAZ source directory.

YAZPP_DIR This must be set to the home of the YAZ++ source directory.

HAVE_XSLT, LIBXSLT_DIR If HAVE_LIBXSLT is set to 1, the proxy is compiled with XSLT and XML support. In this configuration, set LIBXSLT_DIR to the libXSLT source directory.

> **Note**
> If you enable libXSLT you have to enable libxml2 and its sub components zlib and iconv as well.

Windows versions of libXSLT, libxml2, zlib and iconv can be found here.

HAVE_ICONV, ICONV_DIR If HAVE_ICONV is set to 1, the proxy is compiled with iconv support. In this configuration, set ICONV_DIR to the iconv source directory.

HAVE_LIBXML2, LIBXML2_DIR If HAVE_LIBXML2 is set to 1, the proxy is compiled with XML support. In this configuration, set LIBXML2_DIR to the libxml2 source directory and ZLIB_DIR to the zlib directory.

> **Note**
> YAZ++ is not using ZLIB. But libxml2 is.

When satisfied with the settings in the makefile, type

```
nmake
```

> **Tip**
> If the nmake command is not found on your system you probably haven't defined the environment variables required to use that tool. To fix that, find and run the batch file vcvars32.bat. You need to run it from within the command prompt or set the environment variables "globally"; otherwise it doesn't work.

If you wish to recompile YAZ++ - for example if you modify settings in the makefile you can delete object files, etc by running.

```
nmake clean
```

The following files are generated upon successful compilation:

bin/yazproxy.dll YAZ proxy DLL.

lib/yazproxy.lib Import library for yazproxy.dll.

bin/yazproxy.exe YAZ proxy. It's a WIN32 console application.

Chapter 3

Using YAZ proxy

As mentioned in the introduction the YAZ proxy has many uses. This chapter includes a few examples.

The YAZ Proxy is useful for debugging SRU/Z39.50 software, logging APDUs, redirecting Z39.50 packages through firewalls, etc. Furthermore, it offers facilities that often boost performance for connection-less Z39.50 clients such as web gateways.

Unlike most other server software, the proxy runs single-threaded, single-process. Every I/O operation is non-blocking so it is very lightweight and extremely fast. It does not store any state information on the hard drive, except any log files you ask for.

Example 3.1 Using the Proxy to Log APDUs

Suppose you use a commercial Z39.50 client for which you do not have source code, and it's not behaving how you think it should when running against some specific server that you have no control over. One way to diagnose the problem is to find out what packets (APDUs) are being sent and received, but not all client applications have facilities to do APDU logging.
No problem. Run the proxy on a friendly machine, get it to log APDUs, and point the errant client at the proxy instead of directly at the server that's causing it problems.
Suppose the server is running on `foo.bar.com`, port 18398. Run the proxy on the machine of your choice, say `your.company.com` like this:

```
yazproxy -a - -t tcp:foo.bar.com:18398 tcp:@:9000
```

(The `-a -` option requests APDU logging on standard output, `-t tcp:foo.bar.com:18398` specifies where the backend target is, and `tcp:@:9000` tells the proxy to listen on port 9000 and accept connections from any machine.)
Now change your client application's configuration so that instead of connecting to `foo.bar.com` port 18398, it connects to `your.company.com` port 9000, and start it up. It will work exactly as usual, but all the packets will be sent via the proxy, which will generate a log like this:

```
decode choice
initRequest {
    referenceId OCTETSTRING(len=4) 69 6E 69 74
    protocolVersion BITSTRING(len=1)
    options BITSTRING(len=2)
    preferredMessageSize 1048576
    maximumRecordSize 1048576
```

```
        implementationId 'Mike Taylor (id=169)'
        implementationName 'Net::Z3950.pm (Perl)'
        implementationVersion '0.31'
    }
    encode choice
    initResponse {
        referenceId OCTETSTRING(len=4) 69 6E 69 74
        protocolVersion BITSTRING(len=1)
        options BITSTRING(len=2)
        preferredMessageSize 1048576
        maximumRecordSize 1048576
        result TRUE
        implementationId '81'
        implementationName 'GFS/YAZ / Zebra Information Server'
        implementationVersion 'YAZ 1.9.1 / Zebra 1.3.3'
    }
    decode choice
    searchRequest {
        referenceId OCTETSTRING(len=1) 30
        smallSetUpperBound 0
        largeSetLowerBound 1
        mediumSetPresentNumber 0
        replaceIndicator TRUE
        resultSetName 'default'
        databaseNames {
            'gils'
        }
        {
            smallSetElementSetNames choice
            generic 'F'
        }
        {
            mediumSetElementSetNames choice
            generic 'B'
        }
        preferredRecordSyntax OID: 1 2 840 10003 5 10
        {
            query choice
            type_1 {
                attributeSetId OID: 1 2 840 10003 3 1
                RPNStructure choice
                {
                    simple choice
                    attributesPlusTerm {
                        attributes {
                        }
                        term choice
                        general OCTETSTRING(len=7) 6D 69 6E 65 72 61 6C
                    }
                }
```

```
            }
        }
    }
```

Example 3.2 Using a configuration file

In Example 3.1 the default backend server was specified by a command line option. The same proxy behavior can be achieved by creating a configuration with the following contents:

```
<?xml version="1.0"?>
<proxy xmlns="http://indexdata.dk/yazproxy/schema/0.9/">
  <target name="foo" default="1">
    <url>foo.bar.com:18398</url>
    <log>client-apdu</log>
  </target>
  <target name="*">
  </target>
</proxy>
```

The proxy is started with

```
yazproxy -c config.xml @:9000
```

The last target section is used for all servers except foo. Had the the last section been omitted, then *only* foo could be reached via the proxy.

Example 3.3 Offering SRU/Z39.50 service

In order to offer SRU service we must be specify sufficient information to allow the proxy to convert from SRU to Z39.50. This involves translating CQL queries to Type-1 (also called RPN/PQF), since most Z39.50 servers do not support CQL. The conversion is specified by the `cql2rpn` element.

We must also ensure that the server can return at least one kind of XML record (Dublin-Core recommended). An explain record for the SRU service must also be created.

The following is a relatively simple configuration file for such a service. This service lives on `indexdata.dk`, port 9000. The database is `gils`. The backend server is also `indexdata.dk` (port 210) as given by `url`.

The server may return USMARC/MARC21 (Z39.50/SRU) and MARCXML (SRU only) as specified by the syntax elements.

```
<?xml version="1.0"?>
<proxy xmlns="http://indexdata.dk/yazproxy/schema/0.9/">
  <target name="bagel">
    <url>indexdata.dk</url>
    <target-timeout>240</target-timeout>
    <client-timeout>180</client-timeout>
    <attribute type="1" value="1-11,13-1016"/>
    <attribute type="1" value="*" error="114"/>
    <syntax type="usmarc"/>
    <syntax type="none"/>
```

```
    <syntax type="xml" marcxml="1"
      identifier="info:srw/schema/1/marcxml-v1.1" >
      <name>marcxml</name>
    </syntax>
    <syntax type="*" error="238"/>
    <preinit>0</preinit>
    <explain xmlns="http://explain.z3950.org/dtd/2.0/">
       <serverInfo>
       <host>indexdata.dk</host>
       <port>9000</port>
       <database>gils</database>
       </serverInfo>
    </explain>
    <cql2rpn>pqf.properties</cql2rpn>
  </target>
</proxy>
```

The conversion from CQL to RPN is specified by a file whose name, relative to the working directory, is given in the cql2rpn element. A complete Bath/DC conversion file, `pqf.properties` is provided as part of the yazproxy distribution in the `etc` subdirectory.
Explain information is embedded in the configuration file. Note that in this example,only a few mandatory explain elements are specified. A well-behaving server should describe index sets, indexes, record schemas as well.

Chapter 4

Proxy Reference

4.1 Operating Environment

The YAZ proxy is a console program. After startup it spawns a child process (except on Windows or if option -X is given). The child process is the core of the proxy and it handles all communication with clients and servers. The parent process will restart the child process if it dies unexpectedly and report the reason. For options for YAZ proxy, see Section 4.10.

As an option, the proxy may change user identity to a less privileged user.

4.2 Choosing the Backend Server

When the proxy receives a Z39.50 Initialize Request from a Z39.50 client, it determines the backend server by the following rules:

1. If the `InitializeRequest` PDU from the client includes an otherInfo element with OID `1.2.840.10` then the contents of that element specify the server to be used, in the usual YAZ address format (typically `tcp:`*`hostname`*`:`*`port`*) as described in the Addresses section of the YAZ manual.
2. Otherwise, the Proxy uses the default server, if one was specified in the proxy configuration file. See Section 4.9.2.
3. Otherwise, the Proxy uses the default server, if one was specified on the command-line with the `-t` option.
4. Otherwise, the proxy closes the connection with the client.

If the proxy receives an SRU request, the following rules are used.

1. If default target has Explain information with a `database` that matches the path of the HTTP request of SRU that backend server is used for SRU operation.
2. Otherwise the service will return HTTP 404 (Not found).

4.3 Keep-alive Facility

The keep-alive is a facility where the proxy keeps the connection to the backend server - even if the client closes the connection to the proxy.

If a new or another client connects to the proxy again and requests the same backend it will be reassigned to this backend. In this case, the proxy sends an initialize response directly to the client and an initialize handshake with the backend is omitted.

When a client reconnects, query and record caching works better, if the proxy assigns it to the same backend as before. And the result set (if any) is re-used. To achieve this, Index Data defined a session cookie which identifies the backend session.

The cookie is defined by the client and is sent as part of the Initialize Request and passed in an otherInfo element with OID `1.2.840.10003.10.1000.81.2`.

Clients that do not send a cookie as part of the initialize request may still better performance, since the init handshake is saved.

Refer to Section 4.9.7 on how to setup configuration parameters for keepalive.

4.4 Query Caching

Simple stateless clients often send identical Z39.50 searches in a relatively short period of time (e.g. in order to produce a results-list page, the next page, a single full-record, etc). And for many targets, it's much more expensive to produce a new result set than to reuse an existing one.

The proxy tries to solve that by remembering the last query for each backend target, so that if an identical query is received next, it is turned into Present Requests rather than new Search Requests.

Note

In a future we release will will probably allows for an arbitrary-sized cache for targets supporting named result sets.

You can enable/disable query caching using option -o.

4.5 Record Caching

As an option, the proxy may also cache result set records for the last search. The proxy takes into account the Record Syntax and CompSpec. The CompSpec includes simple element set names as well. By default the cache is 200000 bytes per session.

4.6 Query Validation

The Proxy may also be configured to trap particular attributes in Type-1 queries and send Bib-1 diagnostics back to the client without even consulting the backend target. This facility may be useful if a target does not properly issue diagnostics when unsupported attributes are send to it.

4.7 Record Syntax Validation

The proxy may be configured to accept, reject or convert records. When accepted, the target passes search/present requests to the backend target under the assumption that the target can honor the request (In fact it may not do that). When a record is rejected because the record syntax is "unsupported" the proxy returns a diagnostic to the client. Finally, the proxy may convert records.

The proxy can convert from MARC to MARCXML and thereby offer an XML version of any MARC record as long as it is ISO2709 encoded. If the proxy is compiled with libXSLT support it can also perform XSLT on XML.

4.8 Other Optimizations

We've had some plans to support global caching of result set records, but this has not yet been implemented.

4.9 Proxy Configuration File

The Proxy may read a configuration file using option `-c` followed by the filename of a config file.

The config file is XML based. The YAZ proxy must be compiled with libxml2 and libXSLT support in order for the config file facility to be enabled.

See Section 4.12 for an XML schema for the configuration.

Tip

To check for a config file to be well-formed, the yazproxy may be invoked without specifying a listening port, i.e.

```
yazproxy -c myconfig.xml
```

If this does not produce errors, the file is well-formed.

4.9.1 Proxy Configuration Header

The proxy config file must have a root element called `proxy` and scoped within namespace `xmlns="http://`
All information except an optional XML header must be stored within the `proxy` element.

```
<?xml version="1.0"?>
<proxy xmlns="http://indexdata.dk/yazproxy/schema/0.9/">
 <!-- content here .. -->
</proxy>
```

4.9.2 target

The element `target` which may be repeated zero or more times with parent element `proxy` contains information about each backend target. The `target` element have two attributes: `name` which holds the logical name of the backend target (required) and `default` (optional) which (when given) specifies that the backend target is the default target - equivalent to command line option `-t`.

```
<?xml version="1.0"?>
<proxy xmlns="http://indexdata.dk/yazproxy/schema/0.9/">
 <target name="server1" default="1">
  <!-- description of server1 .. -->
 </target>
 <target name="server2">
  <!-- description of server2 .. -->
 </target>
</proxy>
```

4.9.3 url

The `url` which may be repeated one or more times should be the child of the `target` element. The CDATA of `url` is the Z-URL of the backend.

Multiple `url` element may be used. In that case, then a client initiates a session, the proxy chooses the URL with the lowest number of active sessions, thereby distributing the load. It is assumed that each URL represents the same database (data).

4.9.4 target-timeout

The element `target-timeout` is the child of element `target` and specifies the amount in seconds before a target session is shut down.

This can also be specified on the command line by using option `-T`. Refer to OPTIONS in Section 4.10.

4.9.5 client-timeout

The element `client-timeout` is the child of element `target` and specifies the amount in seconds before a client session is shut down.

This can also be specified on the command line by using option `-i`. Refer to OPTIONS in Section 4.10.

4.9.6 max-sockets

The element `max-sockets` is the child of element `target` and specifies the maximum number of sockets to use for the target for all sessions using it. In other words: maximum number of Z39.50 session to the target.

4.9.7 keepalive

The `keepalive` element holds information about the keepalive Z39.50 sessions. Keepalive sessions are proxy-to-backend sessions that is no longer associated with a client session.

The `keepalive` element which is the child of the `target`holds two elements: `bandwidth` and `pdu`. The `bandwidth` is the maximum total bytes transferred to/from the target. If a target session exceeds this limit, it is shut down (and no longer kept alive). The `pdu` is the maximum number of requests sent to the target. If a target session exceeds this limit, it is shut down. The idea of these two limits is that avoid very long sessions that use resources in a backend (that leaks!).

The following sets maximum number of bytes transferred in a target session to 1 MB and maximum of requests to 400.

```
<keepalive>
 <bandwidth>1048576</bandwidth>
 <pdu>400</pdu>
</keepalive>
```

4.9.8 limit

The `limit` section specifies bandwidth/pdu requests limits for an active session. The proxy records bandwidth/pdu requests during the last 60 seconds (1 minute). The `limit` may include the elements `bandwidth`, `pdu`, `retrieve` and `search`. The `bandwidth` measures the number of bytes transferred within the last minute. The `pdu` is the number of requests in the last minute. The `retrieve` holds the maximum records to which may be retrieved in one Present Request. The `search` is the maximum number of searches within the last minute.

If a bandwidth/pdu/search limit is reached the proxy will postpone the requests to the target and wait one or more seconds. The idea of the limit is to ensure that clients that downloads hundreds or thousands of records do not hurt other users.

The following sets maximum number of bytes transferred per minute to 500Kbytes, maximum number of records retrievals to 40 and maximum number of searches to 20.

```
<limit>
 <bandwidth>524288</bandwidth>
 <retrieve>40</retrieve>
 <search>20</search>
</limit>
```

Note
Typically the values in the keepalive section are mugh higher than their equivalent limit counterparts (bandwidth, pdu).

4.9.9 attribute

The `attribute` element specifies accept or reject or a particular attribute type, value pair. Well-behaving targets will reject unsupported attributes on their own. This feature is useful for targets that do not gracefully handle unsupported attributes.

Attribute elements may be repeated. The proxy inspects the attribute specifications in the order as specified in the configuration file. When a given attribute specification matches a given attribute list in a query, the proxy takes appropriate action (reject, accept).

If no attribute specifications matches the attribute list in a query, it is accepted.

The `attribute` element has two required attributes: `type` which is the Attribute Type-1 type, and `value` which is the Attribute Type-1 value. The special value/type `*` matches any attribute type/value. A value may also be specified as a list with each value separated by comma, a value may also be specified as a list: low value - dash - high value.

If attribute `error` is given, that holds a Bib-1 diagnostic which is sent to the client if the particular type, value is part of a query.

If attribute `error` is not given, the attribute type, value is accepted and passed to the backend target.

A target that supports use attributes 1,4, 1000 through 1003 and no other use attributes, could use the following rules:

```
<attribute type="1" value="1,4,1000-1003"/>
<attribute type="1" value="*" error="114"/>
```

4.9.10 syntax

The `syntax` element specifies accept or reject or a particular record syntax request from the client. It also allows record conversion of XML records via XSLT.

The `syntax` has one required attribute: `type` which is the Preferred Record Syntax.

If attribute `error` is given, that holds a Bib-1 diagnostic which is sent to the client if the particular record syntax is part of a present - or search request.

If attribute `error` is not given, the record syntax is accepted and passed to the backend target.

If attribute `marcxml` is given, the proxy will perform MARC21 to MARCXML conversion. In this case the `type` should be XML. The proxy will use preferred record syntax USMARC/MARC21 or `backendtype` (if given) against the backend target. For the special case where `backendtype` is `opac` the proxy will convert the OPAC record to OPACXML.

When `marcxml` is used, yazproxy assumes that records retrieved from the backend are encoded in the MARC-8 character set. This is correct for most MARC21 based systems, but not for other MARC variants or UTF-8 based MARC21 systems. The `backendcharset` attribute specifies the character set of the MARC records to be converted.

If attribute `backendtype` is given, that holds the record syntax to be transmitted to backend.

If attribute `backendelementset` is given, that holds elementset to be transmitted to backend. An empty value of `backendelementset` has the effect of omitting any Comp-Spec (and elementset) sent to backend.

If `backendelementset` is omitted, the element set from client is used, except if `marcxml` is used. In that case (using `marcxml`), no Comp-Spec and no elementset is sent to backend.

If attribute `stylesheet` is given, the proxy will convert XML record from server via XSLT. It is important that the content from server is XML. If used in conjunction with attribute `marcxml`, the MARC to MARCXML/OPACXML conversion takes place before the XSLT conversion takes place.

If attribute `identifier` is given that is the SRU record schema identifier for the resulting output record (after MARCXML and/or XSLT conversion).

If sub element `title` is given (as child element of `syntax`, then that is the official SRU name of the resulting record schema.

If sub element `name` is given that is an alias for the record schema identifier. Multiple `names` may be specified.

Example 4.1 MARCXML conversion

To accept USMARC and offer MARCXML XML plus Dublin Core (via XSLT conversion) but the following configuration could be used:

```
<proxy>
<target name="mytarget">
..
<syntax type="usmarc"/>
<syntax type="xml" marcxml="1"
  identifier="info:srw/schema/1/marcxml-v1.1"
  <title>MARCXML<title>
  <name>marcxml<name>
</syntax>
<syntax type="xml" marcxml="1" stylesheet="MARC21slim2SRWDC.xsl"
  identifier="info:srw/schema/1/dc-v1.1">
  <title>Dublin Core<title>
  <name>dc<name>
</syntax>
<syntax type="*" error="238"/>
..
</target>
</proxy>
```

4.9.11 explain

The `explain` element includes Explain information for SRU about the server in the target section. This information must have a `serverInfo` element with a database that this target must be available as (URL path). For example,

```
<explain xmlns="http://explain.z3950.org/dtd/2.0/">
  <serverInfo>
    <host>myhost.org</host>
    <port>8000</port>
    <database>mydatabase</database>
  </serverInfo>
```

```
      <!-- remaining Explain stuff -->
    </explain>
```

In the above case, the SRU service is available as `http://myhost.org:8000/mydatabase`.

4.9.12 cql2rpn

The content of the `cql2rpn` element specifies the path from the working directory to a CQL-to-RPN conversion file for the server in the target section. This element is required for SRU searches to operate against Z39.50 servers that don't support CQL. Most Z39.50 servers only support Type-1/RPN so this is usually required.

See YAZ documentation for more information about the CQL to PQF conversion. See also the `pqf.properties` in the `etc` (or *`prefix/share/yazproxy`*) directory of the YAZ proxy distribution.

4.9.13 preinit

The element `preinit` is the child of element `target` and specifies the number of spare connection to a target. By default no spare connection are created by the proxy. If the proxy uses a target exclusive or a lot, the preinit session will ensure that target sessions have been made before the client makes a connection and will therefore reduce the connect-init handshake dramatically. Never set this to more than 5.

4.9.14 target-authentication

The element `target-authentication` specifies fixed authentication information to be sent to the backend target.

This element takes a an attribute `type` which is the authenticatin type to be used..

`none` No authentication. There is no CDATA associated with this.

`anonymous` Anonymous authentication. There is no CDATA associated with this.

`open` Open authentication. The CDATA consists of the open authentication string.

`idPass` IdPass authentication. The CDATA consists of three terms: user, group and password.

4.9.15 target-charset

The element `target-charset` specifies the native character set that the target uses for queries.

If this is specified the proxy will act as a Z39.50 server supporting character set negotiation. And in SRU mode it will convert from UTF-8 (UNICODE) to this native character set (if possible).

4.9.16 max-clients

The element `max-clients` is the child of element `proxy` and specifies the total number of allowed connections to targets (all targets). If this limit is reached the proxy will close the least recently used connection.

Note, that many Unix systems impose a system on the number of open files allowed in a single process, typically in the range 256 (Solaris) to 1024 (Linux). The proxy uses 2 sockets per session + a few files for logging. As a rule of thumb, ensure that 2*max-clients + 5 can be opened by the proxy process.

Tip
Using the bash shell, you can set the limit with `ulimit -n`*no*. Use `ulimit -a` to display limits.

4.9.17 log

The element `log` is the child of element `proxy` and specifies what to be logged by the proxy.

Specify the log file with command-line option `-l`.

The text of the `log` element is a sequence of options separated by white space. See the table below:

Option	Description
`client-apdu`	Log APDUs as reported by YAZ for the communication between the client and the proxy. This facility is equivalent to the APDU logging that happens when using option `-a`, however this tells the proxy to log in the same file as given by `-l`.
`server-apdu`	Log APDUs as reported by YAZ for the communication between the proxy and the server (backend).
`clients-requests`	Log a brief description about requests transferred between the client and the proxy. The name of the request and the size of the APDU is logged.
`server-requests`	Log a brief description about requests transferred between the proxy and the server (backend). The name of the request and the size of the APDU is logged.
`client-ip`	Log the client IP for each log entry. By default, the client IP is only logged when a new session starts.

Table 4.1: Logging options

To log communication in details between the proxy and the backend, th following configuration could be used:

```
<target name="mytarget">
 <log>server-apdu server-requests</log>
</target>
```

4.9.18 max-connect

The element `max-connect` is a child of element `proxy` and specifies the maximum number of connections to be initiated within the last minute (or value of period-connect.

If the maximum number is reached the proxy will terminate the just initiated session (connection terminated).

4.9.19 limit-connect

The element `max-connect` is a child of element `proxy` and specifies the limit of number of connections to be initiated within the last minute (or value of period-connect.

If the maximum number is reached the proxy delays the first operation in the session by one second.

4.9.20 period-connect

The element `period-connect` is a child of element `proxy` and specifies period - in the number of seconds that limit-connect and max-connect should measure connections.

If `period-connect` is omitted, 60 seconds is used.

4.9.21 docpath

The element `docpath` is a child of element `proxy` and specifies an allowed HTTP path for local file access. Using `docpath` the proxy may return static file content.

The value of docpath both serves as a HTTP path prefix *and* as a local file prefix. If a value of `etc` is used only URLs with the prefix `/etc/` results in a local file access to the directory `etc` within the working directory of yazproxy.

Note

Care has been taken to ensure that hostile URLs are rejected - including strings such as `..` and `/` (absolute file system access).

4.10 Proxy Manual Pages

4.10.1 yazproxy

yazproxy — The YAZ toolkit's transparent Z39.50/SRU proxy

Synopsis

`yazproxy` [-a *`filename`*] [-c *`config`*] [-D] [-i *`seconds`*] [-l *`filename`*] [-m *`num`*] [-n *`num`*] [-o *`level`*] [-t *`target`*] [-p *`pidfile`*] [-T *`seconds`*] [-u *`userid`*] [-v *`level`*] [-V] [-X] *`host`*:*`port`*

DESCRIPTION

yazproxy is a proxy that accepts connections from Z39.50/SRU clients and contacts a Z39.50 backend. The listening port must be specified on the command-line. **inetd** operation is not supported. The *host*:*port* argument specifies host address to listen to, and the port to listen on. Use the host `@` to listen for connections coming from any address.

yazproxy can be configured using command-line options or a configuration file. Configuration file options override values specified on the command-line.

yazproxy rereads its configuration file and reopens log files when it receives the hangup signal, SIGHUP.

OPTIONS

-a *filename* Specifies the name of a file to which to write a log of the APDUs (protocol packets) that pass through the proxy. The special filename − may be used to indicate standard output.

-c *config* Specifies config filename. Configuration is in XML and is only supported if the YAZ proxy is compiled with libxml2.

-D Puts YAZ proxy in the background after startup. This is similar to using shell's & operator but often better since it allows the start / stop script to capture startup errors.

-i *seconds* Specifies in seconds the idle time for communication between client and proxy. If a connection is inactive for this long it will be closed. Default: 600 seconds (10 minutes).

-l *filename* Specifies the name of a file to which to write a log of the YAZ proxy activity. This uses the logging facility as provided by the YAZ toolkit. If this options is omitted, the output directed to stderr.

-m *num* Specifies the maximum number of client connections to be offered [default 150].

-n *num* Sets maximum number of open files to *num*. This is only available on systems that offers the `setrlimit(2)` call.

-o *level* Sets level for optimization. Use zero to disable; non-zero to enable. Handling for this is not fully implemented; we will probably use a bit mask to enable/disable specific features. By default optimization is enabled (value 1).

-p *pidfile* When specified, yazproxy will create *pidfile* with the process ID of the proxy. The pidfile will be generated before the process changes identity (see option `-u`).

-t *target* Specifies the default backend target to use when a client connects that does not explicitly specify a target in its `initRequest`.

-T *seconds* Specifies in seconds the idle time for communication between proxy and backend target. If a connection is inactive for this long it will be closed. Default: 600 seconds (10 minutes).

-u *userid* When specified, yazproxy will change identity to the user ID specified, just after the proxy has started listening to a possibly privileged port and after the PID file has been created if specified by option `-u`.

-v *level* Sets the logging level. *level* is a comma-separated list of members of the set {fatal,debug,warn,lo

-V Displays yazproxy version and exits with status code 0. Should not be used in conjunction with other options.

-X Enables debugging mode for the proxy. When specified, the proxy will not fork itself, thus any violations becomes fatal. Useful if you run yazproxy inside gdb. Don't run this in production.

EXAMPLES

The following command starts the proxy, listening on port 9000, with its default backend target set to Index Data's test server:

```
$ yazproxy -t indexdata.dk:210 @:9000
```

You can connect to the proxy via yaz-client as follows:

```
$ ./yaz-client  localhost:9000/gils
Connecting...OK.
Sent initrequest.
Connection accepted by v3 target.
ID     : 81
Name   : Zebra Information Server/GFS/YAZ (YAZ Proxy)
Version: Zebra 1.3.15/1.23/2.0.19
Options: search present delSet scan sort extendedServices namedResultSets
Elapsed: 0.152108
Z> f computer
Sent searchRequest.
Received SearchResponse.
Search was a success.
Number of hits: 3, setno 1
SearchResult-1: computer(3)
records returned: 0
Elapsed: 0.172533
```

The YAZ command-line client, **yaz-client**, allows you to set the proxy address by specifying option -p. In that case, the actual backend target is specified as part of the Initialize Request.

Suppose the proxy running on localhost, port 9000. To connect to British Library's server at blpcz.bl.uk:2102 use:

```
yaz-client -p localhost:9000 blpcz.bl.uk:21021/BLPC-ALL
```

4.11 OtherInformation Encoding

The proxy uses the OtherInformation definition to carry information about the target address and cookie.

```
  OtherInformation   ::= [201] IMPLICIT SEQUENCE OF SEQUENCE{
    category           [1]   IMPLICIT InfoCategory OPTIONAL,
    information        CHOICE{
      characterInfo            [2]  IMPLICIT InternationalString,
      binaryInfo               [3]  IMPLICIT OCTET STRING,
      externallyDefinedInfo    [4]  IMPLICIT EXTERNAL,
      oid                      [5]  IMPLICIT OBJECT IDENTIFIER}}
--
  InfoCategory ::= SEQUENCE{
      categoryTypeId   [1]   IMPLICIT OBJECT IDENTIFIER OPTIONAL,
      categoryValue    [2]   IMPLICIT INTEGER}
```

The `categoryTypeId` is either OID 1.2.840.10003.10.1000.81.1, 1.2.840.10003.10.1000.81.2 for proxy target and proxy cookie respectively. The `categoryValue` is set to 1. The value proxy and cookie is stored in element `characterInfo` of the `information` choice.

4.12 YAZ Proxy Configuration Schema

Here an XML Schema for the YAZ proxy configuration file. The schema, `yazproxy.xsd` is located in sub directory `etc` of the distribution.

```
<?xml version="1.0"?>
<!-- XML Schema for YAZ proxy config file.
-->
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:exp="http://explain.z3950.org/dtd/2.0/"
  xmlns="http://indexdata.dk/yazproxy/schema/0.9/"
  targetNamespace="http://indexdata.dk/yazproxy/schema/0.9/"
  >
 <xs:import namespace="http://explain.z3950.org/dtd/2.0/"
      schemaLocation="zeerex-2.0.xsd"/>
 <xs:element name="proxy">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="target" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="max-clients" minOccurs="0"/>
    <xs:element ref="log" minOccurs="0"/>
    <xs:element ref="module" minOccurs="0"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>

 <xs:element name="target">
  <xs:complexType>
   <xs:sequence>
     <xs:element ref="url" minOccurs="0" maxOccurs="unbounded"/>
     <xs:element ref="target-timeout" minOccurs="0"/>
```

```
    <xs:element ref="client-timeout" minOccurs="0"/>
    <xs:element ref="max-sockets" minOccurs="0"/>
    <xs:element ref="keepalive" minOccurs="0"/>
    <xs:element ref="limit" minOccurs="0"/>
    <xs:element ref="attribute" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="syntax" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="preinit" minOccurs="0"/>
    <xs:element ref="exp:explain" minOccurs="0"/>
    <xs:element ref="cql2rpn" minOccurs="0"/>
    <xs:element ref="target-authentication" minOccurs="0"/>
    <xs:element ref="client-authentication" minOccurs="0"/>
    <xs:element ref="negotiation-charset" minOccurs="0"/>
    <xs:element ref="negotiation-lang" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="default" type="xs:string" use="optional"/>
  <xs:attribute name="name" type="xs:string"/>
  <xs:attribute name="database" type="xs:string"/>
 </xs:complexType>
</xs:element>

<xs:element name="url" type="xs:string"/>
<xs:element name="target-timeout" type="xs:integer"/>
<xs:element name="client-timeout" type="xs:integer"/>
<xs:element name="max-sockets" type="xs:integer"/>
<xs:element name="bandwidth" type="xs:integer"/>
<xs:element name="pdu" type="xs:integer"/>
<xs:element name="retrieve" type="xs:integer"/>
<xs:element name="preinit" type="xs:integer"/>
<xs:element name="cql2rpn" type="xs:string"/>
<xs:element name="target-authentication">
  <xs:complexType>
   <xs:simpleContent>
     <xs:extension base="xs:string">
       <xs:attribute name="type" type="xs:string"/>
     </xs:extension>
   </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name="client-authentication">
  <xs:complexType>
   <xs:simpleContent>
     <xs:extension base="xs:string">
       <xs:attribute name="module" type="xs:string"/>
       <xs:attribute name="args" type="xs:string"/>
     </xs:extension>
   </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="negotiation-charset" type="xs:string"/>
<xs:element name="negotiation-lang" type="xs:string"/>

<xs:element name="keepalive">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="bandwidth" minOccurs="0"/>
   <xs:element ref="pdu" minOccurs="0"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="limit">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="bandwidth" minOccurs="0"/>
   <xs:element ref="pdu" minOccurs="0"/>
   <xs:element ref="retrieve" minOccurs="0"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>

<xs:element name="attribute">
 <xs:complexType>
  <xs:attribute name="type" type="xs:string"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="error" type="xs:integer"/>
 </xs:complexType>
</xs:element>

<xs:element name="syntax">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="title" minOccurs="0"/>
   <xs:element ref="name" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="error" type="xs:string" />
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="marcxml" type="xs:string" />
  <xs:attribute name="identifier" type="xs:string" />
  <xs:attribute name="stylesheet" type="xs:string" />
  <xs:attribute name="backendtype" type="xs:string" />
  <xs:attribute name="backendcharset" type="xs:string" />
  <xs:attribute name="usemarconstage1" type="xs:string" />
  <xs:attribute name="usemarconstage2" type="xs:string" />
  <xs:attribute name="backendelementset" type="xs:string" />
 </xs:complexType>
</xs:element>

<xs:element name="title" type="xs:string"/>
<xs:element name="name" type="xs:string"/>
```

```
 <xs:element name="max-clients" type="xs:integer"/>
 <xs:element name="log" type="xs:string"/>
 <xs:element name="module" type="xs:string"/>

</xs:schema>
```

Appendix A

License

A.1 GPL

YAZ Proxy, Copyright © 1998-2019 Index Data.

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

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

You should have received a copy of the GNU General Public License along with YAZ Proxy; see the file LICENSE.proxy. If not, write to the Free Software Foundation, 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

```
        GNU GENERAL PUBLIC LICENSE
           Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
                       51 Franklin St, Fifth Floor, Boston, MA  02110-1301  ↩
                           USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

          Preamble

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

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

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

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

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

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

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

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

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

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in

```
the term "modification".)  Each licensee is addressed as "you".

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

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

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

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

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

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

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

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

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

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

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

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

    a) Accompany it with the complete corresponding machine-readable
    source code, which must be distributed under the terms of Sections
    1 and 2 above on a medium customarily used for software interchange; or ↩
       ,

    b) Accompany it with a written offer, valid for at least three
    years, to give any third party, for a charge no more than your
    cost of physically performing source distribution, a complete
    machine-readable copy of the corresponding source code, to be
    distributed under the terms of Sections 1 and 2 above on a medium
    customarily used for software interchange; or,

    c) Accompany it with the information you received as to the offer
    to distribute corresponding source code.  (This alternative is
    allowed only for noncommercial distribution and only if you
    received the program in object code or executable form with such
    an offer, in accord with Subsection b above.)

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

If distribution of executable or object code is made by offering
access to copy from a designated place, then offering equivalent
access to copy the source code from the same place counts as
```

```
distribution of the source code, even though third parties are not
compelled to copy the source along with the object code.

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

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

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

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

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

It is not the purpose of this section to induce you to infringe any
patents or other property right claims or to contest validity of any
such claims; this section has the sole purpose of protecting the
integrity of the free software distribution system, which is
```

implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

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

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

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

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

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

            NO WARRANTY

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

```
REPAIR OR CORRECTION.

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

         END OF TERMS AND CONDITIONS
```