

NNTPSwitch Manual

Tommy van Leeuwen

tommy@nntpswitch.org

\$Revision: 1.4 \$

Implementation Guidelines and Configuration Manual for NNTPSwitch.

Table of Contents

1. What is NNTPSwitch?	1
2. Background	1
3. Installing NNTPSwitch.....	2
4. Statistics	3
5. Global Configuration (nntpswitch.conf)	4
6. Backend Server Configuration (servers.conf)	5
7. Client Access Configuration (access.conf)	7
8. Authorization and Accounting Modules	9
9. NNTP Protocol Differences	12
10. Further Info	12
11. License.....	12

1. What is NNTPSwitch?

NNTPSwitch is a NNRP content router. It can pass client NNRP streams to backend servers. Each backend server can be running a set of newsgroups. NNTPSwitch is especially designed for speed and flexibility. It can handle hundreds of client connections simultaneously. Authentication subsystems can be switched on the fly. Postings support cleanfeed or other spam filtering. NNRP actually is the reader part of commands from the NNTP protocol.

NNTPSwitch is in beta progress in terms of configuration, documentation and general code building. Stability is absolutely perfect and a number one priority during development. Currently NNTPSwitch runs production stable at *News-Service.com* (<http://www.news-service.com>) which is a popular european news service.

2. Background

Clients connect with their favorite newsreader to NNTPSwitch. The client and nntpswitch perform authentication and optionally the client gets some active or newsgroup file lists. When the client sends a command which requires a newsgroup itself, nntpswitch connects to a backend spool server. The list of which groups belong to which server are specified in the nntpswitch specific active file format. If the client performs a

command inside a newsgroup then actually nntpswitch sends that command to a backend server and sends all data from that backend server directly back to the client. Hardly any processing is done with data from a backend server passed back to the client.

Posts work differently in the fact that nntpswitch handles them itself. After a client has posted something nntpswitch creates a separate connection to a posting server and sends the article with the IHAVE command. If the article is posted in a moderated group nntpswitch emails the message to the moderator. Also by default nobody can post approved postings. Ip addresses of people with approved posting rights should be listed with the apost option in the access.conf.

No data besides the active file, which is generated by updategroups, is cached or stored to disk with the assumption that a backend spool server will always be faster than a frontend machine. Or better, the backend servers *should* take most of the load. Backend servers can be load-balanced or made redundant more easily than frontend servers. Backend servers are good at storing articles and overview data while frontend servers take all the authentication and accounting stuff.

3. Installing NNTPSwitch

3.1. System Requirements and Planning

A Linux system is required. Portability is not a priority at the moment. To estimate system requirements you need to know the maximum number of concurrent users you will have. A p3/1.3ghz can handle about 1500 concurrent broadband connections doing about 250 mbps. If you have dialup users you'll probably can handle 3000 concurrent connections. Memory footprint is average to high. NNTPSwitch needs about 350k for each client and a shared space of about 20mb for roughly 65000 groups (35mb for 100k groups).

3.2. Installing Programs and Modules

You should have Perl installed in your PATH.

```
make
make install
```

make calls make config automatically. make config parses the nntpswitch.conf.in file and uses these values as defaults.

Create a directory /etc/nntpswitch or /usr/local/etc/nntpswitch (or whatever else is specified in config.c) and copy nntpswitch.conf, etc/access.conf etc/groups.conf etc/auth.conf and etc/overview.fmt to your /etc/nntpswitch directory.

```
mkdir /etc/nntpswitch
cp nntpswitch.conf-dist /etc/nntpswitch/nntpswitch.conf
cp etc/overview.fmt /etc/nntpswitch/
cp etc/access.conf /etc/nntpswitch/
cp etc/auth.conf /etc/nntpswitch/
cp etc/profiles.conf /etc/nntpswitch/
```

Edit nntpswitch.conf, access.conf and groups.conf to match your configuration. If you're finished run updategroups for the first time to retrieve the active, newsgroups and active.times from the backend servers.

You need to have Time::ParseDate installed in order for the Perl statistics script to work.

Once you have build a server configuration you can test the configuration by running the updategroups command:

```
/usr/local/sbin/updategroups
```

To test all AAA configuration run nntpswitch in test mode:

```
/usr/local/sbin/nntpswitchd -t
```

If it's ok you can start it up with no parameters or use the init.d script.

3.3. Installing extra authentication and accounting modules

To install all available extra modules (currently only Radius, PostgreSQL and MySQL) type:

```
make more
make install
```

3.3.1. Radius modules

The radius authorization and accounting modules depend on FreeBSD/Juniper libradius. The freeradius libradius does not work. To make only the radius module type:

```
make radius
make install
```

Download *libradius for Linux* (<http://portal-to-web.de/tacacs/libradius.php>)

3.3.2. PostgreSQL and MySQL modules

The postgres module depends on libpq-fe. Mysql module depends on libmysqlclient. Both should be installed by your database development package.

To install one or both type:

```
make radius
make mysql
make install
```

3.4. Daily Operation

You should run statistics, rotate your logfiles, and update newsgroups lists at least once a day. It's recommended you run the nsstats.sh shell script from cron every 24hr. The nsstats.sh does the following things:

- Rotate news.debug logfile by one.
- Retrieve and merge latest active and newsgroups information using the updategroups command.
- Create statistics using nsstats.pl
- Execute any nsstats.sh.local script if found.

4. Statistics

NNTPSwitch can create two kind of statistics. A detailed usage report of the server is created everyday. Also realtime profile usage statistics are available using rrd.

4.1. Daily Usage Reports

Daily usage reports are usually generated from the nsstats.sh shell script every night. However you might create intermediate reports using a command like this:

```
nsstats.pl -d /html/www/dir < news.debug
```

Statistics are normally generated from the nsstats.sh maintenance script. The news.debug log can get rather large on bigger sites, to make sure not too much is being logged, turn LogCommands, LogWriteclient, LogReadserver and LogWriteserver off.

4.2. RRD Statistics

Realtime RRD profile usage statistics can be plotted using RRDBrowse. You should use RRDBrowse 1.7 or one of the latest cvs versions for it to work.

Your RRD host should have stats access in an ACL, for example:

```
192.168.1.1/32 read,post,stats
```

For each profile you should create 3 nfo files. One for the number of active/connected users and connections, one for the bandwidth and one for the number of articles per second.

Example of Users nfo:

```
Type: ntsprofile
Target: 119@your.server
Profile: default
Description: Default Profile Users
```

Example of Bandwidth nfo:

```
Type: ntsprofbw
Target: 119@your.server
Profile: default
Description: Default Profile Bandwidth
```

Example of Articles nfo:

```
Type: ntsprofarts
Target: 119@your.server
Profile: default
Description: Default Profile Articles per Second
```

RRDBrowse is available at [rrdbrowse.org](http://www.rrdbrowse.org) (<http://www.rrdbrowse.org>). See the *RRDBrowse Readme* (<http://www.rrdbrowse.org/README>) or *RRDBrowse Modules Readme* (<http://www.rrdbrowse.org/readmemodules.html>) for more details on RRDBrowse.

5. Global Configuration (nntpswitch.conf)

This chapter will describe the system-wide nntpswitch configuration. This configuration is stored in `nntpswitch.conf` and it handles all the global parameters like the host/port to listen on, paths, logging and that kind of things.

- `DenyNoIndexField` If this parameter is set to 1 then XPAT/XHDR searches are denied for headers which are not in the `overview.fmt`. A setting of 0 allows searches in all headers

INN alike servers store real overview data, only the headers from `overview.fmt` are stored in the overview database. Diablo alike servers don't have a real overview database, instead Diablo stores *all* headers for each article and retrieves whatever headers the client want. So, for Diablo it doesn't matter which fields are searched, Diablo stores them all anyway. INN however has to search the article spool in case someone searches for a header which is not stored in the overview database. Like expected, this has an enormous impact on system performance! Be aware of that. If you have INN backend servers set this option to 1, if you have Diablo you can set this option to 0.

- `DropUnknown` This parameter is the number of unknown commands received from a client before a forced disconnect. Some clients are really sending a lot of garbage every now and then.
- `DownDelay` Backoff this many seconds before sending `Remote Server Unavailable` to a client when a server is down.
- `RetryDelay` Backoff this many seconds before sending `Too many concurrent connections` to a client when he reaches one of his limits.
- `LogCommands` Turn on logging of each command and its parameters. If you leave it off a summary of the command usage is printed when the client disconnects.
- `LogWriteclient` Log everything written to the client (except articles, headers, overview, etc). To log everything read from the client use `LogCommands` instead.
- `LogReadserver` and `LogWriteserver` Log everything read from or written to a backend server.

6. Backend Server Configuration (servers.conf)

All backend servers are configured in the `servers.conf`. As with most other configuration file there is one special server which is called `default`. The default server contains no `Hostname` or `Groups` statements, but contains all other possibly needed parameters.

The following example demonstrates the most basic configuration. A binaries server and a text server. The `textserver` also has the descriptions and knows about `active.times`.

```
server default
  Port          119
  Timeout       300
  Descriptions  False
  ActiveTimes   False
  Type          Spool
  Policy        Single
  Level         1
  SplitList     False
end

server bunnies
```

```

Groups      *bin*
Hostname    bin.example.com
Level       100
end

server tekst
Groups      *
Hostname    text.example.com
Level       999
Descriptions True
ActiveTimes True
end

```

Please make sure that you retrieve `active.times` and `newsgroups` information from atleast one server. This must be done by setting the options `activetimes` and `descriptions` to true. Also note that the most significant servers are listed at the bottom and must have the highest level. If you have servers for single newsgroups or patterns thereof put them at the top and give them a lower level.

- `SplitList` If you have a backend server which doesn't support the LIST ACTIVE command with multiple patterns, set this option to True. For Diablo put it to 1, for INN put it to 0.

6.1. Backup and Load Balancing Levels and Policies

You can configure backup or balanced servers by using the same `Level` and same `Groups` values. `Policy` can be either `backup` or `balance`. Use a separate level for each set of groups. The level must be between 1 and 1000.

The backup and balancing checks take place upon connecting to a group. Only when a group is selected `nntpswitch` checks if we need to (re)connect to a different backend server. A backup server is used when the primary server fails, failure is tested on every connect and the primary server will be retried first at all times. The balancing takes place by just selecting a different server on each connect. Only if one of the balancing servers failes, `nntpswitch` just uses the backup mechanism to find another server.

An example of 2 *backup* servers. Each connection will try "movies1" first. In case of some connection or NNTP error, the next server "movies2" is tried.

```

server movies2
  Hostname    movies-2.example.com
  Groups      alt.binaries.movies*,alt.binaries.multimedia*
  Policy      Backup
  Level       100
end

server movies1
  Hostname    movies-1.example.com
  Groups      alt.binaries.movies*,alt.binaries.multimedia*
  Policy      Backup
  Level       100
end

```

An example of 2 *load-balanced* servers. Each connection to the backend server will connect to the other one.

```

server movies2
  Hostname    movies-2.example.com
  Groups      alt.binaries.movies*,alt.binaries.multimedia*
  Policy      Balance
  Level       100

```

```

end

server movies1
  Hostname      movies-1.example.com
  Groups        alt.binaries.movies*,alt.binaries.multimedia*
  Policy        Balance
  Level         100
end

```

Note in the above examples, the server “movies1” is most likely used for updating the groups with the `updategroups` command.

7. Client Access Configuration (access.conf)

All authentication, accounting and user profiles must be configured in `access.conf`. NNTPSwitch uses several kinds of configuration objects. Wildmats, Aliases, ACL's, Authenticators and Profiles. When a client connects, it's IP address is matched against an ACL and assigned a profile accordingly. If a client authenticates, the username is matched against the Authenticator string and, if successfully authenticated, assigned the profile from the authenticator. Upon connecting a client its ip address is matched against an ACL. Based on this ACL, a Profile is assigned to the client. A profile contains among other things, configuration options like the maximum number of clients for that profile and patterns for group based authentication.

If a client authenticates the username is looked up in one of the Authenticators. Based on the Authenticator two Aliases are applied to the authentication information from the client. One Alias specifies how to authenticate a client and one Alias specifies how to process accounting traffic for that user.

7.1. ACLs or IP Based Access Control

ACLs are configured in the `access.conf` file. The format is as follows:

```

CIDR/Address      Options      Profile

```

CIDR/Address defines the ip address or ip range in CIDR format. Examples are `127.0.0.1/32` or `192.168.1.0/24`.

The Options are one of `read`, `post`, `apost`, `nolimit`, `auth` or `deny`. `Read` and `post` are pretty obvious. `apost` specifies a client is allowed an `Approved:` header in hist posts. If `apost` is not specified the `Approved:` header is automatically stripped. `Nolimit` option specifies that the `MaxConcurrent` option from the Profile is not applied, the client can have as many connections as possible. When the `auth` keyword is specified it enforces client authentication. The client must authenticate before using privileged commands. Finally if `deny` is specified the client doesn't have any access at all.

The Profile keyword defines the name of the profile applied to the connecting client.

Always, at least the catch-all `0/0` address should be defined. Examples (choose only *one* of them):

```

acl CIDR/Address Options      Profile
acl 0/0          deny          # deny everything (profile "default")
acl 0/0          read,post,auth # force auth, read/post after auth (profile "default")
acl 0/0          read          local    # read only profile "local"

```

7.2. Aliases

Aliases specify alternate names for utility library functions and its arguments. Aliases are specified in `profiles.conf` like:

```
alias name          library.so          arguments
```

Name is just a short name from which you can reference to within the Authenticators configuration. Library is one of the included authentication or accounting libraries. Depending on the library a different set of arguments is required. Examples include:

```
alias CustomerOne   auth_passwd.so      /etc/nntpswitch/passwd.customer
alias RemoteAuth    auth_remote.so      news.news-service.com:119
alias AcctGeneric    acct_syslog.so      local4.info
```

Look for more examples at the descriptions of the different modules. See chapter Authorization-and-Accounting for details.

7.3. Authenticators

Authenticators are patterns which are matched to the username. Depending on the pattern an Alias for accounting and one for authorization can be specified. If the authentication succeeds the Profile is applied. All authenticators are specified in the file `auth.conf`.

```
auth pattern          auth-alias      acct-alias      profile
```

Pattern isn't really a pattern. A pattern has to start with an asterix(*) or has to end with one. Examples:

```
auth *@example.com    CustomerOne    AcctGeneric    ExampleCom
auth chiparus!*       AuthChiparus   AcctChiparus   ChiparusProfile
auth *                 # auth_default acct_default    default
```

Always leave the last line where the pattern is only an asterix. By default the aliases `auth_default` and `acct_default` are used with profile `default`.

7.4. Wildmats

Wildmats are just aliases for wildmat pattern so you can easily access them from the profile configuration. One pattern is matched for read access (the `group()` and `list` commands) and the other is matched for posting. If no wildmat names are used, the default is `all`, which matches `*`. Examples:

```
wildmat all           *
wildmat none          !*
wildmat nl            nl.*
wildmat comprec       comp.*,rec.*
wildmat nobin         *,!*bin*
```

7.5. Profiles

Profiles takes care of configuring a client, or rather a user. There is only one special profile which is called `default`. The default profile must have all options configured to a reasonably default setting.

Patterns `ReadPat` and `PostPat` define the wildmat names of the patterns applied to the group the client is selecting.

The variables `XComplaintsTo`, `Hostname`, `Organization`, `ForceOrganization` and `AddNNTPPostingHost` are used for posting only. This way you can have different headers for different users.

The `Banner` and `MaxConcurrent` variables only work when a profile is applied to an ACL and have no use when applied to an Authenticator. If you set `ReplaceBanner` to 1 all the NNTPSwitch versioning stuff and group/connection counters aren't displayed.

`MaxConnections` is the maximum number of connections for a profile. When a client logs in initially a profile is selected from an ACL. If the client logs in the profile is switched to one from an Authenticator. Only one profile (the currently selected) keeps track of the `MaxConnections`. Also the `MaxConnections` from `nntpswitch.conf` is global while the one from a profile only applies to that profile.

Additionally the profile specifies the `MaxSessionTime` and `MaxSessionBytes` which speak for themselves.

7.6. Setting user and connection limits

There are two things to keep in mind. A Connection (or port, or slot) is a single TCP session to the server. A User is a set of slots from the same hostname, or - if the user is authenticated - the username.

`MaxUsers` and `MaxConnections` are limits for the complete profile. Setting `MaxConnections` to 500 and `MaxUsers` to 400 will start to disconnect users either when it reaches 500 connections, or if there are 400 different users connected. Normally `MaxUsers` is the tunable parameter, we don't really care about the `MaxConnections` so you could just set `MaxConnections` high in the default profile.

`MaxHostConcurrent` and `MaxUserConcurrent` can basically both be the same value. If you want to allow different users connecting from the same ip address you should make `MaxHostConcurrent` atleast N times `MaxUserConcurrent`.

Rate-limits can and must be configured on a user basis. It is not possible to limit single connections which doesn't make much sense anyway. Either set `MaxUserMbit` or `MaxUserKbit` which both initialize `MaxUserBPS` actually. Each user will have a bandwidth limit of this amount.

Rate-limits for the complete profile can be set with almost the same options. `MaxProfileMbit`, `MaxProfileKbit` or `MaxProfileBPS`. Profile rate-limits will balance all the available bandwidth equally amongst all connections, not all users. So with profile rate-limits activated, users with more connections are faster than users with only 1 connection.

Important: `MaxUserBPS` and `MaxProfileBPS` are in *bytes per second* and `MaxUserMbit`, `MaxUserKbit`, `MaxProfileMbit` and `MaxProfileKbit` are in *bits per second*.

8. Authorization and Accounting Modules

8.1. No Accounting (`acct_none.so`)

This library takes no arguments because it does nothing.

8.2. Syslog Accounting (`acct_syslog.so`)

This library takes the syslog facility and/or priority as argument. Example:

```
alias syslog1 acct_syslog.so local3
alias syslog2 acct_syslog.so news.crit
alias syslog3 acct_syslog.so user.info
```

If no priority is specified then `.info` is used. If no facility is specified then `auth` is used. This means if you don't specify a facility/priority at all the default will be `auth.info`. In the above first example accounting data is logged to `local3.info`.

The client username is logged together with number of bytes downloaded, number of groups, number of articles, number of posts and postbytes. An example syslog accounting entry looks like this:

```
Mar 11 23:36:31 hostname nntpswitchd[13087]: accounting_entry: username 56427506 169 2 1 7620
```

This means the user `username` has downloaded 56MB in 169 articles. This user has joined 2 newsgroups and made 1 7.6kb post.

8.3. Passwordfile Authorization (`auth_passwd.so`)

The argument to this library is the name of the passwordfile. It's pretty stupid but it's plaintext. It contains a line of text which is should be OK or any other text which is passed to the client if the authorization is denied.

Example usage:

```
alias supportnet auth_passwd.so /etc/passwd.supportnet
```

The password file looks like:

```
guest:guest:No guest login allowed
test:secret:OK
john:g3helm:Account Disabled
whatever:password:OK
```

8.4. POP3 relay Authorization (`auth_pop3.so`)

This library can be used to proxy the authentication information to a POP3 server. The argument is the hostname of the server followed by the portnumber. Example:

```
alias support.net auth_pop3.so mail.support.net:110
```

The pattern from which this authentication module is matched is stripped from the username part. So if the user logs in initially with `"user@example"`, the username passed to the POP3 server will be only `"user"`. considering an Authenticator match like

```
*@example AuthExample AcctExample ExampleProfile
```

8.5. NNTP relay Authorization (`auth_remote.so`)

This is almost the same as the `pop3` authentication library except that this connects to a remote NNTP server instead of a POP3 server.

8.6. Ignore Authorization (auth_ignore.so)

The argument to this library can be “true” or anything else to make it “false”. If it’s false the authentication is denied, if it is true the authentication is accepted. Regardless of what the user entered for username and password.

8.7. Postgres Authorization (auth_postgres.so)

This library is for PostgreSQL authorization. The argument is a PostgreSQL connect string for example:

```
alias dbUsers auth_postgres.so hostaddr=192.168.1.1 dbname=users user=test
```

The pattern from which this authentication module is matched is stripped from the username part.

This module is quite simple and probably needs some hacking to suit your needs. By default it reads the fields “login” and “password” from a table named “users”. An example database layout looks like:

```
create table users (
    login      varchar(32),
    password   varchar(32)
);
```

8.8. MySQL Authorization (auth_mysql.so)

Look for MySQL notes at the PostgreSQL. Basically the connect string from postgres is emulated but the only supported fields are hostname, dbname, username and password. A sample alias looks like:

```
alias dbUsers auth_mysql.so hostname=192.168.1.1 dbname=users user=test password=test
```

The connect string can be separated by spaced, commas or collons. The database query is again quite simple and might need some hacking.

8.9. Radius Authorization (auth_radius.so)

The radius module obviously connects to a radius server. The hostname or ip address and the secret are the only two arguments to this module:

```
alias radUsers auth_radius.so radiusd1.example.com:l33ts3cr3t
```

This module was originally submitted by ADS of Voicenet, thanks!

8.10. Radius Accounting (acct_radius.so)

The specification of this module is the same as the radius authorization module:

```
alias radacct acct_radius.so radiuslog1.example.com:mysecret
```

Everything possibly interesting item is currently logged. At the moment only stop records are sent. Future versions will probably support Start and Intermediate records too.

This is a summary of attributes logged. Not all radius servers support logging of NAS-Port if the client is not first authenticated so that’s commented out by default in de source.

Table 1. Radius Accounting Parameters

<i>Framed-IP-Address</i>	The IP Address of the client
<i>Acct-Session-Time</i>	Total time in seconds this client was connected
<i>Acct-Input-Packets</i>	The number of articles downloaded.
<i>Acct-Input-Octets</i>	The number of bytes downloaded.
<i>Acct-Output-Packets</i>	The number of articles posted.
<i>Acct-Output-Octets</i>	The number of bytes posted.

9. NNTP Protocol Differences

9.1. XPAT and XHDR

By default the configuration option `DenyNoIndexField` is true which means that XPAT and XHDR searches for headers which are not in the `overview.fmt` will be disabled. To enable searches for all possible headers fields put this option to false (0).

Allowing searches for arbitrary headers mostly means the backend server has to traverse all articles in the spool looking for headers. Known headers can be looked up in the overview database instead of the spool. Traversing the spool can mean a drastic load on the backend server.

9.2. NEWNEWS

In short, NEWNEWS only works for single groups instead of wildmats. This is because of the same reason NNTPSwitch has to be connected to a backend server. Using wildmats could mean multiple server connections. Probably the full version of NEWNEWS will never be supported by NNTPSwitch.

10. Further Info

Take a look at the BUGS file which is more likely a TODO list for the whole project.

Need more info? *Contact info@nntpswitch.org* (<mailto:info@nntpswitch.org>)

NNTPSwitch Homepage (<http://www.nntpswitch.org>)

If you like it, get yourself a news account at *News-Service.com* (<http://www.news-service.com>)

NNTPSwitch is developed at news-service.com. It's a spare-time product and made open-source so others can enjoy it. Future plans include more authentication and accounting options as well as to be more compliant with new rfc's.

11. License

This product includes software developed by the Internet Systems Consortium (ISC) and its contributors. The wildmat routines are taken from INN.

Copyright (c) 2003, 2004, Tommy van Leeuwen

Portions Copyright (c) 1991-2003, The Internet Systems Consortium (ISC) and its contributors.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Tommy van Leeuwen nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.