libmarias3 Documentation

Release 3.1.2-d2dcab4

Andrew Hutchings

Jun 26, 2023

Contents

Release 3.1.2-d2dcab4

Date Jun 26, 2023

CHAPTER 1

Introduction

1.1 What is libMariaS3?

libMariaS3 is a lightweight library to connect to Amazon's S3 storage.

It is LGPL 2.1 licensed so that it is possible to use both with Open Source and Commercial applications. It is also designed to provided a relatively easy to use API.

1.2 Licensing

1.2.1 Documentation Content

The libMariaS3 Documentation is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

1.2.2 libMariaS3 License

libMariaS3 is licensed under the Lesser GNU General Public License, Version 2.1.

```
                     GNU LESSER GENERAL PUBLIC LICENSE
                      Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, 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.
```

(continues on next page)

(continued from previous page)

```
[This is the first released version of the Lesser GPL.  It also counts
 as the successor of the GNU Library Public License, version 2, hence
 the version number 2.1.]

                            Preamble

  The licenses for most software are designed to take away your
freedom to share and change it.  By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

  This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it.  You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

  When we speak of free software, we are referring to freedom of use,
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 and use pieces of
it in new free programs; and that you are informed that you can do
these things.

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

  For example, if you distribute copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave
you.  You must make sure that they, too, receive or can get the source
code.  If you link other code with the library, you must provide
complete object files to the recipients, so that they can relink them
with the library after making changes to the library and recompiling
it.  And you must show them these terms so they know their rights.

  We protect your rights with a two-step method: (1) we copyright the
library, and (2) we offer you this license, which gives you legal
permission to copy, distribute and/or modify the library.

  To protect each distributor, we want to make it very clear that
there is no warranty for the free library.  Also, if the library is
modified by someone else and passed on, the recipients should know
that what they have is not the original version, so that the original
author's reputation will not be affected by problems that might be
introduced by others.

  Finally, software patents pose a constant threat to the existence of
any free program.  We wish to make sure that a company cannot
effectively restrict the users of a free program by obtaining a
restrictive license from a patent holder.  Therefore, we insist that
any patent license obtained for a version of the library must be
consistent with the full freedom of use specified in this license.
```

(continues on next page)

(continued from previous page)

```
  Most GNU software, including some libraries, is covered by the
ordinary GNU General Public License.  This license, the GNU Lesser
General Public License, applies to certain designated libraries, and
is quite different from the ordinary General Public License.  We use
this license for certain libraries in order to permit linking those
libraries into non-free programs.

  When a program is linked with a library, whether statically or using
a shared library, the combination of the two is legally speaking a
combined work, a derivative of the original library.  The ordinary
General Public License therefore permits such linking only if the
entire combination fits its criteria of freedom.  The Lesser General
Public License permits more lax criteria for linking other code with
the library.

  We call this license the "Lesser" General Public License because it
does Less to protect the user's freedom than the ordinary General
Public License.  It also provides other free software developers Less
of an advantage over competing non-free programs.  These disadvantages
are the reason we use the ordinary General Public License for many
libraries.  However, the Lesser license provides advantages in certain
special circumstances.

  For example, on rare occasions, there may be a special need to
encourage the widest possible use of a certain library, so that it becomes
a de-facto standard.  To achieve this, non-free programs must be
allowed to use the library.  A more frequent case is that a free
library does the same job as widely used non-free libraries.  In this
case, there is little to gain by limiting the free library to free
software only, so we use the Lesser General Public License.

  In other cases, permission to use a particular library in non-free
programs enables a greater number of people to use a large body of
free software.  For example, permission to use the GNU C Library in
non-free programs enables many more people to use the whole GNU
operating system, as well as its variant, the GNU/Linux operating
system.

  Although the Lesser General Public License is Less protective of the
users' freedom, it does ensure that the user of a program that is
linked with the Library has the freedom and the wherewithal to run
that program using a modified version of the Library.

  The precise terms and conditions for copying, distribution and
modification follow.  Pay close attention to the difference between a
"work based on the library" and a "work that uses the library".  The
former contains code derived from the library, whereas the latter must
be combined with the library in order to run.

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

  A. This License Agreement applies to any software library or other
program which contains a notice placed by the copyright holder or
other authorized party saying it may be distributed under the terms of
this Lesser General Public License (also called "this License").
```

(continues on next page)

(continued from previous page)

```
Each licensee is addressed as "you".

  A "library" means a collection of software functions and/or data
prepared so as to be conveniently linked with application programs
(which use some of those functions and data) to form executables.

  The "Library", below, refers to any such software library or work
which has been distributed under these terms.  A "work based on the
Library" means either the Library or any derivative work under
copyright law: that is to say, a work containing the Library or a
portion of it, either verbatim or with modifications and/or translated
straightforwardly into another language.  (Hereinafter, translation is
included without limitation in the term "modification".)

  "Source code" for a work means the preferred form of the work for
making modifications to it.  For a library, 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 library.

  Activities other than copying, distribution and modification are not
covered by this License; they are outside its scope.  The act of
running a program using the Library is not restricted, and output from
such a program is covered only if its contents constitute a work based
on the Library (independent of the use of the Library in a tool for
writing it).  Whether that is true depends on what the Library does
and what the program that uses the Library does.

  A. You may copy and distribute verbatim copies of the Library's
complete 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 distribute a copy of this License along with the
Library.

  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.

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

    I) The modified work must itself be a software library.

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

    I) You must cause the whole of the work to be licensed at no
    charge to all third parties under the terms of this License.

    I) If a facility in the modified Library refers to a function or a
    table of data to be supplied by an application program that uses
    the facility, other than as an argument passed when the facility
    is invoked, then you must make a good faith effort to ensure that,
```

(continues on next page)

(continued from previous page)

```
    in the event an application does not supply such function or
    table, the facility still operates, and performs whatever part of
    its purpose remains meaningful.

    (For example, a function in a library to compute square roots has
    a purpose that is entirely well-defined independent of the
    application.  Therefore, Subsection 2d requires that any
    application-supplied function or table used by this function must
    be optional: if the application does not supply it, the square
    root function must still compute square roots.)

These requirements apply to the modified work as a whole.  If
identifiable sections of that work are not derived from the Library,
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 Library, 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 Library.

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

  A. You may opt to apply the terms of the ordinary GNU General Public
License instead of this License to a given copy of the Library.  To do
this, you must alter all the notices that refer to this License, so
that they refer to the ordinary GNU General Public License, version 2,
instead of to this License.  (If a newer version than version 2 of the
ordinary GNU General Public License has appeared, then you can specify
that version instead if you wish.)  Do not make any other change in
these notices.

  Once this change is made in a given copy, it is irreversible for
that copy, so the ordinary GNU General Public License applies to all
subsequent copies and derivative works made from that copy.

  This option is useful when you wish to copy part of the code of
the Library into a program that is not a library.

  A. You may copy and distribute the Library (or a portion or
derivative of it, under Section 2) in object code or executable form
under the terms of Sections 1 and 2 above provided that you 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.

  If distribution of object code is made by offering access to copy
from a designated place, then offering equivalent access to copy the
```

(continues on next page)

(continued from previous page)

```
source code from the same place satisfies the requirement to
distribute the source code, even though third parties are not
compelled to copy the source along with the object code.

  A. A program that contains no derivative of any portion of the
Library, but is designed to work with the Library by being compiled or
linked with it, is called a "work that uses the Library".  Such a
work, in isolation, is not a derivative work of the Library, and
therefore falls outside the scope of this License.

  However, linking a "work that uses the Library" with the Library
creates an executable that is a derivative of the Library (because it
contains portions of the Library), rather than a "work that uses the
library".  The executable is therefore covered by this License.
Section 6 states terms for distribution of such executables.

  When a "work that uses the Library" uses material from a header file
that is part of the Library, the object code for the work may be a
derivative work of the Library even though the source code is not.
Whether this is true is especially significant if the work can be
linked without the Library, or if the work is itself a library.  The
threshold for this to be true is not precisely defined by law.

  If such an object file uses only numerical parameters, data
structure layouts and accessors, and small macros and small inline
functions (ten lines or less in length), then the use of the object
file is unrestricted, regardless of whether it is legally a derivative
work.  (Executables containing this object code plus portions of the
Library will still fall under Section 6.)

  Otherwise, if the work is a derivative of the Library, you may
distribute the object code for the work under the terms of Section 6.
Any executables containing that work also fall under Section 6,
whether or not they are linked directly with the Library itself.

  A. As an exception to the Sections above, you may also combine or
link a "work that uses the Library" with the Library to produce a
work containing portions of the Library, and distribute that work
under terms of your choice, provided that the terms permit
modification of the work for the customer's own use and reverse
engineering for debugging such modifications.

  You must give prominent notice with each copy of the work that the
Library is used in it and that the Library and its use are covered by
this License.  You must supply a copy of this License.  If the work
during execution displays copyright notices, you must include the
copyright notice for the Library among them, as well as a reference
directing the user to the copy of this License.  Also, you must do one
of these things:

    I) Accompany the work with the complete corresponding
    machine-readable source code for the Library including whatever
    changes were used in the work (which must be distributed under
    Sections 1 and 2 above); and, if the work is an executable linked
    with the Library, with the complete machine-readable "work that
    uses the Library", as object code and/or source code, so that the
    user can modify the Library and then relink to produce a modified
```

(continues on next page)

(continued from previous page)

```
    executable containing the modified Library.  (It is understood
    that the user who changes the contents of definitions files in the
    Library will not necessarily be able to recompile the application
    to use the modified definitions.)

    I) Use a suitable shared library mechanism for linking with the
    Library.  A suitable mechanism is one that (1) uses at run time a
    copy of the library already present on the user's computer system,
    rather than copying library functions into the executable, and (2)
    will operate properly with a modified version of the library, if
    the user installs one, as long as the modified version is
    interface-compatible with the version that the work was made with.

    I) Accompany the work with a written offer, valid for at
    least three years, to give the same user the materials
    specified in Subsection 6a, above, for a charge no more
    than the cost of performing this distribution.

    I) If distribution of the work is made by offering access to copy
    from a designated place, offer equivalent access to copy the above
    specified materials from the same place.

    A) Verify that the user has already received a copy of these
    materials or that you have already sent this user a copy.

  For an executable, the required form of the "work that uses the
Library" must include any data and utility programs needed for
reproducing the executable from it.  However, as a special exception,
the materials to be 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.

  It may happen that this requirement contradicts the license
restrictions of other proprietary libraries that do not normally
accompany the operating system.  Such a contradiction means you cannot
use both them and the Library together in an executable that you
distribute.

  A. You may place library facilities that are a work based on the
Library side-by-side in a single library together with other library
facilities not covered by this License, and distribute such a combined
library, provided that the separate distribution of the work based on
the Library and of the other library facilities is otherwise
permitted, and provided that you do these two things:

    I) Accompany the combined library with a copy of the same work
    based on the Library, uncombined with any other library
    facilities.  This must be distributed under the terms of the
    Sections above.

    I) Give prominent notice with the combined library of the fact
    that part of it is a work based on the Library, and explaining
    where to find the accompanying uncombined form of the same work.

  A. You may not copy, modify, sublicense, link with, or distribute
```

(continues on next page)

(continued from previous page)

```
the Library except as expressly provided under this License.  Any
attempt otherwise to copy, modify, sublicense, link with, or
distribute the Library 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.

  I. 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 Library or its derivative works.  These actions are
prohibited by law if you do not accept this License.  Therefore, by
modifying or distributing the Library (or any work based on the
Library), you indicate your acceptance of this License to do so, and
all its terms and conditions for copying, distributing or modifying
the Library or works based on it.

  A. Each time you redistribute the Library (or any work based on the
Library), the recipient automatically receives a license from the
original licensor to copy, distribute, link with or modify the Library
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 with
this License.

  A. 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 Library at all.  For example, if a patent
license would not permit royalty-free redistribution of the Library 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 Library.

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.

  A. If the distribution and/or use of the Library is restricted in
certain countries either by patents or by copyrighted interfaces, the
```

(continues on next page)

(continued from previous page)

```
original copyright holder who places the Library 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.

  A. The Free Software Foundation may publish revised and/or new
versions of the Lesser 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 Library
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 Library does not specify a
license version number, you may choose any version ever published by
the Free Software Foundation.

  A. If you wish to incorporate parts of the Library into other free
programs whose distribution conditions are incompatible with these,
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

  A. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO
WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW.
EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR
OTHER PARTIES PROVIDE THE LIBRARY "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
LIBRARY IS WITH YOU.  SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME
THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  A. 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 LIBRARY 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
LIBRARY (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 LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF
SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH
DAMAGES.

                     END OF TERMS AND CONDITIONS

           How to Apply These Terms to Your New Libraries

  If you develop a new library, and you want it to be of the greatest
possible use to the public, we recommend making it free software that
```

(continues on next page)

(continued from previous page)

```
everyone can redistribute and change.  You can do so by permitting
redistribution under these terms (or, alternatively, under the terms of the
ordinary General Public License).

  To apply these terms, attach the following notices to the library.  It is
safest to attach them to the start of each source file to most effectively
convey the exclusion of warranty; and each file should have at least the
"copyright" line and a pointer to where the full notice is found.

    <one line to give the library's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This library is free software; you can redistribute it and/or
    modify it under the terms of the GNU Lesser General Public
    License as published by the Free Software Foundation; either
    version 2.1 of the License, or (at your option) any later version.

    This library 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
    Lesser General Public License for more details.

    You should have received a copy of the GNU Lesser General Public
    License along with this library; if not, write to the Free Software
    Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301  USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary.  Here is a sample; alter the names:

  Yoyodyne, Inc., hereby disclaims all copyright interest in the
  library `Frob' (a library for tweaking knobs) written by James Random Hacker.

  <signature of Ty Coon>, 1 April 1990
  Ty Coon, President of Vice

That's all there is to it!
```

1.3 Compiling libMariaS3

libMariaS3 is designed to be compiled with GCC or CLang on a modern Linux distrubition or Mac OSX.

1.3.1 Prerequisites

libMariaS3 requires *libcurl 7.x* and *libxml2* to be installed. For RPM based distributions this can be installed using:

```
sudo dnf install libcurl-devel libxml2-devel
```

1.3.2 Building

On most systems you can use the following commands, this is especially useful for customising your install:

```
autoreconf -fi
./configure
make
make install
```

The build system will automatically detect how many processor cores you have (physicaly and virtual) and set the `--jobs` options of make accordingly.

1.3.3 Testing

libMariaS3 comes with a basic test suite which we recommend executing, especially if you are building for a new platform.

You will need the following OS environment variables set to run the tests:

Variable	Desription
S3KEY	Your AWS access key
S3SECRET	Your AWS secret key
S3REGION	The AWS region (for example us-east-1)
S3BUCKET	The S3 bucket name
S3HOST	OPTIONAL hostname for non-AWS S3 service
S3NOVERIFY	Set to `1` if the host should not use SSL verification

The test suite is automatically built along with the library and can be executed with `make check` or `make distcheck`. If you wish to test with valgrind you can use:

```
TESTS_
↪ENVIRONMENT="./libtool --mode=execute valgrind --error-exitcode=1 --leak-check=yes --track-fds=yes
↪make check
```

1.3.4 Building RPMs

The build system for libMariaS3 has the capability to build RPMs. To build RPMs simply do the following:

```
autoreconf -fi
./configure
make dist-rpm
```

Note: The package `redhat-rpm-config` is required for building the RPM because this generates the debuginfo RPM.

1.3.5 Building DEBs

Debian packages for libMariaS3 can be built using the standard `dpkg-buildpackage` tool as follows:

```
autoreconf -fi
dpkg-buildpackage
```

Note: You may need to add `--no-sign` to dpkg-buildpackage to build unsigned packages.

CHAPTER 2

API Documentation

2.1 Functions

2.1.1 ms3_library_init()

void **ms3_library_init**(void)

Initializes the library for use. Should be called before any threads are spawned.

2.1.2 ms3_library_deinit()

void **ms3_library_deinit**(void)

Cleans up the library, typically for the end of the application's execution.

2.1.3 ms3_library_init_malloc()

uint8_t **ms3_library_init_malloc**(*ms3_malloc_callback* *m*, *ms3_free_callback* *f*, *ms3_realloc_callback* *r*, *ms3_strdup_callback* *s*, *ms3_calloc_callback* *c*)

Initialize the library for use with custom allocator replacement functions. These functions are also fed into libcurl. The function prototypes should be as follows:

void ***ms3_malloc_callback**(size_t *size*)

To replace `malloc()`.

void **ms3_free_callback**(void **ptr*)

To replace `free()`.

void ***ms3_realloc_callback**(void **ptr*, size_t *size*)

To replace `realloc()`.

char ***ms3_strdup_callback**(const char **str*)

To replace `strdup()`.

void ***ms3_calloc_callback**(size_t *nmemb*, size_t *size*)
To replace `calloc()`.

Should be called before any threads are spawned. All parameters are required or the function *will* fail.

Remember: With great power comes great responsibility.

Parameters

- **m** – The malloc callback
- **f** – The free callback
- **r** – The realloc callback
- **s** – The strdup callback
- **c** – The calloc callback

Returns `0` on success, `MS3_ERR_PARAMETER` if a parameter is `NULL`

2.1.4 ms3_init()

ms3_st ***ms3_init**(const char **s3key*, const char **s3secret*, const char **region*, const char **base_domain*)
Initializes a `ms3_st` object. This object should only be used in the thread that created it because it reuses connections. But it is safe to have other `ms3_st` objects running at the same time in other threads.

Note: You *MUST* call `ms3_library_init()` before spawning threads when using this access method.

Parameters

- **s3key** – The AWS access key
- **s3secret** – The AWS secret key
- **region** – The AWS region to use (such as `us-east-1`)
- **base_domain** – A domain name to use if AWS S3 is not the desired server (set to `NULL` for S3)

Returns A newly allocated marias3 object

2.1.5 ms3_deinit()

void **ms3_deinit**(*ms3_st* **ms3*)
Cleans up and frees a `ms3_st` object.

Parameters

- **ms3** – The marias3 object

2.1.6 ms3_server_error()

const char ***ms3_server_error**(*ms3_st* **ms3*)
Returns the last error message from the S3 server or underlying Curl library.

Parameters

- **ms3** – The marias3 object

Returns The error message string or `NULL` if there is no message.

2.1.7 ms3_error()

const char ***`ms3_error`**(*uint8_t errcode*)

Returns an error message for a given error code

Parameters

- **`errcode`** – The error code to translate

Returns The error message

2.1.8 ms3_debug()

void **`ms3_debug`**()

Enables and disables debugging output on stderr. Each call toggles enable / disable.

Note:: This enables/disables globally for the library

2.1.9 ms3_list()

uint8_t **`ms3_list`**(*ms3_st *ms3*, const char **bucket*, const char **prefix*, *ms3_list_st **list*)

Retrieves a list of files from a given S3 bucket and fills it into a *`ms3_list_st`*.

The list generated is the eqivilent of a recursive directory listing but only has files in it, no entries for directories.

The list will automatically be freed on the next list/list_dir call or *`ms3_deinit()`*

Parameters

- **`ms3`** – The marias3 object
- **`bucket`** – The bucket name to use
- **`prefix`** – An optional path/file prefix to use (`NULL` for all files)
- **`list`** – A pointer to a pointer that will contain the returned list

Returns `0` on success, a positive integer on failure

Example

```
char *s3key= getenv("S3KEY");
char *s3secret= getenv("S3SECRET");
char *s3region= getenv("S3REGION");
char *s3bucket= getenv("S3BUCKET");
ms3_list_st *list= NULL, *list_it= NULL;
uint8_t res;

ms3_library_init();
ms3_st *ms3= ms3_thread_init(s3key, s3secret, s3region, NULL);

res= ms3_list(ms3, s3bucket, NULL, &list);
if (res)
{
    printf("Error occurred: %d\n", res);
```

(continues on next page)

(continued from previous page)

```
    return;
}
list_it= list;
while(list_it)
{
  printf("File: %s, size: %ld, tstamp: %ld\n", list_it->key, list_it->length, list_it-
↪>created);
  list_it= list_it->next;
}
ms3_deinit(ms3);
```

2.1.10 ms3_list_dir()

uint8_t **ms3_list_dir**(*ms3_st *ms3*, const char **bucket*, const char **prefix*, *ms3_list_st **list*)

Retrieves a list of files from a given S3 bucket and fills it into a `ms3_list_st`.

The list generated will automatically add the delimiter `/` and therefore filter up to the first `/` after the prefix. Unlike `ms3_list()` it includes directory entries. This is the eqivilent of doing a regular directory listing in a current directory (as designated by `prefix`).

The list will automatically be freed on the next list/list_dir call or `ms3_deinit()`

Parameters

- **ms3** – The marias3 object
- **bucket** – The bucket name to use
- **prefix** – An optional path/file prefix to use (`NULL` for all files)
- **list** – A pointer to a pointer that will contain the returned list

Returns `0` on success, a positive integer on failure

2.1.11 ms3_list_free()

void **ms3_list_free**(*ms3_list_st *list*)

Deprecated since version 3.1.1: Now a NULL operation which be removed in 4.0

A NULL operation, previously free'd `ms3_list()`, but this is now done internally on `ms3_deinit()` or when a new list is requested.

Parameters

- **list** – The list to free

2.1.12 ms3_put()

uint8_t **ms3_put**(*ms3_st *ms3*, const char **bucket*, const char **key*, const *uint8_t *data*, size_t *length*)

Puts a binary data from a given pointer into S3 at a given key/filename. If an existing key/file exists with the same name this will be overwritten.

Parameters

- **ms3** – The marias3 object
- **bucket** – The bucket name to use

- **key** – The key/filename to create/overwrite
- **data** – A pointer to the data to write
- **length** – The length of the data to write

Returns 0 on success, a positive integer on failure

Example

```
char *s3key= getenv("S3KEY");
char *s3secret= getenv("S3SECRET");
char *s3region= getenv("S3REGION");
char *s3bucket= getenv("S3BUCKET");
uint8_t res;
const char *test_string= "Another one bites the dust";

ms3_library_init();
ms3_st *ms3= ms3_thread_init(s3key, s3secret, s3region, NULL);

res= ms3_put(ms3, s3bucket, "test/ms3.txt", (const uint8_t*)test_string, strlen(test_
↪string));
if (res)
{
    printf("Error occurred: %d\n", res);
    return;
}
ms3_deinit(ms3);
```

2.1.13 ms3_copy()

uint8_t **ms3_copy** (*ms3_st *ms3*, const char **source_bucket*, const char **source_key*, const char **dest_bucket*, const char **dest_key*)

S3 internally copies an object from a source bucket and key to a destination bucket and key.

Parameters

- **ms3** – The marias3 object
- **source_bucket** – The bucket where the source object is
- **source_key** – The key/filename of the source object
- **dest_bucket** – The destination bucket (can be the same as source)
- **dest_key** – The destination key/filename

Returns 0 on success, a positive integer on failure

2.1.14 ms3_move()

uint8_t **ms3_move** (*ms3_st *ms3*, const char **source_bucket*, const char **source_key*, const char **dest_bucket*, const char **dest_key*)

Moves an object from source to destination. Internally the library performs a copy and if successful performs a delete on the source object.

Parameters

- **ms3** – The marias3 object

- **source_bucket** – The bucket where the source object is
- **source_key** – The key/filename of the source object
- **dest_bucket** – The destination bucket (can be the same as source)
- **dest_key** – The destination key/filename

Returns 0 on success, a positive integer on failure

2.1.15 ms3_get()

uint8_t **ms3_get** (*ms3_st *ms3*, const char **bucket*, const char **key*, *uint8_t **data*, size_t **length*)

Retrieves a given object from S3.

Note: The application is expected to free the resulting data pointer after use

Parameters

- **ms3** – The marias3 object
- **bucket** – The bucket name to use
- **key** – The key/filename to retrieve
- **data** – A pointer to a pointer the data to be retrieved into
- **length** – A pointer to the data length

Returns 0 on success, a positive integer on failure

Example

```
char *s3key= getenv("S3KEY");
char *s3secret= getenv("S3SECRET");
char *s3region= getenv("S3REGION");
char *s3bucket= getenv("S3BUCKET");
uint8_t res;
uint8_t *data= NULL;
size_t length;

ms3_library_init();
ms3_st *ms3= ms3_thread_init(s3key, s3secret, s3region, NULL);

res= ms3_get(ms3, s3bucket, "test/ms3.txt", &data, &length);
if (res)
{
    printf("Error occurred: %d\n", res);
    return;
}
printf("File contents: %s\n", data);
printf("File length: %ld\n", length);
ms3_free(data);
ms3_deinit(ms3);
```

2.1.16 ms3_free()

void **ms3_free** (*uint8_t *data*)

Used to free the data allocated by *ms3_get()*.

Parameters

- **data** – The data to free

2.1.17 ms3_set_option()

uint8_t **ms3_set_option** (*ms3_st *ms3*, *ms3_set_option_t option*, void **value*)

Sets a given connection option. See *ms3_set_option_t* for a list of options.

Parameters

- **ms3** – The marias3 object
- **option** – The option to set
- **value** – A pointer to the value for the option (if required, NULL if not)

Returns 0 on success, a positive integer on failure

2.1.18 ms3_delete()

uint8_t **ms3_delete** (*ms3_st *ms3*, const char **bucket*, const char **key*)

Deletes an object from an S3 bucket

Parameters

- **ms3** – The marias3 object
- **bucket** – The bucket name to use
- **key** – The key/filename to delete

Returns 0 on success, a positive integer on failure

Example

```
char *s3key= getenv("S3KEY");
char *s3secret= getenv("S3SECRET");
char *s3region= getenv("S3REGION");
char *s3bucket= getenv("S3BUCKET");
uint8_t res;

ms3_library_init();
ms3_st *ms3= ms3_thread_init(s3key, s3secret, s3region, NULL);

res = ms3_delete(ms3, s3bucket, "test/ms3.txt");
if (res)
{
    printf("Error occurred: %d\n", res);
    return;
}
ms3_deinit(ms3);
```

2.1.19 ms3_status()

uint8_t **ms3_status** (*ms3_st *ms3*, const char **bucket*, const char **key*, *ms3_status_st *status*)

Retreives the status of a given filename/key into a `ms3_status_st` object. Will return an error if not found.

Parameters

- **ms3** – The marias3 object
- **bucket** – The bucket name to use
- **key** – The key/filename to status check
- **status** – A status object to fill

Returns `0` on success, a positive integer on failure

Example

```
char *s3key= getenv("S3KEY");
char *s3secret= getenv("S3SECRET");
char *s3region= getenv("S3REGION");
char *s3bucket= getenv("S3BUCKET");
uint8_t res;
ms3_status_st status;

ms3_library_init();
ms3_st *ms3= ms3_thread_init(s3key, s3secret, s3region, NULL);

res= ms3_status(ms3, s3bucket, "test/ms3.txt", &status);
if (res)
{
    printf("Error occurred: %d\n", res);
    return;
}
printf("File length: %ld\n", status.length);
printf("File timestamp: %ld\n", status.created);
ms3_deinit(ms3);
```

2.2 Structs

ms3_st

An internal struct which contains authentication information

ms3_list_st

A linked-list struct which contains a list of files/keys and information about them

char ***key**

The key/filename for the object

size_t **length**

The data size for the object

time_t **created**

The created / updated timestamp for the object

struct *ms3_list_st* ***next**

A pointer to the next struct in the list

ms3_status_st

An struct which contains the status of an object

size_t **length**

The data size for the object

time_t **created**

The created / updated timestamp for the object

2.3 Constants

ms3_set_option_t

Options to use for *ms3_set_option()*. Possible values:

- `MS3_OPT_USE_HTTP` - Use `http://` instead of `https://`. The `value` parameter of *ms3_set_option()* is unused and each call to this toggles the flag (HTTPS is used by default)
- `MS3_OPT_DISABLE_SSL_VERIFY` - Disable SSL verification. The `value` parameter of *ms3_set_option()* is unused and each call to this toggles the flag (SSL verification is on by default)
- `MS3_OPT_BUFFER_CHUNK_SIZE` - Set the chunk size in bytes for the receive buffer. Default is 1MB. If you are receiving a large file a realloc will have to happen every time the buffer is full. For performance reasons you may want to increase the size of this buffer to reduce the reallocs and associated memory copies. The `value` parameter of *ms3_set_option()* should be a pointer to a `size_t` greater than 1.
- `MS3_OPT_FORCE_LIST_VERSION` - An internal option for the regression suite only. The `value` parameter of *ms3_set_option()* should be a pointer to a *uint8_t* of value `1` or `2`
- `MS3_OPT_FORCE_PROTOCOL_VERSION` - Set to 1 to force talking to the S3 server using version 1 of the List Bucket API, this is for S3 compatible servers. Set to 2 to force talking to the S3 server version 2 of the List Bucket API. This is for use when the autodetect bsaed on providing a base_domain does the wrong thing. The `value` parameter of *ms3_set_option()* should be a pointer to a *uint8_t* of value `1` or `2`
- `MS3_OPT_READ_CB` - Custom read callback for *ms3_get()*. The `value` parameter of *ms3_set_option()* should be a `ms3_read_callback` function.
- `MS3_OPT_USER_DATA` - User data for the custom read callback. The `value` parameter of *ms3_set_option()* is the pointer that will be passed as the `userdata` argument of the callback.

2.4 Built-In Types

NULL

A null pointer as defined in the standard header `string.h`.

uint8_t

An unsigned single byte character as defined in the standard header `stdint.h`

bool

A boolean type as defined in the standard header `stdbool.h`

2.5 Error Codes

Code	Details
MS3_ERR_NONE	Success (always equal to 0)
MS3_ERR_PARAMETER	A required function parameter is missing
MS3_ERR_NO_DATA	No data is supplied to a function that requires data
MS3_ERR_URI_TOO_LONG	The generated URI for the request is too long
MS3_ERR_RESPONSE_PARSE	The API could not parse the response from S3
MS3_ERR_REQUEST_ERROR	The API could not send the request to S3
MS3_ERR_OOM	Could not allocate required memory
MS3_ERR_IMPOSSIBLE	A theortically impossible condition occurred
MS3_ERR_AUTH	Authentication failed
MS3_ERR_NOT_FOUND	Object not found
MS3_ERR_SERVER	Unknown error code in S3 response
MS3_ERR_TOO_BIG	PUT data is too large, 4GB maximum

2.6 Compiling Your Application

2.6.1 Include Files

Make sure that your application includes the main libMariaS3 include as follows:

```
#include <libmarias3/marias3.h>
```

This will pull in all the libMariaS3 functions and constants you may require for your application.

2.6.2 Package Config

libMaria3e includes a utility called `libmarias3-config`. This can give you all the options used to compile the library as well as the compiler options to link the library. For a full list of what it providesrun:

```
libmarias3-config --help
```

2.6.3 Compiling

If the library is installed correctly in your Linux distribution compiling your application with libMariaS3 should be a simple matter of adding the library to link to as follows:

```
gcc -o basic basic.c -lmarias3
```

And likewise for CLang:

```
clang -o basic basic.c -lmarias3
```

CHAPTER 3

Contributing

3.1 Introduction to Contributing

There are many ways to contribute to libMariaS3. Simply using it and creating an issue report when you found a bug or have a suggestion is a great contribution. Documentation and code contribituions are also greatly appreciated.

3.1.1 Layout

The code for libMariaS3 in several parts:

Directory	Contents
`/src`	The API source code
`/libmarias3`	The public API headers
`/tests`	Unit tests for the public API

In each case if any files are added or removed the `include.am` file in that directory will require updating to reflect the change.

3.1.2 Submitting to Github

The main hub for the code is GitHub. The main tree is the libMariaS3 GitHub tree. Anyone is welcome to submit pull requests or issues. All requests will be considered and appropriate feedback given.

3.1.3 Modifying the Build System

The build system is an m4 template system called DDM4. If any changes are made to the scripts in `m4` directory the *serial* line will need incrementing in that file. You should look for a line near the top that looks like:

```
#serial 7
```

Shared Library Version

If any of the source code has changed please see LIBMARIAS3_LIBRARY_VERSION in configure.ac. This gives rules on bumping the shared library versioning, not to be confused with the API public version which follows similar rules as described in the next section.

3.1.4 API Version

API versioning is stored in the VERSION.txt file which is used by the build system to version the API and docs. The versioning scheme follows the Semantic Versioning Rules.

3.1.5 Function Visibility

The code and build system only exposes public API functions as usable symbols in the finished binary. This cuts down on binary size quite significantly and also discourages use of undocumented functionality that was not designed for public use.

When adding a new API function to /libmarias3 always add MS3_API on its own on the line above the function definition in the header. This tells the build system this is an API function to be included.

3.1.6 License Headers

Please make sure before committing that all new files have appropriate license headers in. Only add to the copyright of older headers if you have made a significant contribution to that file (25 - 50 lines is typically classed as significant for Open Souce projects).

3.2 Coding Standard

3.2.1 General

We are aiming for a minimum of C99 support. A script in extra can be found called astyle.sh. This uses the Linux tool Artistic Style to enforce coding standards.

3.2.2 Coding Style

Everyone has a preferred coding style, there is no real correct style. What is important is that we stick to one style throughout the code.

We should use a variant of the Allman coding style. The variation is to use 2 spaces instead of tabs. The exception to the rule is Makefiles where space indentation can break them.

Allman style specifies that braces associated with a statement should be on the following line with the same indentation and the statements inside the braces are next level indented. The closing braces are also on a new line at the same indentation as the original statement.

For example:

```
while (x == y)
{
  something();
  somethingelse();
}
finalthing();
```

3.2.3 Types

Use C99 types (where possible), this will very much help us to find conversion bugs. So:

- Use bool, not my_bool.
- Use true and false, not TRUE and FALSE (those macros need to die).
- ulong → uint32_t
- ulonglong uint64_t
- long int → int32_t

The keyword *NULL* should always be used when referring to the pointer NULL

3.2.4 Allocation

For performance reasons we should try to limit the number of times we allocate and deallocate memory. Do not do thousands of allocates and deallocates to save 32k of RAM.

3.2.5 Naming style

Variable names

Variables should be verbosely names, no caps, underscores with spaces. Do not just use `i` in for loops, again we have developers with bad eyes.

Types

New types should use the `_t` postfix. Private structs should be typedef'ed and also use this.

Public Structs

Public structs should be typedef'ed and use the `_st` postfix

Conventions

- use *column* instead of *field*
- use *schema* instead of *database*

3.2.6 Include Files

Includes that will be installed need to be written like:

```
#include <drizzled/field/blob.h>
```

The following should only be used in cases where we are to never install these libraries in the filesystem:

```
#include "item.h"
```

3.2.7 Comments

Where it is not obvious what is going on. Hopefully most of the code will be self-commenting.

All code should have license headers.

Comment blocks should use the format:

```
/* Comment Block
 * This is a multi-line comment block
 */
```

C99 style in-line and single line comments are allowed for small comments

```
// small comment
```

3.2.8 Line lengths

Whilst there is no hard limit on line lengths it is recommended that lines stay under 80 characters unless going above this increases readability of the code.

3.3 Updating Documentation

3.3.1 Overview

This documentation is stored along with the source in the `docs` directory of the git tree and uses the reStructuredText format.

We recommend reading this reStructuredText Primer before editing the docs for the first time.

3.3.2 Compiling Docs

The docs are compiled using Sphinx Python Documentation Generator. The libMariaS3 build system already knows how to use this. To compile the docs please follow theses steps:

1. Install the `python-sphinx` package using your distribution's package manager
2. Re-run bootstrap as follows so that it picks up that Sphinx is installed:

   ```
   ./bootstrap.sh -m
   ```

3. To compile in HTML format:

```
make html
```

There will now be an HTML version of the docs in the `/html` directory of the source.

3.3.3 Compiling PDF Docs

Sphinx required LaTeX to build PDF docs. The following steps show you how to build PDF docs:

1. Install `python-sphinx` as above
2. Install the full *TeXLive* package. In Fedora this is `texlive-scheme-full` and `texlive-full` in Ubuntu
3. Re-run bootstrap as follows so that it picks up that Sphinx and LaTeX are installed:

   ```
   ./bootstrap.sh -m
   ```

4. To compile in PDF format:

   ```
   make latexpdf
   ```

The generated PDF will be in the `/docs/latex/` directory.

3.4 Writing Test Cases

libMariaS3 uses DDM4's YATL library to create unit tests, this provides macros to test if the outcomes are as expected.

3.4.1 Adding a Test Case

Test cases are basic C applications in the `tests/` directory. To add a test case to the suite. To add a test edit the `include.am` and add the following (replacing *mytest* with whatever the test is called):

```
t_mytest_SOURCES= tests/mytest.c
t_mytest_LDADD= src/libmarias3.la
check_PROGRAMS+= t/mytest
noinst_PROGRAMS+= t/mytest
```

3.4.2 Using YATL

YATL is needed to make sure conditions within the test program are met. To include it in your test application, add the following:

```
#include <yatl/lite.h>
```

A test skip can be added if certain conditions aren't met:

```
SKIP_IF_(!is_connected, "Cannot connected to a database server")
```

There are many types of assert provided as can be seen in the next section, they can be used as follows:

```
ASSERT_EQ_(3, column, "Column count unexpected)
ASSERT_FALSE_(false_condition, "False condition is not false")
ASSERT_STREQ_("test", some_data, "Unexpected data")
```

3.4.3 YATL Library

Parameter Definitions

__expression
: An expression typically used in an `if` statement.

__expected
: An expected variable or expression

__actual
: The actual variable or expression

__expected_str
: The expected string

__actual_str
: The actual string to compare with

__length
: The length of a string for comparison

Function Definitions

SKIP_IF (*__expression*)
: Skips the test if the expression is true

SKIP_IF_ (*__expression*, ...)
: Skips the test if the expression is true and uses a printf style format message

ASSERT_TRUE (*__expression*)
: Make sure the expression is true, test will fail if it is false

ASSERT_FALSE (*__expression*)
: Make sure the expression is false, test will fail if it is true

ASSERT_FALSE_ (*__expression*, ...)
: Make sure the expression is false and use a printf style format message to fail if it is true.

ASSERT_NULL_ (*__expression*, ...)
: Make sure the expression is `NULL` and use a printf style format message to fail if it isn't.

ASSERT_NOT_NULL (*__expression*)
: Make sure the expression is not `NULL`, test will fail if it is `NULL`.

ASSERT_NOT_NULL_ (*__expression*, ...)
: Make sure the expression is not `NULL` and use a printf style format message to fail if it is.

ASSERT_TRUE_ (*__expression*, ...)
: Make sure the expression is `true` and use a printf style format message to fail if it is not.

ASSERT_EQ (*__expected*, *__actual*)
: Make sure that one condition or variable matches another one.

> **Note:** Not suitable for string matching

ASSERT_EQ_ (*__expected*, *__actual*, ...)
: Make sure that one condition or variable matches another one and use a printf style format message to fail if the do not match.

Note: Not suitable for string matching

ASSERT_NEQ (*__expected*, *__actual*)
Make sure that one condition or variable does not match another one.

Note: Not suitable for string matching

ASSERT_NEQ_ (*__expected*, *__actual*, ...)
Make sure that one condition or variable does not match another one and use a printf style format message to fail if they do match.

Note: Not suitable for string matching

ASSERT_STREQ (*__expected_str*, *__actual_str*)
Compare one `NUL` terminated string with another one and fail if they do not match.

ASSERT_STREQ_ (*__expected_str*, *__actual_str*, ...)
Compare one `NUL` terminated string with another one and use a printf style format message to fail if they do not match.

ASSERT_STREQL_ (*__expected_str*, *__actual_str*, *__length*, ...)
Compare a string of `__length` to another one and use a printf style format message to fail if they do not match.

Note: This is designed for use with non-NUL-terminated strings.

ASSERT_STRNE (*__expected_str*, *__actual_str*)
Compare one `NUL` terminated string with another one and fail if they match.

ASSERT_STRNE_ (*__expected_str*, *__actual_str*, ...)
Compare one `NUL` terminated string with another one and use a printf style format message to fail if they match.

3.5 Using GitHub

GitHub contributions typically work by creating a fork of the project on your user account, making a branch on that fork to work on and then filing a pull request to upstream your code. This is how you would go about it.

3.5.1 Forking

Go to the libMariaS3 GitHub page and click the *Fork* button near the top. Once you have forked you can get a local copy of this fork to work on (where *user* is your username):

```
git clone https://github.com/user/libmarias3.git
```

You then need to make your local clone aware of the upstream repository:

```
cd libmarias3
git remote add upstream https://github.com/mariadb-corporation/libmarias3.git
```

3.5.2 Branch

Before creating a branch to work on you should first make sure your local copy is up to date:

```
git checkout master
git pull --ff-only upstream master
git push
```

You can then create a branch from master to work on:

```
git checkout -b a_new_feature
```

3.5.3 Hack on code!

Hack away at your feature or bug.

3.5.4 Test

Once your code is ready the test suite should be run locally:

```
make
make check
```

If there are documentation changes you should install `python-sphinx` and try to build the HTML version to run a syntax check:

```
make html
```

3.5.5 Commit and push

If you have never contributed to GitHub before then you need to setup git so that it knows you for the commit message:

```
git config --global user.name "Real Name"
git config --global user.email "me@me.com"
```

Make sure you use *git add* to add any new files to the repository and then commit:

```
git commit -a
```

Your editor will pop up to enter a commit messages above the comments. The first line should be no more than 50 characters and be a subject of the commit. The second line should be blank. The third line onwards can contain details and these should be no more than 72 characters each.

If your commit fixes an issue you can add the following (for issue #93):

```
Fixes mariadb-corporation/libmarias3#93
```

Once all your commits are done a quick rebase may be needed to make sure your changes will merge OK with what is in master:

```
git fetch upstream
git rebase -i upstream/master
```

This should bring up a commit-style message in the editor with *pick* as the first word. Save this and the rebase will complete. If the rebase tells you there is a conflict you will need to locate the problem using `git diff`, fix it and do:

```
git add <filename>
git rebase --continue
```

If things look like they are going wrong you can undo the rebase using the following and can get in touch with us:

```
git rebase --abort
```

You should now be ready to push up to GitHub:

```
git push --set-upstream origin a_new_feature
```

If you go to your repository on GitHub's website you will an option to file a *Pull Request*. Use this to submit a pull request upstream for your branch.

3.5.6 Help

If you get stuck at any point feel free to reach out to us by filing an issue on Github.

CHAPTER 4

Appendix

4.1 Version History

4.1.1 Version 3.1

Version 3.1.3 GA

- Fix *ms3_copy()* not working correctly with non-alphanumeric characters (also affected *ms3_move()*)

Version 3.1.2 GA

- Make library work with quirks in Google Cloud's S3 implementation
- Detect when libcurl was built with OpenSSL < 1.1.0 and add workaround to thread safety issues in the older OpenSSL versions (affects Ubuntu 16.04 in particular)
- Remove libxml and replace it with a modified version of xml.c which handles <? ?> tags and other minor changes
- Fix issue where an empty key for *ms3_get()* turns it into a list call
- Partially fix issue with `AC_MSG_ERROR`. Will still fail if you don't have `libtool` and `pkg-config` installed.

Version 3.1.1 GA (2019-06-28)

- Fix bad host header when path based buckets are used
- Make autodetection of access type and list version *much* smarter:
  - Checks for S3 domain in provided domain and uses list version 2
  - Checks for IP provided domain and turns on list version 1 and path based buckets
  - Any other domain uses list version one and domain based buckets

- Reduced linked list mallocs for `ms3_list()` and `ms3_list_dir()`. This also deprecates `ms3_list_free()`.

Version 3.1.0 GA (2019-06-24)

- Fix compiling issues when `-Wdeclaration-after-statement` is enabled
- Add `MS3_OPT_FORCE_PROTOCOL_VERSION` for use with `ms3_set_option()` which will force use of AWS S3 methods and paths (version 2) or compatible methods and paths (version 1)
- Fix double-free upon certain errors
- Add snowman UTF-8 test and minor cleanups
- Cleanup build system

4.1.2 Version 3.0

Version 3.0.2 GA (2019-05-24)

- Fix libm linkage
- Remove mhash dependency and use a modified cut-down version of wpa_supplicant's BSD licensed crypto code (required for Windows compiling)
- Several minor performance optimizations
  - Removed 2x1kb mallocs on every request (now on `ms3_init()` instead)
  - Compiling with `-O3` by default
  - Stop executing string compares in list loop when something is found
  - Remove unneeded `strdup()` usage

Version 3.0.1 GA (2019-05-16)

- Improve performance of PUT
- Fix a few potential pointer arithmatic issues
- Fix race condition on time generation
- Added TSAN to ci-scripts
- Fix minor issues found in cppcheck
- Stop buffer overrun if the buffer chunk size is set smaller than packet
- Fix `ms3_get()` returning random data if a CURL request completely fails
- Fix potential crash if the server error message is junk
- Fix double-free if a server error message is `NULL`

Version 3.0.0 GA (2019-05-13)

- Allow compiling to work with gnu89 compiler mode
- Fix building in CLang
- Removed previous deprecated `ms3_thread_init` and `ms3_buffer_chunk_size`
- Remove `bool` from frontend API by:
  - Making *ms3_debug()* a toggle
  - Making the boolean options of *ms3_set_option()* toggles

4.1.3 Version 2.3

Version 2.3.0 GA (2019-05-07)

- Allow compiling with a C++ compiler
- Fix logic error in *ms3_move()*
- Stop *ms3_get()* returning the error message as the object data on error
- Add *ms3_list_dir()* to get a non-recursive directory listing
- Setting the buffer chunk size using `ms3_buffer_chunk_size` or *ms3_set_option()* no longer has a lower limit of 1MB

4.1.4 Version 2.2

Version 2.2.0 GA (2019-04-23)

- Add *ms3_init()* to replace `ms3_thread_init` and deprecate the latter.
- Add *ms3_library_init_malloc()* to add custom allocators
- Add *ms3_library_deinit()* to cleanup‘
- Add *ms3_copy()* and *ms3_move()* to use S3’s internal file copy

4.1.5 Version 2.1

Version 2.1.1 GA (2019-04-02)

- Remove iso646.h support in codebase
- Autoswitch to bucket path instead of bucket domain access method (for IP urls)
- Fixed issue with SSL disabled verification
- Fixed minor leak when base_domain is set
- Add `S3NOVERIFY` env var to tests which will disable SSL verification when set to `1`

Version 2.1.0 GA (2019-03-29)

- Add *ms3_set_option()* to set various connection options
- Deprecated ms3_buffer_chunk_size, use *ms3_set_option()* instead
- Added options to use http instead of https and to disable SSL verification
- Added debugging output for server/curl error messages
- Added compatibility for V1 bucket list API. Will turn on automatically for non-Amazon S3 compatible servers. Additionally an option has been created to force V1 or V2

4.1.6 Version 2.0

Version 2.0.0 GA (2019-03-28)

- Fix double-free when using ms3_thread_init and an error occurs
- Fix error when a PUT >= 65535 is attempted
- Improve performance of GET for large files
- Make ms3_thread_init treat empty string base_domain as NULL
- Add *ms3_free()*
- Add ms3_buffer_chunk_size
- Cleanup linking
- Removed ms3_init
- Added *ms3_server_error()* to get the last server or Curl error

4.1.7 Version 1.1

Version 1.1.0 GA (2019-03-27)

- Fix memory leak in libxml2 function usage
- Fix memory leaks in libcurl usage
- Fix test collisions causing failures
- Added *ms3_library_init()* and ms3_thread_init for higher-performance acceses

4.1.8 Version 1.0

Version 1.0.1 RC (2019-03-26)

- Fixed issues found with valgrind, cppcheck and scanbuild
- Added RPM & DEB build systems
- Fixed pagination calls for *ms3_list()* so it support > 1000 objects
- Made ms3_init() thread safe

Version 1.0.0 Beta (2019-03-25)

- Initial Beta version

4.2 Credits

The libMariaS3 authors are:

- Andrew (LinuxJedi) Hutchings
- Sergei Golubchik
- Markus Mäkelä

libMariaS3 uses the following Open Source projects:

- libcurl
- xml.c
- DDM4
- Jouni Malinen’s SHA256 hash code
- genindex
- search

Index