# **Compiling ArangoDB from scratch**

The following sections describe how to compile and build the ArangoDB from scratch. The ArangoDB will compile on most Linux and Mac OS X systems. We assume that you use the GNU C/C++ compiler or clang/clang++ to compile the source. ArangoDB has been tested with the GNU C/C++ compiler and clang/clang++, but should be able to compile with any Posix-compliant, C++11-enabled compiler. Please let us know whether you successfully compiled it with another C/C++ compiler.

There are the following possibilities:

- **all-in-one**: this version contains the source code of the ArangoDB, all generated files from the autotools, FLEX, and BISON as well as a version of V8, libev, and ICU.
- **devel**: this version contains the development version of the ArangoDB. Use this branch if you want to make changes to the ArangoDB source.

The **devel** version requires a complete development environment, while the **all-in-one** version allows you to compile the ArangoDB without installing all the prerequisites. The disadvantage is that it takes longer to compile and you cannot make changes to the flex or bison files.

## **Amazon Micro Instance**

@sohgoh has reported that it is very easy to install ArangoDB on an Amazon Micro Instance:

only works on local instance with LDS

amazon> sudo yum install readline-devel amazon> ./configure amazon> make amazon> make install

For detailed instructions the following section.

## **All-In-One Version**

Note: there are separate instructions for the **devel** version in the next section.

## **Basic System Requirements**

Verify that your system contains:

- the GNU C/C++ compilers "gcc" and "g++" and the standard C/C++ libraries. You will need compiler and library support for C++11. To be on the safe side with gcc/g++, you will need version number 4.8.1 or higher. For "clang" and "clang++", you will need at least version 3.4.
- GNU make

In addition you will need the following libraries:

- the GNU readline library
- the OpenSSL library
- Go 1.2 (or higher)

Under Mac OS X you also need to install:

- Xcode
- scons

#### **Download the Source**

Download the latest source using GIT:

```
git clone git://github.com/arangodb/arangodb.git
```

Note: if you only plan to compile ArangoDB locally and do not want to modify or push any changes, you can speed up cloning substantially by using the --single-branch and -- depth parameters for the clone command as follws:

git clone --single-branch --depth 1 git://github.com/arangodb/arangodb.git

RootFile

## **Configure**

Switch into the ArangoDB directory



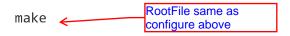
In order to configure the build environment execute

```
RootFile referencing above cd command
```

to setup the makefiles. This will check the various system characteristics and installed libraries.

## **Compile**

Compile the program by executing



This will compile the ArangoDB and create a binary of the server in



#### **Test**

Create an empty directory

```
unix> mkdir /tmp/database-dir RootFile

Check the binary by starting it using the command line.

unix> ./bin/arangod -c etc/relative/arangod.conf --server.endpoint tcp://127.0.0.1:852
```

This will start up the ArangoDB and listen for HTTP requests on port 8529 bound to IP address 127.0.0.1. You should see the startup messages similar to the following:

```
2013-10-14T12:47:29Z [29266] INFO ArangoDB xxx ...
2013-10-14T12:47:29Z [29266] INFO using endpoint 'tcp://127.0.0.1.8529' for non-encryp
2013-10-14T12:47:30Z [29266] INFO Authentication is turned off
2013-10-14T12:47:30Z [29266] INFO ArangoDB (version xxx) is ready for business. Have f
```

If it fails with a message about the database directory, please make sure the database

directory you specified exists and can be written into.

Use your favorite browser to access the URL

This should produce a JSON object like

```
{"server" : "arango", "version" : "..."}
```

as result.

### Install

Install everything by executing



You must be root to do this or at least have write permission to the corresponding directories.

The server will by default be installed in

```
/usr/local/sbin/arangod
```

The configuration file will be installed in

```
/usr/local/etc/arangodb/arangod.conf
```

The database will be installed in

```
/usr/local/var/lib/arangodb
```

The ArangoShell will be installed in

```
/usr/local/bin/arangosh
```

When upgrading from a previous version of ArangoDB, please make sure you inspect ArangoDB's log file after an upgrade. It may also be necessary to start ArangoDB with the --upgradeparameter once to perform required upgrade or initialization tasks.

## **Devel Version**

Note: a seperate blog article is available that describes how to compile ArangoDB from source on Ubuntu.

#### **Basic System Requirements** Constraint Verify that your system contains • the GNU C/C++ compilers "gcc" and "g++" and the standard C/C++ libraries, with support for C++11. You will need version gcc 4.8.1 or higher. For "clang" and "clang++", you will need at least version 3.4, • the GNU autotools (autoconf, automake) Constraint GNU make Constraint the GNU scanner generator FLEX, at least version 2.3.35 Constraint the GNU parser generator BISON, at least version 2.4 Constraint Python, version 2 or 3 the OpenSSL library, version 1.0.1g or higher (development package) Constraint the GNU readline library (development package)

Most Linux systems already supply RPMs or DPKGs for these packages. Some distributions, for example Ubuntu 12.04 or Centos 5, provide only very out-dated versions of compilers, FLEX, BISON, and/or the V8 engine. In that case you need to compile newer versions of the programs and/or libraries.

When compiling with special configure options, you may need the following extra libraries:

• the Boost test framework library (only when using configure option --enable-maintainer-mode)

### **Download the Source**

Go, version 1.2.2

Download the latest ArangoDB source using git:



## Setup

Switch into the ArangoDB directory



The source tarball contains a pre-generated "configure" script. You can regenerate this script by using the GNU auto tools. In order to do so, execute



This will call aclocal, autoheader, automake, and autoconf in the correct order.

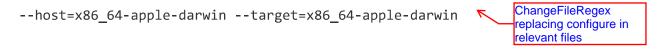
## Configure

In order to configure the build environment please execute

```
unix> ./configure RootFile with reference
```

to setup the makefiles. This will check for the various system characteristics and installed libraries.

Please note that it may be required to set the --host and --targetvariables when running the configure command. For example, if you compile on MacOS, you should add the following options to the configure command:



The host and target values for other architectures vary.

If you also plan to make changes to the source code of ArangoDB, add the following option to the *configure* command: --enable-maintainer-mode. Using this option, you can make changes to the lexer and parser files and some other source files that will generate other files. Enabling this option will add extra dependencies to BISON, FLEX, and PYTHON. These external tools then need to be available in the correct versions on

your system.

The following configuration options exist:



This will make relative paths be used in the compiled binaries and scripts. It allows to run ArangoDB from the compile directory directly, without the need for a *make install* command and specifying much configuration parameters. When used, you can start ArangoDB using this command:



ArangoDB will then automatically use the configuration from fileetc/relative/arangod.conf.

```
--enable-all-in-one-etcd ChangeFileRegex
```

This tells the build system to use the bundled version of ETCD. This is the default and recommended.

```
--enable-internal-go____ChangeFileRegex
```

This tells the build system to use Go binaries located in the 3rdParty directory. Note that ArangoDB does not ship with Go binaries, and that the Go binaries must be copied into this directory manually.



This will also build arangoirb, and interactive MRuby REPL shell. This is an experimental feature.

\_\_\_\_Constraint

--enable-maintainer-mode ChangeFileRegex

Constraint Constraint

This tells the build system to use BISON and FLEX to regenerate the parser and scanner files. If disabled, the supplied files will be used so you cannot make charges to the parser and scanner files. You need at least BISON 2.4.1 and FLEX 2.5.35. This option also allows you to make changes to the error messages file, which is converted to js and C header files using Python. You will need Python 2 or 3 for this. Furthermore, this option enables additional test cases to be executed in a *make unittests* run. You also need to install the Boost test framework for this.

Additionally, turning on the maintainer mode will turn on a lot of assertions in the code.

```
--enable-failure-tests ChangeFileRegex
```

This option activates additional code in the server that intentionally makes the server crash or misbehave (e.g. by pretending the system ran out of memory). This option is useful for writing tests.

```
--enable-v8-debug → ChangeFileRegex
```

Builds a debug version of the V8 library. This is useful only when working on the V8 integration inside ArangoDB.

## **Compiling Go**

Users F21 and duralog told us that some systems don't provide an update-to-date version of go. This seems to be the case for at least Ubuntu 12 and 13. To install go on these system, you may follow the instructions provided here. For other systems, you may follow the instructions here.

To make ArangoDB use a specific version of go, you may copy the go binaries into the 3rdParty/go-32 or 3rdParty/go-64 directories of ArangoDB (depending on your architecture), and then tell ArangoDB to use this specific go version by using the -- enable-internal-go configure option.

ChangeFileRegex

User duralog provided some the following script to pull the latest release version of go into the ArangoDB source directory and build it:

```
cd ArangoDB
hg clone -u release https://code.google.com/p/go 3rdParty/go-64 && \
    cd 3rdParty/go-64/src && \
        ./all.bash
# now that go is installed, run your configure with --enable-internal-go
        ./configure --enable-internal-go
RootFile
```

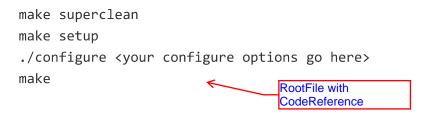
# Re-building ArangoDB after an update

To stay up-to-date with changes made in the main ArangoDB repository, you will need to pull the changes from it and re-runmake.

Normally, this will be as simple as follows:



From time to time there will be bigger structural changes in ArangoDB, which may render the old Makefiles invalid. Should this be the case and make complains about missing files etc., the following commands should fix it:



This will clean up ArangoDB and the 3rd party libraries, and rebuild everything.

If you forgot your previous configure options, you can look them up with



before issuing make superclean (as make superclean also removes the file config.log).