gRPC C++开发环境搭建

零声教育 C/C++Linux服务器开发/高级架构师

特别需要强调，grpc需要6.3以上的gcc/g++版本，如果低于此版本的需要参考文档进行升级。

1. cmake 、gcc的版本， ubuntu16.04默认的版本不支持。

1 安装必要的依赖工具

安装必要的依赖工具

```
sudo apt-get install autoconf automake libtool
```

如果cmake低于3.15， gcc/g++ 低于 7.0 ，请根据文档进行安装。查看版本的方式

```
# 检查cmake版本
cmake -version
# 检查gcc/g++版本
gcc -v
g++ -v
```

1.1 安装 cmake

可以下载更新的版本：

最低版本为3.15（低于该版本才需要程序安装）。

1. 卸载已经安装的旧版的CMake

```
sudo apt-get autoremove cmake
```

2. 文件下载解压

```
wget https://cmake.org/files/v3.23/cmake-3.23.0-linux-x86_64.tar.gz
```

解压：

```
tar zxf cmake-3.23.0-linux-x86_64.tar.gz
```

查看解压后目录:

```
tree -L 2 cmake-3.23.0-linux-x86_64

cmake-3.23.0-linux-x86_64
├── bin
│   ├── ccmake
│   ├── cmake
│   ├── cmake-gui
│   ├── cpack
│   └── ctest
├── doc
│   └── cmake
├── man
│   ├── man1
│   └── man7
└── share
    ├── aclocal
    ├── applications
    ├── bash-completion
    ├── cmake-3.23
    ├── emacs
    ├── icons
    ├── mime
    └── vim
```

bin下面有各种cmake家族的产品程序.

3. 创建软链接

注: 文件路径是可以指定的, 一般选择在 `/opt` 或 `/usr` 路径下, 这里选择 `/opt`

```
sudo mv cmake-3.23.0-linux-x86_64 /opt/cmake-3.23.0
sudo ln -sf /opt/cmake-3.23.0/bin/*  /usr/bin/
```

4. 测试版本

```
ubuntu@VM-16-11-ubuntu:~/rpc$ cmake -version
cmake version 3.23.0

CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

1.2 安装gcc/gdb

升级gcc和gdb的版本，至少需要6.3以上的版本。

Operating System	Architectures	Versions	Support Level
Linux - Debian, Ubuntu, CentOS	x86, x64	clang 6+, GCC 6.3+	Officially Supported
Windows 10+	x86, x64	Visual Studio 2017+	Officially Supported
MacOS	x86, x64	XCode 12+	Officially Supported
Linux - Others	x86, x64	clang 6+, GCC 6.3+	Best Effort

注意：如果已经是高于7.0 不需要再次安装。

目标： 安装 gcc g++ 7的安装包

1. 安装

```
sudo apt-get install -y software-properties-common
sudo add-apt-repository ppa:ubuntu-toolchain-r/test
sudo apt update
sudo apt install g++-7 -y
```

2. 建立软连接并检查

```
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 60 \
                         --slave /usr/bin/g++ g++ /usr/bin/g++-7
sudo update-alternatives --config gcc
gcc -v
g++ -v
```

显示7.5的版本。

2 编译grpc

推荐使用cmake的方式进行编译。 grpc安装过程比较依赖网络的通畅性（容易被墙），我是租用了**腾讯云香港服务器下载的grpc源代码**，他不仅是grpc源码本身，还依赖了很多第三方库，比如protobufer。大家可以直接使用我提供的源码包（**900+MB**，记得先解压）进行编译。

如果不能翻墙，直接采用我提供的压缩包 grpc-v1.45.2.tar.bz2，则解压

```
解压方式:
tar -jxf grpc-v1.45.2.tar.bz2
```

解压完直接跳到步骤 4.**编译和安装。**如果能翻墙则可以从 步骤1. **下载源码**开始。

1. **下载源码**

```
git clone  https://github.com/grpc/grpc
```

2. **查看版本并选择合适的版本，这里选择v1.45.2相对较新的版本**

```
git tag
git checkout v1.45.2
```

查看此时grpc目录内容的大小du -h --max-depth=1， 可以看到**427M左右**

```
ubuntu@VM-16-11-ubuntu:~/rpc/grpc$ du -h --max-depth=1
348M    ./.git
32K ./summerofcode
1.5M    ./doc
6.5M    ./tools
4.0K    ./spm-core-include
24M ./test
80K ./cmake
3.0M    ./third_party
4.0K    ./spm-cpp-include
1.5M    ./templates
8.0K    ./.bazelci
1.9M    ./include
5.0M    ./examples
34M ./src
268K    ./etc
64K ./.github
284K    ./bazel
427M    .
```

3. **下载第三方依赖库**，下载完后会发现整个grpc目录内容明显变大

```
git submodule update --init
```

再次查看 目录大小，**占用了1.3G。**

```
ubuntu@VM-16-11-ubuntu:~/rpc/grpc$ du -h --max-depth=1
```

```
899M    ./.git
32K ./summerofcode
1.5M    ./doc
6.5M    ./tools
4.0K    ./spm-core-include
24M ./test
80K ./cmake
291M    ./third_party
4.0K    ./spm-cpp-include
1.5M    ./templates
8.0K    ./.bazelci
1.9M    ./include
5.0M    ./examples
34M ./src
268K    ./etc
64K ./.github
284K    ./bazel
1.3G
```

4. **编译和安装**

```
mkdir -p cmake/build
cd cmake/build
cmake ../..
#多核可以使用多线程编译，如果是云服务器的1、2核的，则建议只用make，不要用多线程
make -j4
sudo make install
```

3 protobuf安装

不用手动安装protobuf，不然版本可能和grcp不匹配，必须在 grpc 执行 git submodule update --init 命令之后生成的 **third_party/protobuf** 里面编译安装对应的 protobuf。在编译grpc时已经默认安装

先使用protoc --version测试

```
protoc --version
显示3.19.4说明当前protobuf已经安装了
```

如果protobuf版本不对则参考下面方式安装：

```
cd third_party/protobuf/
./autogen.sh
./configure --prefix=/usr/local
make

sudo make install
sudo ldconfig  # 使得新安装的动态库能被加载

protoc --version
显示3.19.4
```

4 测试环境

编译helloworld

```
cd grpc/examples/cpp/helloworld/
mkdir build
cd build/
cmake ..
make
```

启动服务和客户端

```
# 启动服务端，监听在50051端口
./greeter_server
Server listening on 0.0.0.0:50051
# 启动客户端，服务端返回Hello world
./greeter_client
Greeter received: Hello world
```

5 参考

ubuntu搭建grpc for C++开发环境*wx5bb365de633ed的技术博客*51CTO博客 **该文档提供修改grpc第三方库下载地址的方式进行安装。**

6 辅助-使用scp命令，远程上传下载文件/文件夹

这里只是提供一种方式供大家可以在服务器之间传递文件，不是该节课程的内容，仅供参考。

1. **从服务器下载文件**

```
scp username@servername:/path/filename /local/path
```

例如: scp ubuntu@192.168.1.222:/ubuntu/data/data.txt /desktop/ubuntu 把192.168.1.222上的/ubuntu/data/data.txt 的文件下载到/desktop/ubuntu目录中

2. **上传本地文件到服务器**

```
scp /local/path/local_filename username@servername:/path
```

例如: scp /ubuntu/learning/deeplearning.doc ubuntu@192.168.1.222:/ubuntu/learning 把本机/ubuntu/learning/目录下的deeplearning.doc文件上传到192.168.1.222这台服务器上的/ubuntu/learning目录中

3. **从服务器下载整个目录**

```
scp -r username@servername:/path /path
```

例如: scp -r ubuntu@192.168.1.222:/home/ubuntu/data /local/local_dir “-r”命令是文件夹目录，把当前/home/ubuntu/data目录下所有文件下载到本地/local/local_dir目录中

4. **上传目录到服务器**

```
scp  -r  /path  username@servername:/path
```

例如: scp -r /ubuntu/test ubuntu@192.168.1.222:/ubuntu/tx “-r”命令是文件夹目录，把当前/ubuntu/test目录下所有文件上传到服务器的/ubuntu/tx/目录中