超简单!!centos 6离线源码编译安装升级gcc、binutils、automake、autoconf

发表于2016/12/23 21:05:53  180人阅读

分类： Linux应用开发技术 系统配置

centos 6系列的Linux发行版，是非常成功（具足UNIX精神）的发行版，可能大家都已经用得非常习惯了。  
但问题是，其内核及附带的工具软件，版本却都非常老了。  
centos 7上的软件版本虽然比较新，但centos 7设计风格的突变（主要是引入了很不符合UNIX精神的systemd），可能在业界也引起了不少争议。  
而我们苦逼的开发者，可能面临既需要使用高版本的相关组件，又不想升级到centos 7的困境。  
那就升级centos 6上的内核与工具吧。  
升级内核还算好办，因为内核对编译环境的要求很低。不熟悉内核升级的朋友，可以参考如下博文。  
http://blog.csdn.net/crazycoder8848/article/details/44131735  
  
  
但是升级gcc等工具，如果不熟悉情况的话，可能就不顺利了。  
公司研发环境上网的不便，centos 6环境下各种高版本rpm软件包的缺失，都会给升级带来麻烦。  
本文提供的离线源码升级方法，就非常适合解决上述难题。  
  
  
这里顺便说明一下，本文的标题虽然带了离线二字，但并不表示本人没有通过Linux主机下载相关文件。  
本文为了操作及行文的方便，可能有多处会用wget去下载相关软件包或指示下载行为。  
只所以带离线二字，是因为本文提供的方法非常适用于离线操作。  
最后，由于本文的升级全部是基于源码编译安装，与具体的包管理系统（如rpm、apt等）无关。  
因此，本文的方法应该也适用于其他Linux发行版（如ubuntu等）。  
  
  
先来看看笔者的Linux主机升级前的情况：

[root@localhost ~]# cat /etc/redhat-release

CentOS release 6.5 (Final)

[root@localhost ~]# uname -a

Linux localhost.localdomain 2.6.32-431.el6.i686 #1 SMP Fri Nov 22 00:26:36 UTC 2013 i686 i686 i386 GNU/Linux

[root@localhost ~]# gcc --version

gcc (GCC) 4.4.7 20120313 (Red Hat 4.4.7-4)

[root@localhost ~]# autoconf --version

autoconf (GNU Autoconf) 2.63

[root@localhost ~]# automake --version

automake (GNU automake) 1.11.1

gnu binutils包含as、ld、objdump等工具。从下面的输了信息来看，相关工具的版本号，就是binutils的版本号。

[root@localhost ~]# rpm -qa binutils

binutils-2.20.51.0.2-5.36.el6.i686

[root@localhost ~]# as --version

GNU assembler version 2.20.51.0.2-5.36.el6 20100205

[root@localhost ~]# ld --version

GNU ld version 2.20.51.0.2-5.36.el6 20100205

[root@localhost ~]# objdump -v

GNU objdump version 2.20.51.0.2-5.36.el6 20100205

为什么要升级binutils呢？因为我们熟悉的编译工具gcc自己只能将C代码编译为以.s结尾的文本形式的汇编文件。

而.s到.o的过程，则需要由as来完成。而as属于gnu binutils软件包。因此，如果不升级binutils，可能会出现这种情况：

gcc按照新款cpu的特点编译 C代码，生成的.s文件中含有较新的指令，例如avx2指令（新款x86系列cpu的加速指令）。

而老版的as程序却不认识此指令，结果导致编译失败。另外，像objdump反汇编二进制文件时，也同样可能出现某些指令不能识别的问题。

好了，下面看看相关工具的升级过程吧。

先来升级gcc吧。

//创建一个干净的目录，用于下载及升级诸工具

[root@localhost ~]# mkdir tools\_update

[root@localhost ~]# cd tools\_update/

[root@localhost tools\_update]# wget http://ftp.gnu.org/gnu/gcc/gcc-4.8.2/gcc-4.8.2.tar.bz2

[root@localhost tools\_update]# wget ftp://gcc.gnu.org/pub/gcc/infrastructure/mpfr-2.4.2.tar.bz2

[root@localhost tools\_update]# wget ftp://gcc.gnu.org/pub/gcc/infrastructure/gmp-4.3.2.tar.bz2

[root@localhost tools\_update]# wget ftp://gcc.gnu.org/pub/gcc/infrastructure/mpc-1.0.3.tar.gz

//接下来为gcc的编译做些准备。

//注意，我们这里不用自己去编译上面下载的3个库。

//我们参考gcc-4.8.2/contrib/download\_prerequisites中的方法，

//让gcc的编译脚本自动帮我们配置并编译这3个库。

//具体步骤如下：

[root@localhost tools\_update]# tar -xjf gcc-4.8.2.tar.bz2

[root@localhost tools\_update]# cd gcc-4.8.2

[root@localhost gcc-4.8.2]# tar -xjf ../mpfr-2.4.2.tar.bz2

[root@localhost gcc-4.8.2]# ln -sf mpfr-2.4.2 mpfr

[root@localhost gcc-4.8.2]# tar -xjf ../gmp-4.3.2.tar.bz2

[root@localhost gcc-4.8.2]# ln -sf gmp-4.3.2 gmp

[root@localhost gcc-4.8.2]# tar -xzf ../mpc-1.0.3.tar.gz

[root@localhost gcc-4.8.2]# ln -sf mpc-1.0.3 mpc

//下面开始编译gcc了。编译过程超级漫长。

//注意--prefix=/usr的设置很重要。

//因为这样设置，在后面make install时可以直接覆盖老的gcc，安装后就不用再额外设置其他东西了。

//我们后面其他工具的编译安装，也都采用此设置。这样最简单。

[root@localhost gcc-4.8.2]# ./configure --prefix=/usr --enable-languages=c,c++ --enable--long-long --enable-threads=posix --disable-checking --disable-multilib

[root@localhost gcc-4.8.2]# make

[root@localhost gcc-4.8.2]# make install

//好了，看看升级结果吧^\_^

[root@localhost gcc-4.8.2]# gcc --version

gcc (GCC) 4.8.2

//然后升级binutils，毕竟这和基础的构建功能是强相关的。

[root@localhost tools\_update]# wget http://ftp.gnu.org/gnu/binutils/binutils-2.25.1.tar.bz2

[root@localhost tools\_update]# tar -xjf binutils-2.25.1.tar.bz2

[root@localhost tools\_update]# cd binutils-2.25.1

[root@localhost binutils-2.25.1]# ./configure --prefix=/usr

[root@localhost binutils-2.25.1]# make

[root@localhost binutils-2.25.1]# make install

//好了，看看结果吧^\_^

[root@localhost binutils-2.25.1]# as --version

GNU assembler (GNU Binutils) 2.25.1

[root@localhost binutils-2.25.1]# objdump -v

GNU objdump (GNU Binutils) 2.25.1

[root@localhost binutils-2.25.1]# ld -v

GNU ld (GNU Binutils) 2.25.1

//接下来升级其他工具。注意，请保持顺序与本文一致。否则可能会失败。

[root@localhost automake-1.14.1]# wget http://ftp.gnu.org/gnu/autoconf/autoconf-2.68.tar.xz

[root@localhost tools\_update]# tar -xJf autoconf-2.68.tar.xz

[root@localhost tools\_update]# cd autoconf-2.68

[root@localhost autoconf-2.68]# ./configure --prefix=/usr

[root@localhost autoconf-2.68]# make

[root@localhost autoconf-2.68]# make install

[root@localhost tools\_update]# wget http://ftp.gnu.org/gnu/automake/automake-1.14.1.tar.xz

[root@localhost tools\_update]# tar -xJf automake-1.14.1.tar.xz

[root@localhost tools\_update]# cd automake-1.14.1

[root@localhost automake-1.14.1]# ./configure --prefix=/usr

[root@localhost automake-1.14.1]# make

[root@localhost automake-1.14.1]# make install

//好了，看看结果吧^\_^

[root@localhost automake-1.14.1]# autoconf --version

autoconf (GNU Autoconf) 2.68

[root@localhost automake-1.14.1]# automake --version

automake (GNU automake) 1.14.1

不过有时候，版本高了，也会带来额外的麻烦。

像binutils，从2.22版本开始，在链接生成可执行程序时，不会自动的隐式链接所需的库，而是需要明确的指定链接什么库。

结果这导致编译内核时，执行make menuconfig失败。

不过，解决办法是有的。

按照上文介绍的方法，编译安装一个2.21版的binutils即可。

以后想切到哪个版本了，直接进入源码目录执行一下make install即可。