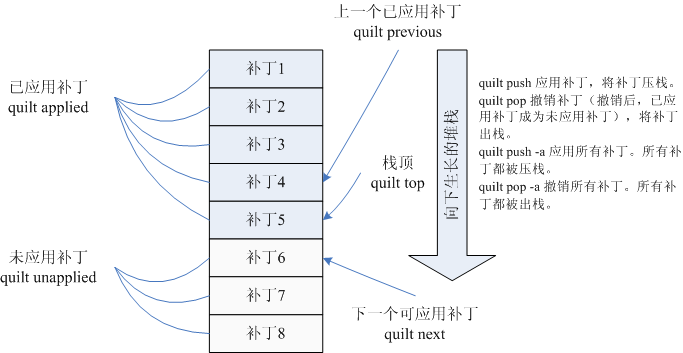
# Quilt使用介绍

1. Quilt简单介绍

Quilt是一个帮助我们管理Patch的程序掌握quilt的关键是了解使用quilt的流程。使用quilt时，我们会在一个完整的源代码树里工作。只要我们在源代码树里使用了quilt命令，quilt就会在源代码树的根目录建立两个特殊目录：patches和.pc。quilt在patches目录保存它管理的所有Patch。quilt用.pc目录保存自己的内部工作状态，用户不需要了解这个目录。

patches/series文件记录了quilt当前管理的Patch。Patch按照加入的顺序排列，早加入的Patch在前。quilt用堆栈的概念管理Patch的应用。



我们在应用PatchA前，必须先应用所有早于PatchA的Patch。所以，patches/series中的Patch总是从上向下应用。例如：上图中，Patch1到Patch5是已经应用的Patch。我们可以将已应用的Patch想象成一个向下生长的堆栈，栈顶就是已应用的最新Patch。应用Patch就是将Patch入栈，撤销Patch就是将Patch出栈。

**我们在源代码树中作任何修改前，必须用"quilt add"命令将要修改的文件与一个Patch联系起来。在完成修改后，用"quilt refresh"命令将修改保存到已联系的Patch。**

**在使用quilt时，请务必记住以上红色的字，否则在使用上有可能会有问题。**

我们这里主要用的是Kernel的Patch维护管理。

1. **Quilt使用介绍**
2. **Quilt的安装**

audo apt-get install quilt

目前版本是 0.50

1. **Quilt的参数**

Usage: quilt [--trace[=verbose]] [--quiltrc=XX] command [-h] ...

quilt --version

Commands are:

add fold new remove top

annotate fork next rename unapplied

applied graph patches revert upgrade

delete grep pop series

diff header previous setup

edit import push shell

files mail refresh snapshot

Global options:

--trace

Runs the command in bash trace mode (-x). For internal debugging.

--quiltrc file

Use the specified configuration file instead of ~/.quiltrc (or

/etc/quilt.quiltrc if ~/.quiltrc does not exist). See the pdf

documentation for details about its possible contents. The

special value "-" causes quilt not to read any configuration

file.

--version

Print the version number and exit immediately.

1. **代码后必须要先全编译一次**

不要先执行QUILT=1，否则后续编译可能会有问题

其实全编译的操作，包括了patch的导入

1. **然后执行一次QUILT=1**

make package/example/{clean,prepare} V=s QUILT=1

或者

make package/example/host/{clean,prepare} V=s QUILT=1

或者

make target/linux/{clean,prepare} V=s QUILT=1

1. **查看所有Patches**

**quilt series**

查看当前或者栈顶Patch

**quilt top**

1. **Patch的导入**

在Patch导入后，会在生成patches的文件夹

比如：

build\_dir/linux-ar71xx\_generic/linux-3.3.8/patches

我们还可以用下面方式导入Patch：

1. 可以用**quilt import**命令导入Patch
2. **quilt push -a**

在导入后，

1. 可以通过以下命令查询当前已应用的Patches

**quilt applied**

1. 可以通过以下命令查询当前未应用的Patches
2. **quilt unapplied**
3. **新建Patch**

**quilt new platform/xxx.patch**

或者

**quilt new gerneric/xxx.patch**

1. **修改文件**

**我们必须将对源代码树所作的任何改动都和一个Patch联系起来。add命令将文件的当前状态与Patch联系起来。add命令的格式为：**

**quilt add [-P patch] {file}**

如果未指定Patch名，文件就与栈顶Patch联系起来。这样add命令为指定Patch保存了指定文件的当前快照，当我们执行refresh命令时，quilt就会检查文件的变化，将差异保存到指定Patch中。

做过一次以上操作后，我们就可以对目标文件用vim进行编辑、修改、保存。

对于需要修改的文件，我们还可以用以下命令执行，其中包括了quit add的操作，只不过是对栈顶Patch的文件操作，并不能指定Patch名：

**quilt edit {file}**

1. **新增文件**

**同上，在新增文件前，我们必须将对源代码树所作的任何改动都和一个Patch联系起来。add命令将文件的当前状态与Patch联系起来。add命令的格式为：**

**quilt add [-P patch] {file}**

然后才能在对应的文件夹下新增需要的文件。

1. **查看已修改的文件**

**quilt files [param]**

使用"quilt files [Patch]"命令可以查看与指定Patch关联的文件。

使用"quilt files -val"可以查看所有Patch联系的所有文件。"-v"参数表示更友好的显示，"-a"参数表示显示所有Patch，"-l"参数显示Patch。

1. **对比修改的文件**

**quilt diff [param]**

使用"quilt diff -z [-P Patch] [文件名]"可以查看指定Patch指定文件的当前改动。省略-P参数表示查看当前Patch的改动，省略文件名表示查看所有改动。

1. **保存修改**

**quilt refresh [patch]**

"quilt refresh [patch]"刷新patch，即将指定Patch的文件变化保存到Patch。省略文件名表示刷新栈顶Patch。

1. **提交Patch**

make package/example/update V=s

或者

make target/linux/update package/index V=s

这时候我们用svn status，就可以找到对应的patch文件更新。

**对于我们修改和提交Kernel的代码，按照以上7-13的流程，基本上就可以了。**

**不过还是需要记住重要的一条：**

**我们在源代码树中作任何修改前，必须用"quilt add"命令将要修改的文件与一个Patch联系起来。在完成修改后，用"quilt refresh"命令将修改保存到已联系的Patch。**

1. **其他几个重要的命令**
2. **Patch删除**

**quilt delete [patch]**

1. **从Patch中移除文件**

**quilt remove [-P patch] {file}**

1. **撤销所有patches**

**quilt pop -a**

Revision history

| Revision | Date | Description | Author(s) |
| --- | --- | --- | --- |
| 0.1 | Aug 20, 2014 | Initial draft | Wancan.You |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |