Linux kernel-Git Workflow

The Linux kernel development process is managed through a combination of tools and workflows to ensure the quality, stability, and security of the kernel. At its core, this workflow revolves around the use of Git as the version control system. Here's a simplified summary of how it works:

1. Clone the Kernel Repository

To start contributing to the Linux kernel development process, you first need to clone the official kernel repository from the kernel.org website. You can do this by using `git clone` in your terminal.

```
git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
```

2. Checkout a Branch

The kernel development process is organized around various branches, each representing a major version of the Linux kernel (e.g., `linux-5.16`, `linux-5.17` for different versions). You would typically checkout a specific branch relevant to your needs.

```
git checkout linux-5.16
```

3. Build and Test

Before submitting changes, you need to ensure that the kernel compiles successfully on your system (assuming you are running a supported architecture) and tests run without errors.

```
 # Compile the kernel
make mrproper && make defconfig && make -j$(nproc)

# Run basic testing
make modules_test
```

4. Develop Your Patchset

Once your build is successful, you can start making changes to the kernel codebase. This could involve modifying existing files or adding new ones.

```
# Make your changes here
```

5. Commit Changes

After completing your development work, it's time to commit your changes into a Git repository. Since kernel development typically involves patching, you would use `git add` and `git commit --amend` (if necessary) to prepare your patches for submission.

```
 # Add files changed in this session
git add <file1> <file2>

# If changes need to be amended before final commit
git commit --amend

# Final commit of the patchset
git commit -m "Patchset: Your Patch Description"
```

6. Push Changes

When your patches are ready, you can push them up to a public repository (like kernel.org) or to Gerrit for review.

```
 # If pushing directly to kernel.org (Note: Not recommended)
git push origin master

# Or if using Gerrit:
git push ssh://<username>@reviews.kernel.org:29418/linux/kernel.git HEAD:refs/for/mas
```

7. Submit Your Patches

Linux kernel development involves submitting your patches via email or through the Git review system (Gerrit). Once submitted, maintainers will evaluate and potentially integrate them into the mainline branch.

```
 # Email submission process is handled manually at this point.
```

This summary highlights the core steps involved in contributing to Linux kernel development. The actual workflow can be complex due to the sheer size of the codebase, strict coding standards, and the necessity for rigorous testing before inclusion in the official repository.

Curated by Brajesh Kumar