

# Using git in a > 150 user corporate environment

David Brown

# Introduction

- 150+ developers to work on Linux kernel
- Most come from Perforce
- Our gradual movement to Git

# Step 1

- Kernel into P4
- git-p4 with fake grafted history on real
- Partial patch process
- All commits end up back in P4

# Problems

- Too easy to check in junk
- Painful branches
- Core kernel team hated Perforce

# Today's Process

- Central repo with 'next' and 'master'
- Two humans push to 'next'
- Automated system tests and advances 'next' to 'master'

# Today...

- `format-patch/send-email`
- and push to `dev/username*`
- pre-commit checks branches and records
- scripts cull records to help maintainers

# Today...

- Acked-by tracks reviews
- Mostly discussed in email
- Biggest problem is figuring out which commit to actually pick into 'next'

# Reality

- More complex.
  - More branches
  - Exchange is evil
  - People are getting paid, patches can't get dropped

# Teaching

- Glazed eyes with concepts
- Difficult to find good subset
- Commit text is difficult
- Basic “dumb things”

# “Dumb Things”

- `commit --amend`, on the ‘master’ merge
- `reset --hard`
- Bad granularity: too many, not enough
- Usually people who most need advanced commands are least equipped to use them

# Independent trees

- >450 commits vs >150 on an upstream
- Many are poorly planned: often same parts of files
- Visibility taken for granted in open-source
- Merge is too hard