ExtFUSE

Extension Framework for File systems in User space

Ashish Bijlani, Umakishore Ramachandran
Georgia Institute of Technology

In-Kernel vs User File systems

- **Examples**
  - Ext4, OverlayFS, etc.
- **Pros**
  - Native performance
- **Cons**
  - Poor security/reliability
  - Not easy to develop/ debug/maintain

- **Examples**
  - EncFS, Gluster, etc.
- **Pros**
  - Improved security/reliability
  - Easy to develop/debug/ maintain
- **Cons**
  - **Poor performance!**

File System in User space (FUSE)

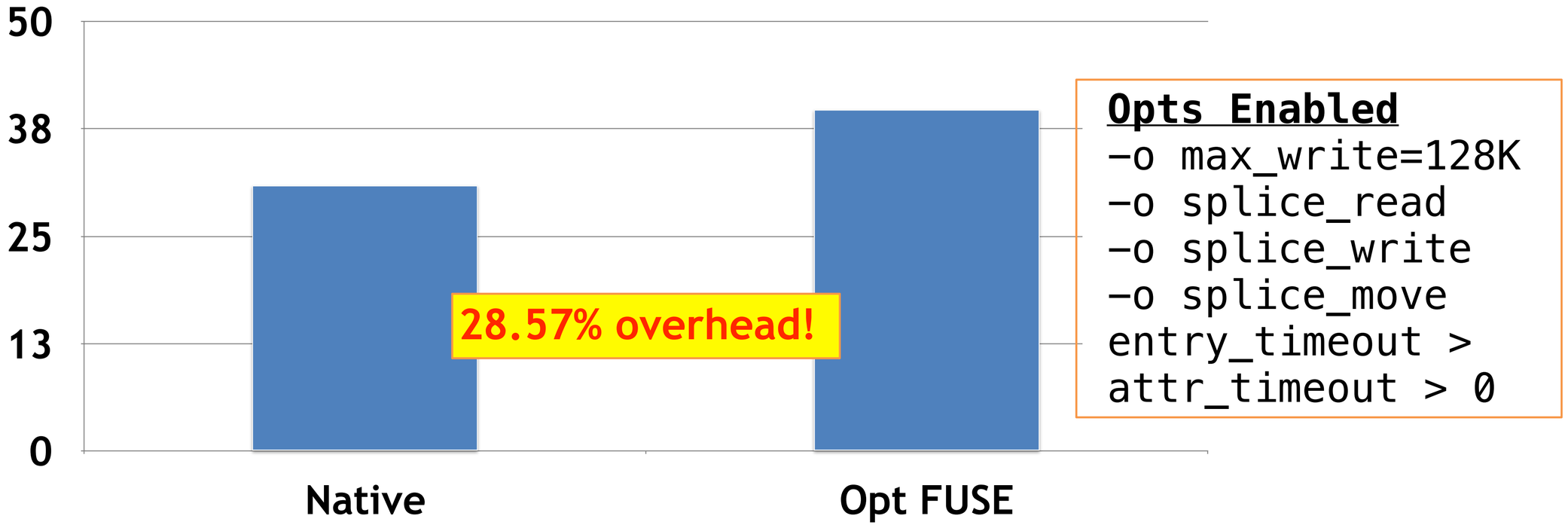
- State-of-the-art framework
  - All file system handlers implemented in user space

- Over 100+ FUSE file systems
  - Stackable: Android SDCardFS, EncFS, etc.
  - Network: GlusterFS, Ceph, Amazon S3FS, etc.

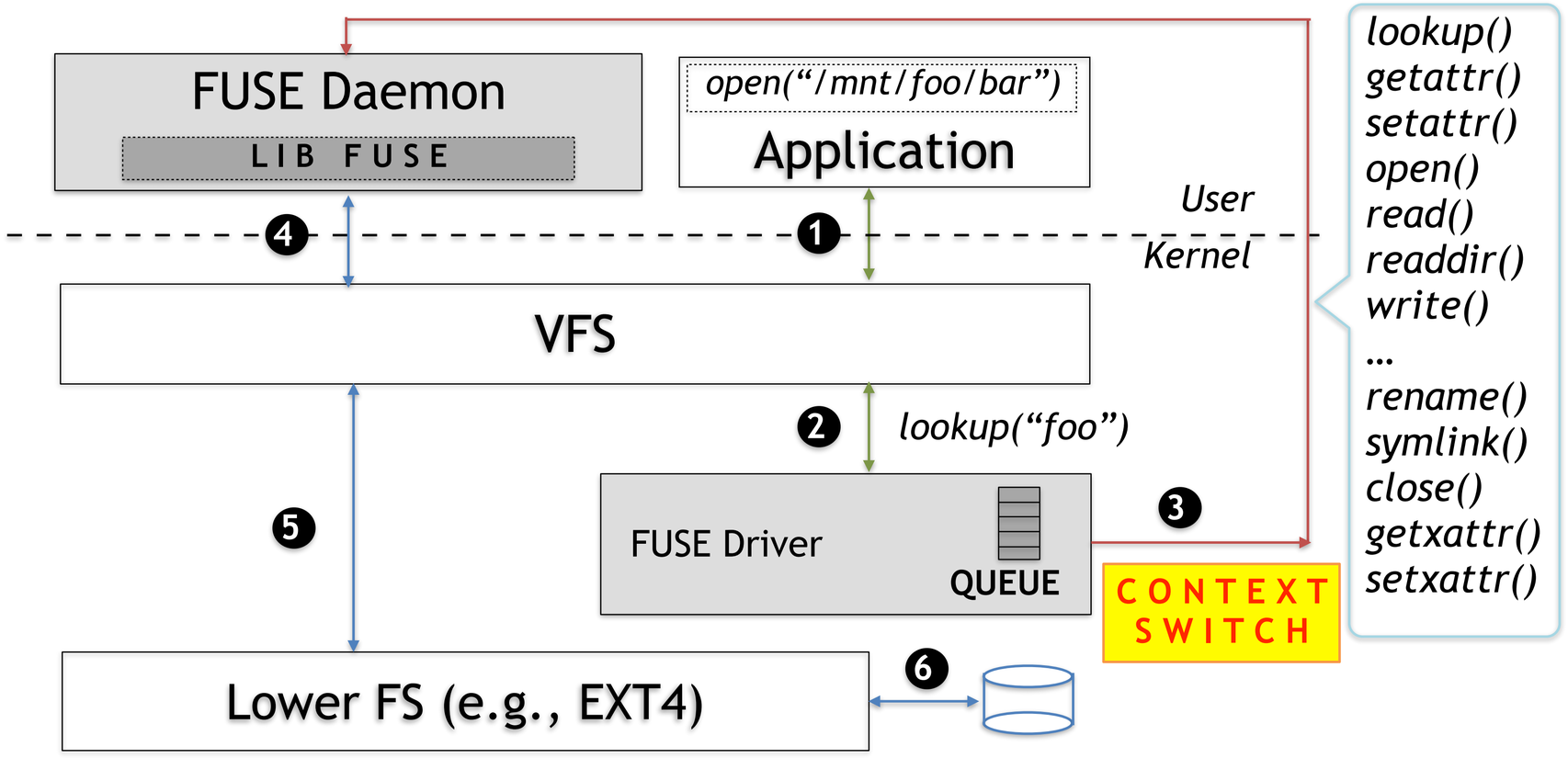
```
struct fuse_lowlevel_ops ops {
   .lookup  = handle_lookup,
   .access  = NULL,
   .getattr = handle_getattr,
   .setattr = handle_setattr,
   .open    = handle_open,
   .read    = handle_read,
   .readdir = handle_readdir,
   .write   = handle_write,
   // more handlers …
   .getxattr = handle_getxattr,
   .rename = handle_rename,
   .symlink = handle_symlink,
   .flush   = NULL,
}
```

FUSE Performance

- ***"cd linux-4.18; make tinyconfig; make -j4"***
  - Intel i5-3350 quad core, Ubuntu 16.04.4 LTS
  - Linux 4.11.0, LibFUSE commit # 386b1b

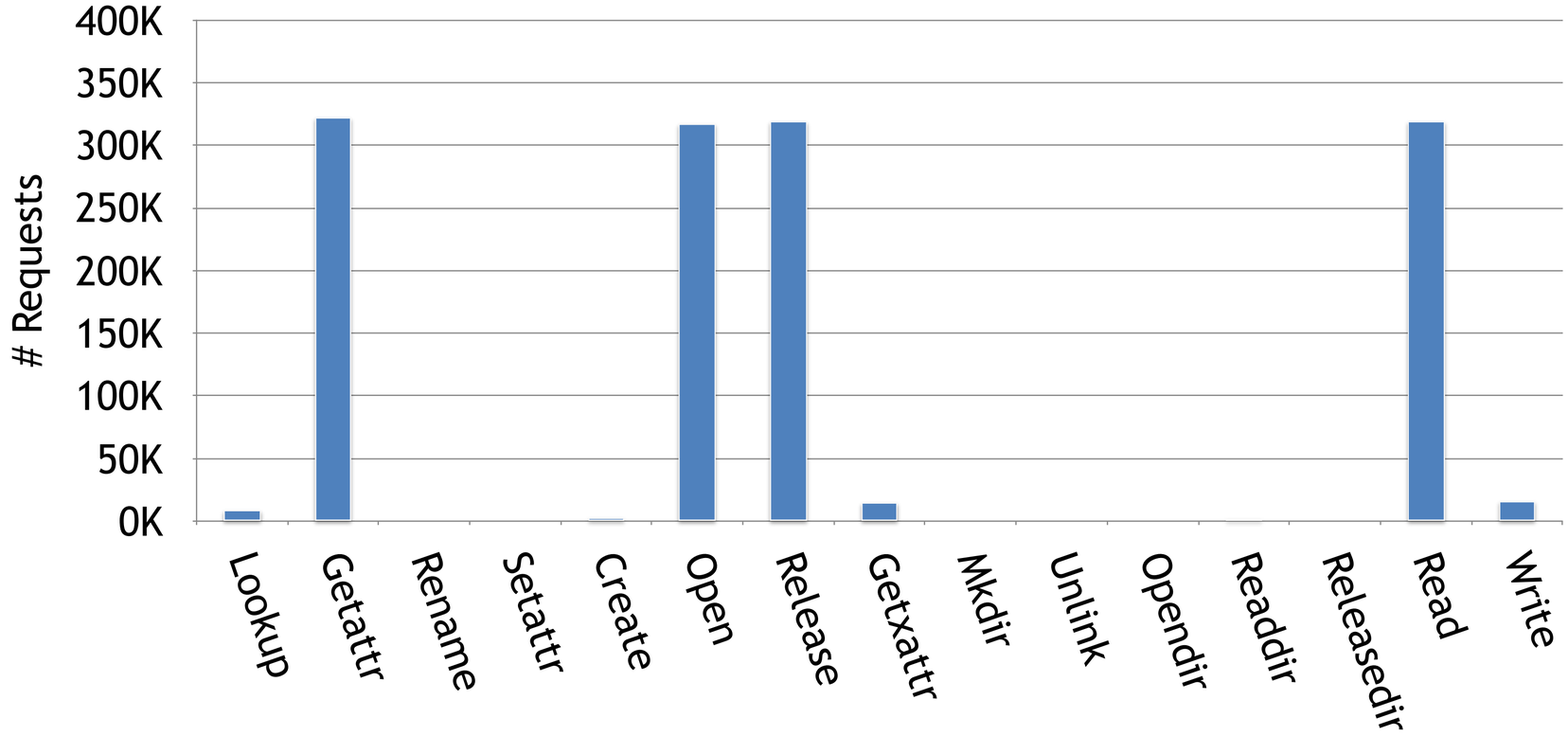


FUSE Performance



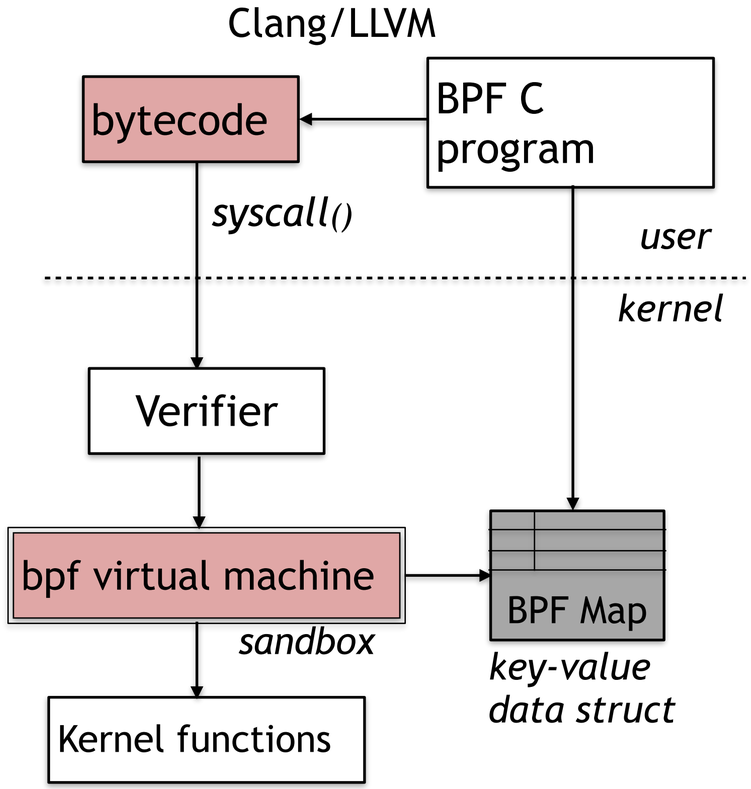
Req received by FUSE

- ***"cd linux-4.17; make tinyconfig; make -j4"***

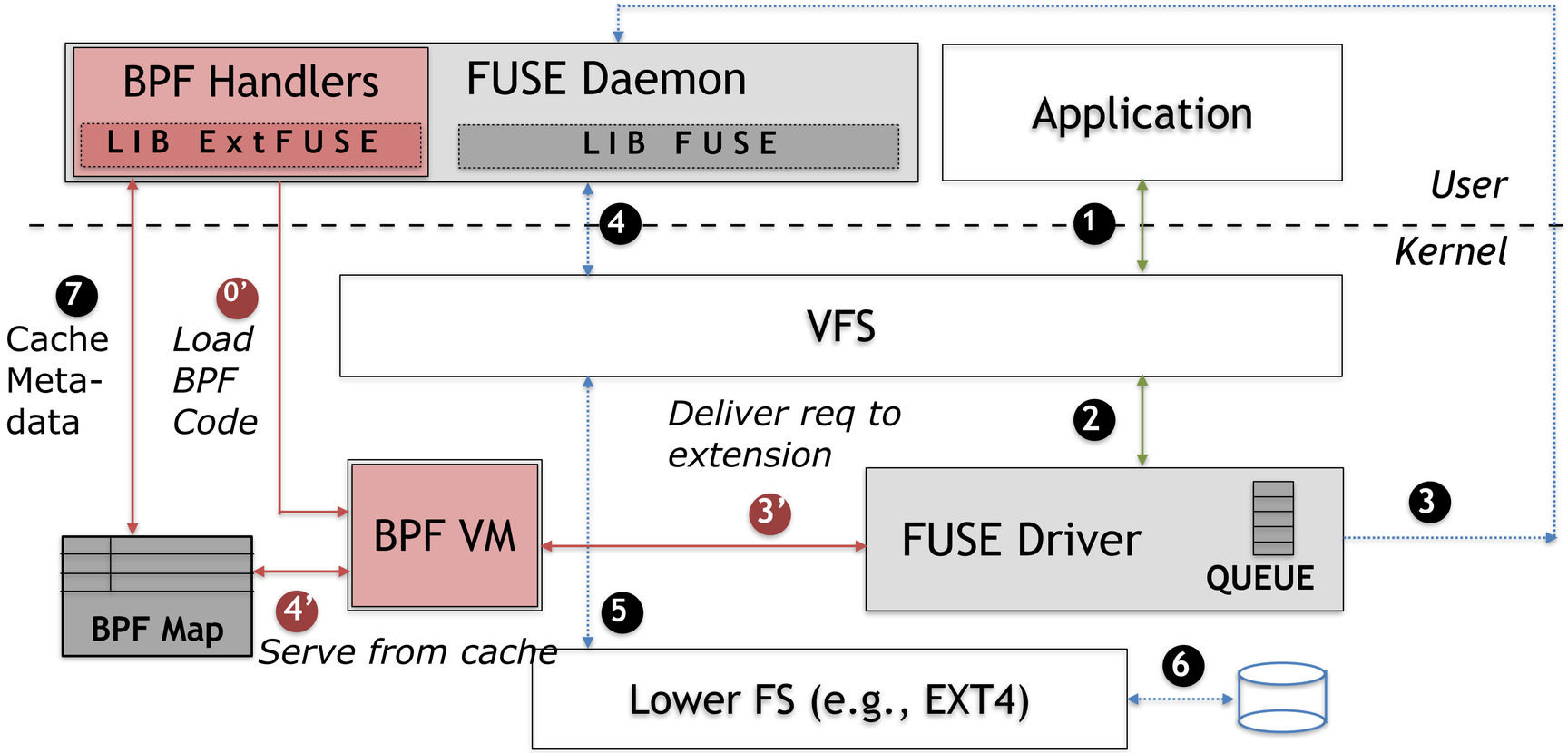


eBPF Overview

- Extensions written in C
- Compiled into BPF code
- Code is verified and loaded into kernel
- Execution under virtual machine runtime
- Shared BPF maps with user space

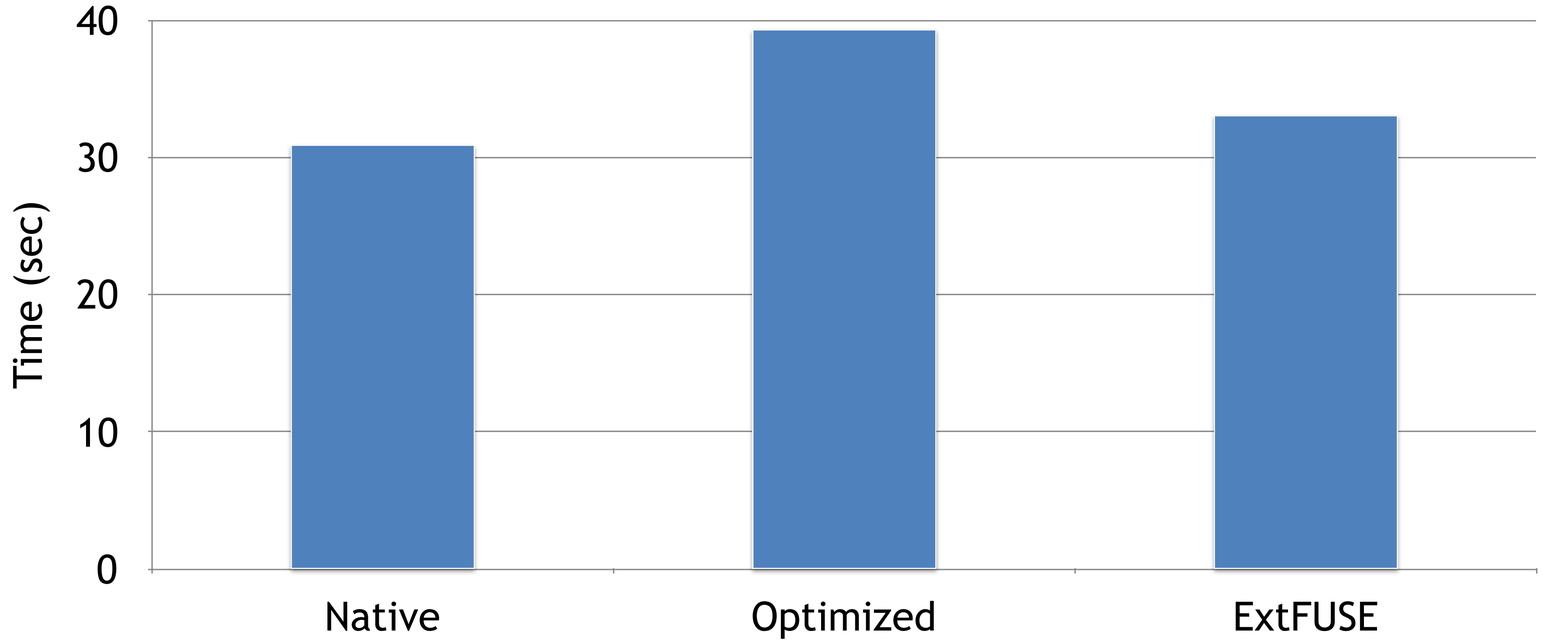


ExtFUSE Architecture



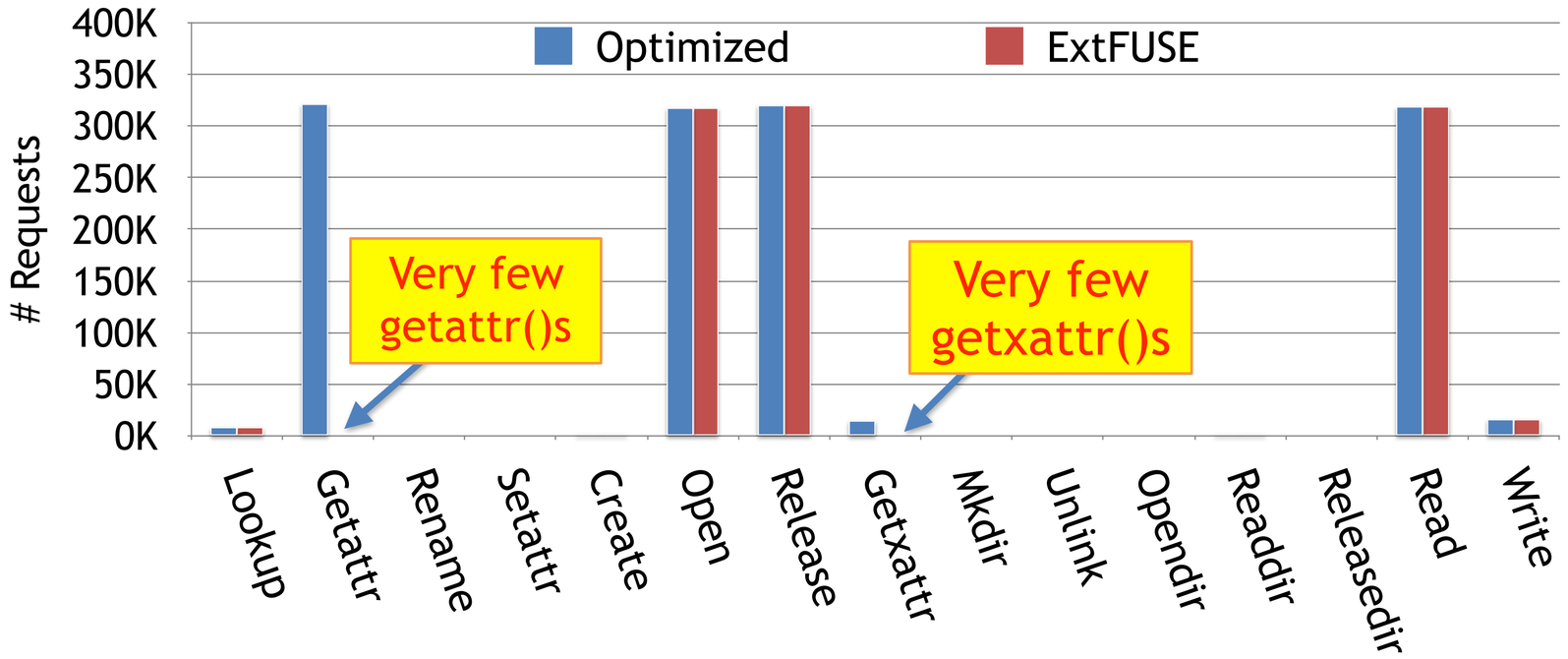
ExtFUSE Performance

- ***"cd linux-4.18; make tinyconfig; make -j4"***
  - Intel i5-3350 quad core, SSD, Ubuntu 16.04.4 LTS
  - Linux 4.11.0, LibFUSE commit # 386b1b



Req received by FUSE

- ***"cd linux-4.17; make tinyconfig; make -j4"***



ExtFUSE Applications

- BPF code to **cache/invalidate meta-data** in kernel
  - Applies potentially to all FUSE file systems
  - e.g., Gluster *readdir ahead* results could be cached
- BPF code to **perform custom filtering or perm checks**
  - e.g., Android SDCardFS *uid* checks in lookup(), open()
- BPF code to **forward I/O requests to lower FS** in kernel
  - e.g., install/remove target file descriptor in BPF map