

# CRFS

## Coherent Remote File System

An introduction to a work in progress.

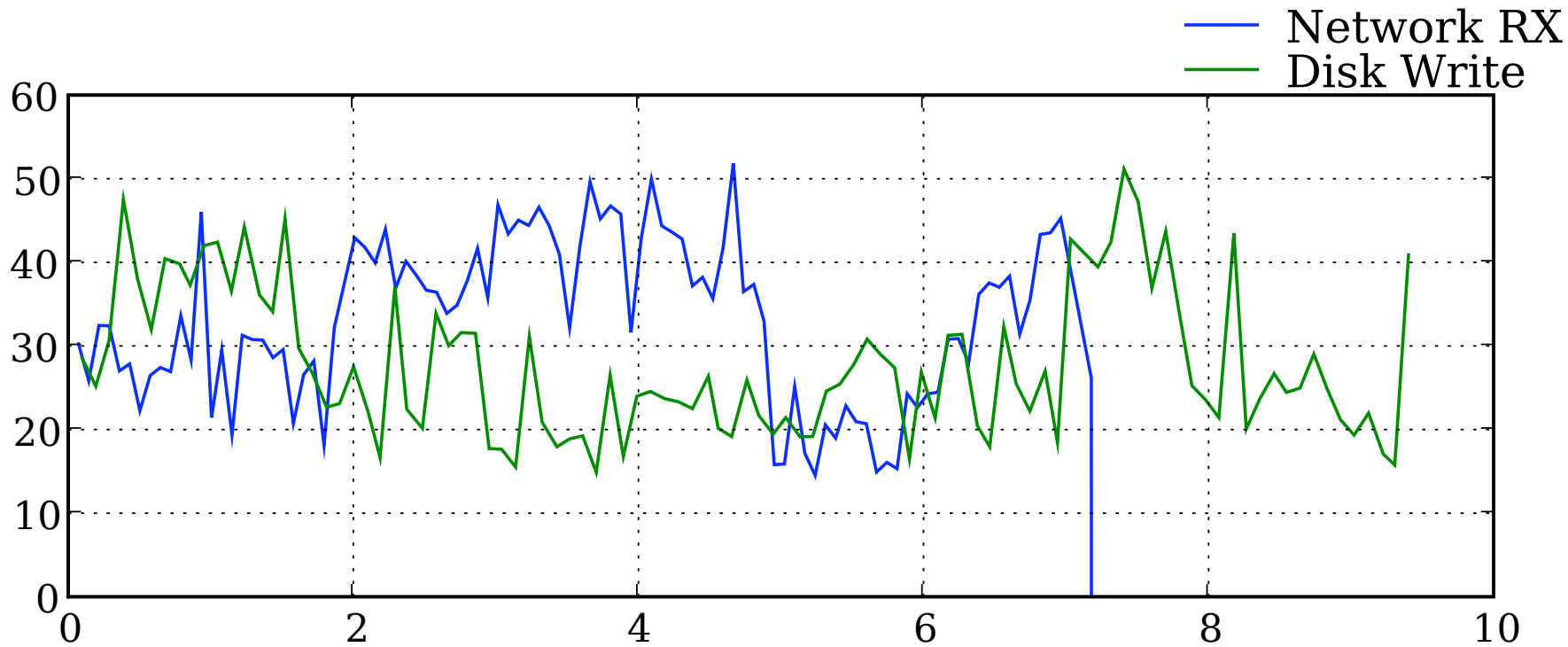
Zach Brown <[zach.brown@oracle.com](mailto:zach.brown@oracle.com)>

[linux.conf.au](http://linux.conf.au)

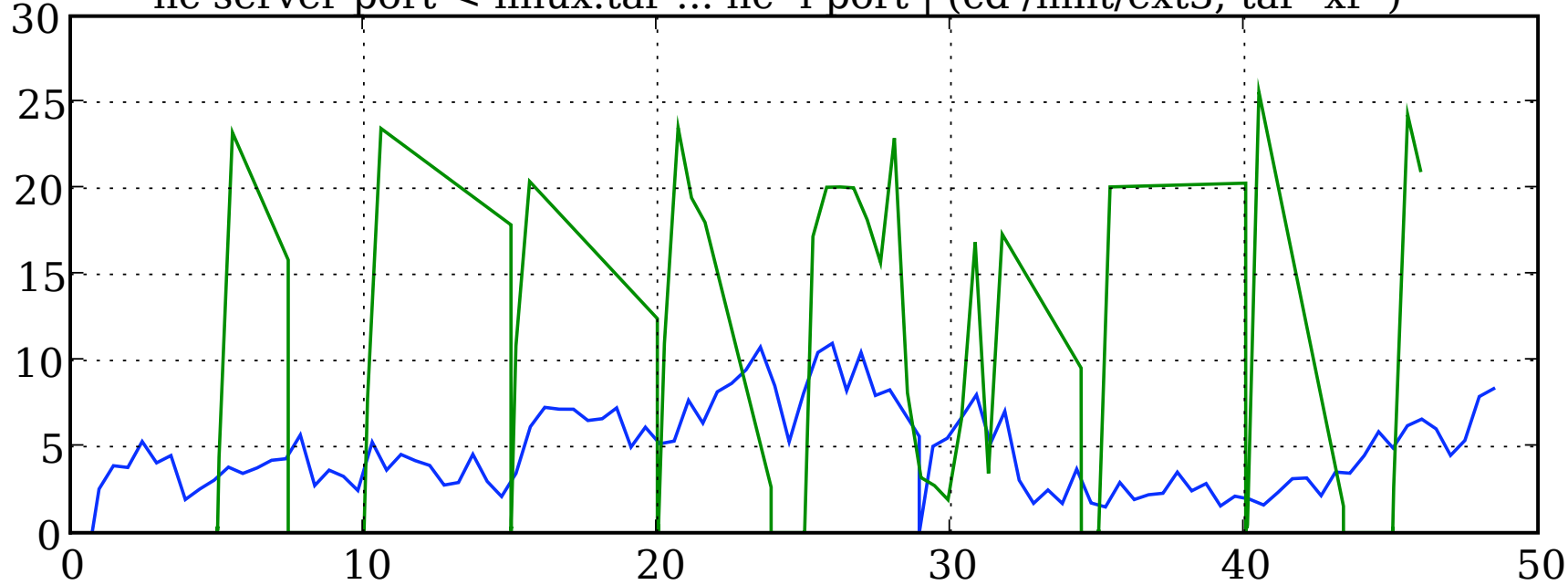
MEL8OURNE2008

# Outline

- Motivation
- System overview
- BTRFS disk format
- CRFS metadata wire format
- Some performance results
- Coherency overview




`nc server port < linux.tar ... nc -l port | (cd /mnt/ext3; tar -xf -)`



`mount server:/mnt/ext3 /mnt/nfs; cd /mnt/nfs; tar -xf linux.tar`


# UNIX (“POSIX”, “local”)



```
mkdir("dir", 0700)           = 0
open("dir", O_RDONLY)       = 3
creat("dir/file", 0600)     = 4
unlink("dir/file")          = 0
rmdir("dir")                 = 0
```

```
getdents(3, 0x7fffe4f404f0, 4096) = -1 ENOENT
```

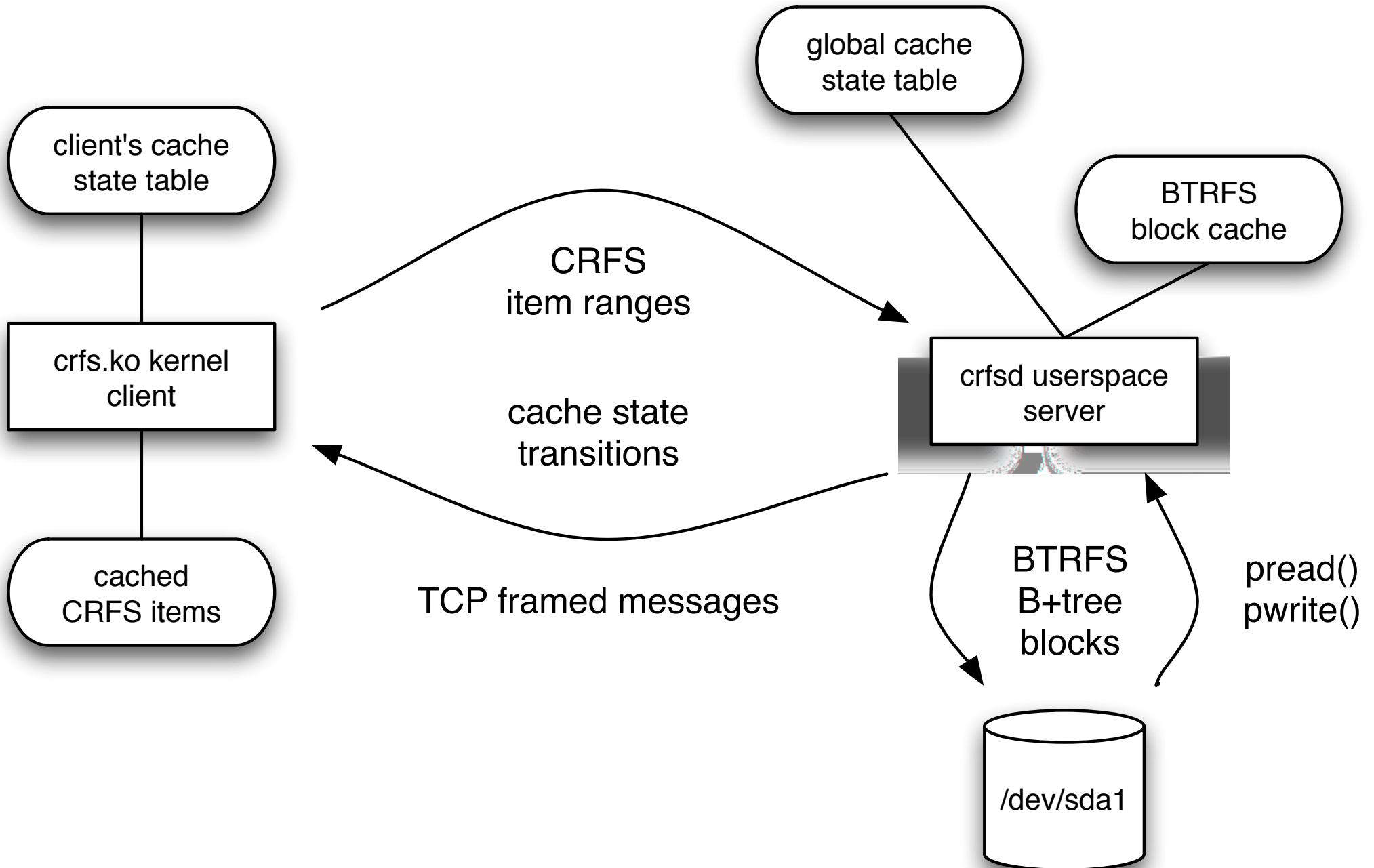
# Linux NFS (“bonghits”)



```
mkdir("dir", 0700)           = 0
open("dir", O_RDONLY)       = 3
creat("dir/file", 0600)     = 4
unlink("dir/file")          = 0
rmdir("dir")                 = -1 ENOTEMPTY
```

```
getdents(3,
  {{d_ino=21825147, d_off=1, d_reclen=24, d_name=".."}}
  {d_ino=21825151, d_off=2, d_reclen=24, d_name="."}
  {d_ino=21825156, d_off=3, d_reclen=48,
    d_name=".nfs00000000014d068400000003c"}}, 4096) = 96
```

# CRFS system overview



# stuff we get from BTRFS

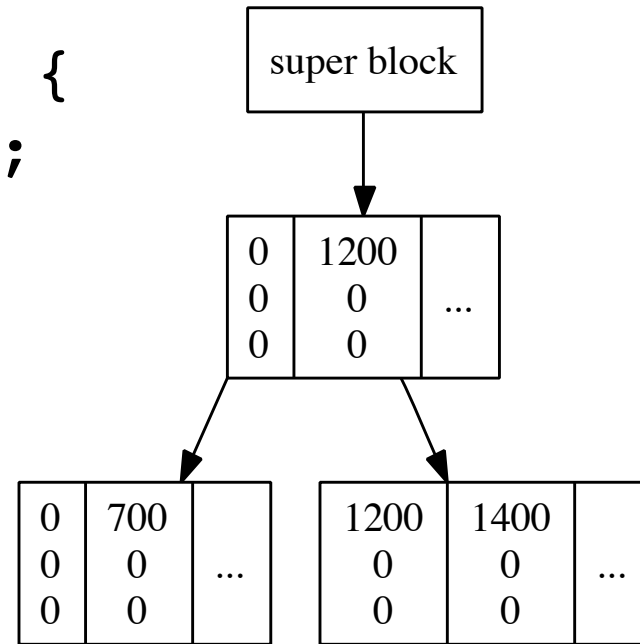
(stuff you can't get from POSIX alone)

- idempotent compound atomic writes
- client-to-drive checksums of file data
- access to unlinked files
- snapshots
- never overwrite referenced data
- walking new data via transaction numbers

```

struct btrfs_key {
    u64 objectid;
    u8  type;
    u64 offset;
};

```



...	...
300 'inode' 0	.st_ino = 300 .st_mode = S_ISDIR
300 'dirent' TEA('hw')	.de_ino = 301 .de_namelen = 2 .de_name[] = 'hw'
301 'inode' 0	.st_ino = 301 .st_mode = S_ISREG .st_size = 11
301 'file data' 0	'hello world'



```

struct crfs_msg_item_query {
    struct crfs_key      key;
    struct crfs_key      min;
    struct crfs_key      max;
    le32                 bytes;
};

struct crfs_item {
    struct crfs_key      key;
    le32                 offset;
    le16                 size;
};

struct crfs_item_range {
    struct crfs_key      min;
    struct crfs_key      max;
    le16                 nr_items;
    struct crfs_item      items[0];
};

```



# empty FS mount query

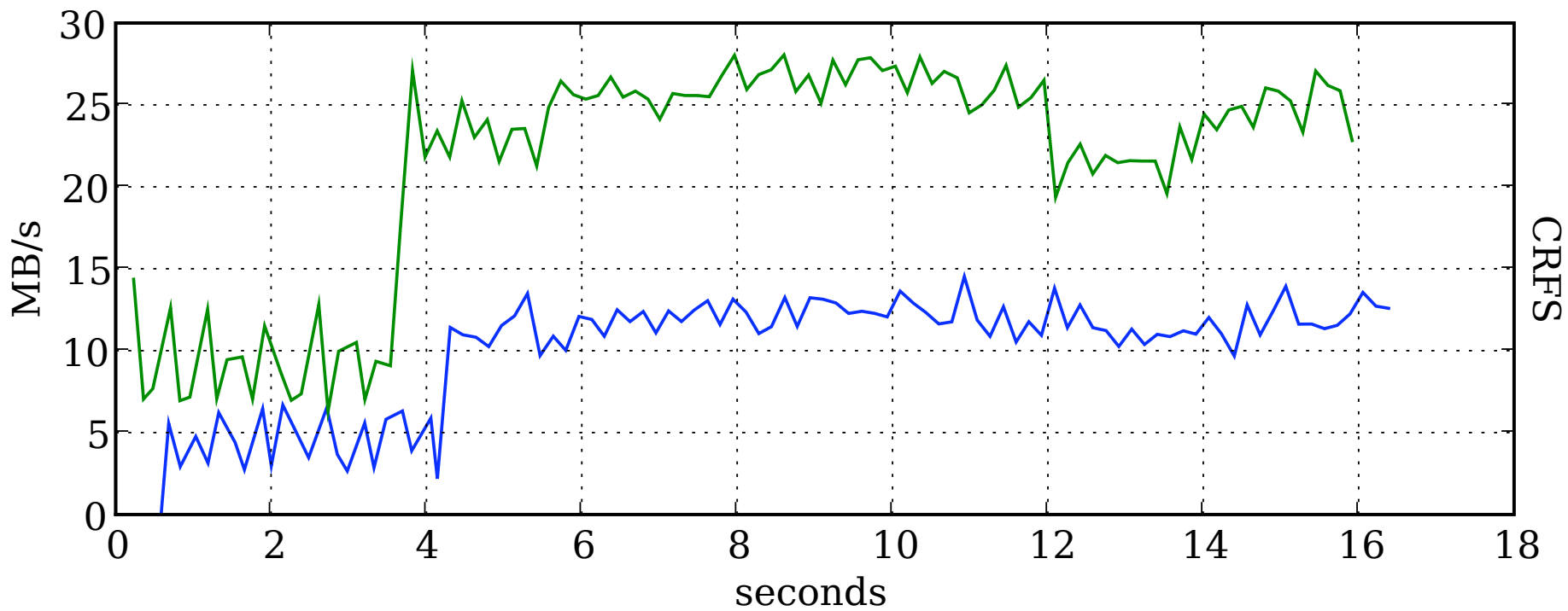
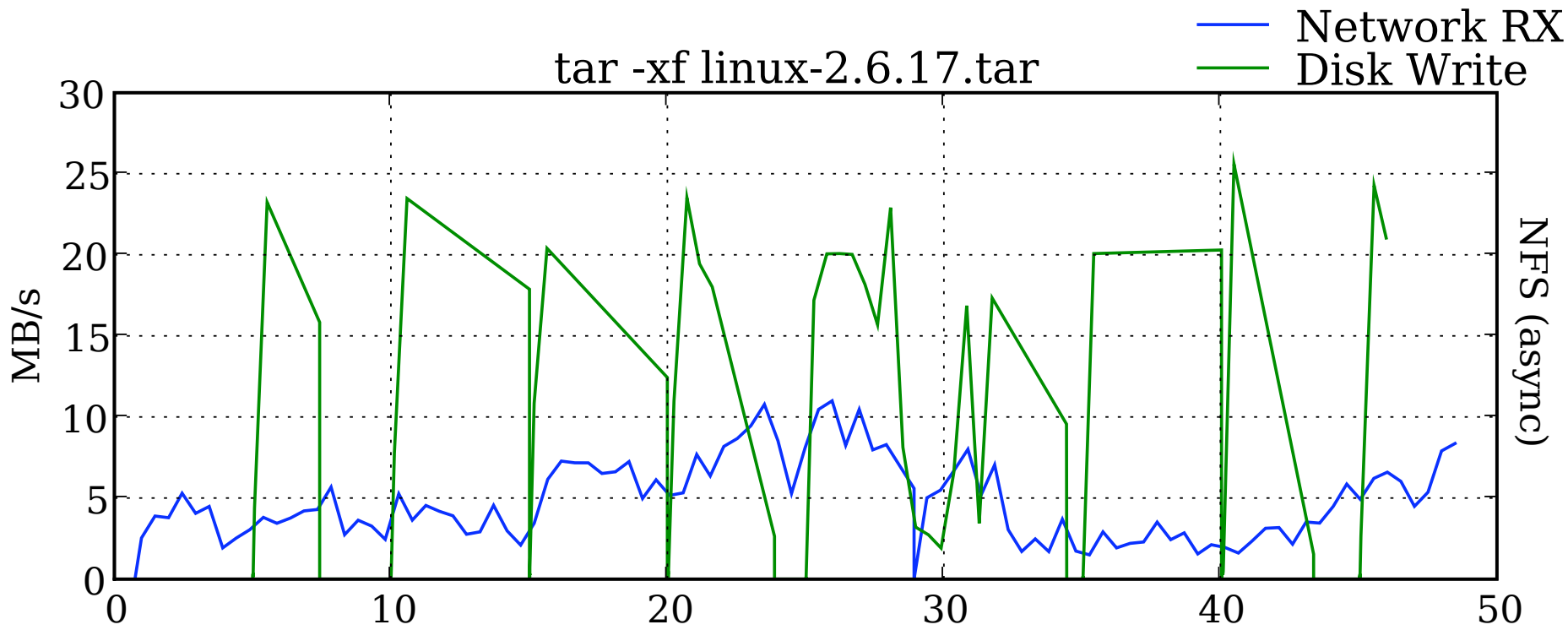
```
header id 1 info 0 bytes 64 type 2 [CRFS_MSG_ITEM_QUERY]
  query key (~0.~0.~0) min (0.0.0) max (~0.~0.~0) bytes 4028
```

```
header id 1 info 0 bytes 356 type 3 [CRFS_MSG_ITEM_RANGE]
  range nr 4 min (5.1.0) max (~0.~0.~0)
    off 104 size 104 key (5.i.0)
      ino sz 42 nlink 1 uid 0 gid 0 mode DIR|0755
    off 208 size 26 key (5.d.0x1)
      dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
    off 234 size 27 key (5.d.0x2)
      dent len 2 type 2 [CRFS_FT_DIR] name '..' loc (5.0.0)
    off 261 size 53 key (5.D.5)
      dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
```

# mkdir transaction

```
header id 2 info 0 bytes 4075 type 6 [CRFS_MSG_TRANS_FRAG]
  frag nr 0 last 0 type 0      range nr 11 min (5.1.0) max (~0.~0.~0)
    off 3818 size 104 key (5.i.0)
      ino sz 42 nlink 2 uid 0 gid 0 mode DIR|0755
    off 3922 size 26 key (5.d.0x1)
      dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
    off 3948 size 27 key (5.d.0x2)
      dent len 2 type 2 [CRFS_FT_DIR] name '..' loc (5.0.0)
off 3681 size 33 key (5.d.0x9371c7261dac5bea)
  dent len 8 type 2 [CRFS_FT_DIR] name 'some-dir' loc (6.i.0)
off 3975 size 53 key (5.D.5)
  dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
off 3648 size 33 key (5.D.6)
  dent len 8 type 2 [CRFS_FT_DIR] name 'some-dir' loc (6.i.0)
off 3714 size 104 key (6.i.0)
  ino sz 2 nlink 2 uid 0 gid 0 mode DIR|0755
off 3622 size 26 key (6.d.0x1)
  dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (6.i.0)
off 3569 size 27 key (6.d.0x2)
  dent len 2 type 2 [CRFS_FT_DIR] name '..' loc (5.i.0)
off 3542 size 27 key (6.D.5)
  dent len 2 type 2 [CRFS_FT_DIR] name '..' loc (5.i.0)
off 3596 size 26 key (6.D.6)
  dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (6.i.0)

header id 2 info 0 bytes 0 type 1 [CRFS_MSG_FINAL_REPLY]
```



# contrived optimal deletion

(but not as unlikely as you might suspect!)

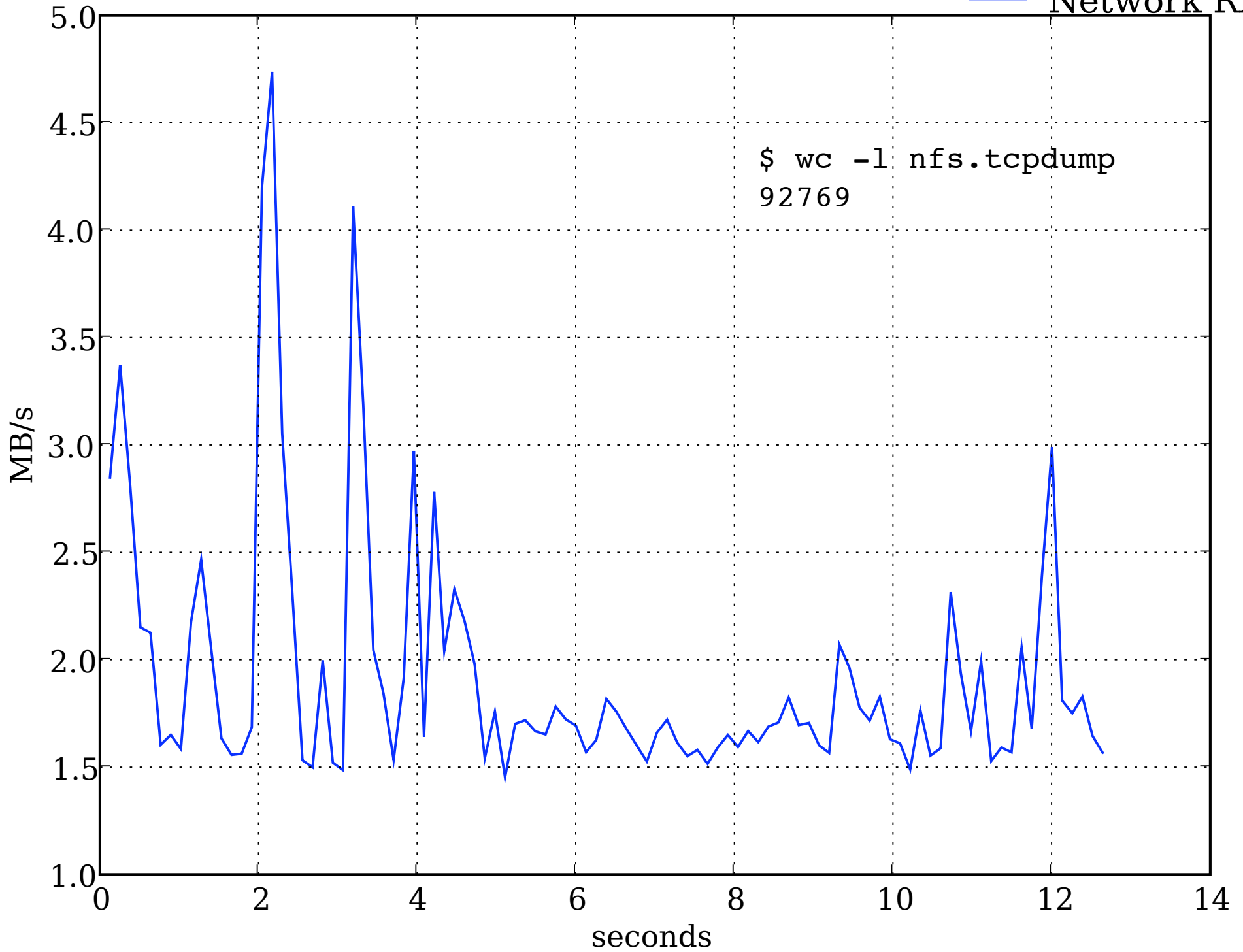
```
mount -t crfs server /mnt/crfs && cd /mnt/crfs
tar -zxf /somewhere/mercurial-0.9.1.tar.gz && sync
crfsdctl snoop > /tmp/output-down-there &
rm -rf mercurial-0.9.1/ && sync
```

```
header id 14 info 0 bytes 4075 type 6 [CRFS_MSG_TRANS_FRAG]
  frag nr 0 last 0 type 0      range nr 4 min (5.1.0) max (~0.~0.~0)
    off 3924 size 104 key (5.i.0)
      ino sz 42 nlink 1 uid 0 gid 0 mode DIR|0755
    off 3898 size 26 key (5.d.0x1)
      dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
    off 3871 size 27 key (5.d.0x2)
      dent len 2 type 2 [CRFS_FT_DIR] name '..' loc (5.0.0)
    off 3778 size 53 key (5.D.5)
      dent len 1 type 2 [CRFS_FT_DIR] name '.' loc (5.0.0)
```

```
header id 14 info 0 bytes 0 type 1 [CRFS_MSG_FINAL_REPLY]
```

# async NFS: rm -rf linux-2.6.17

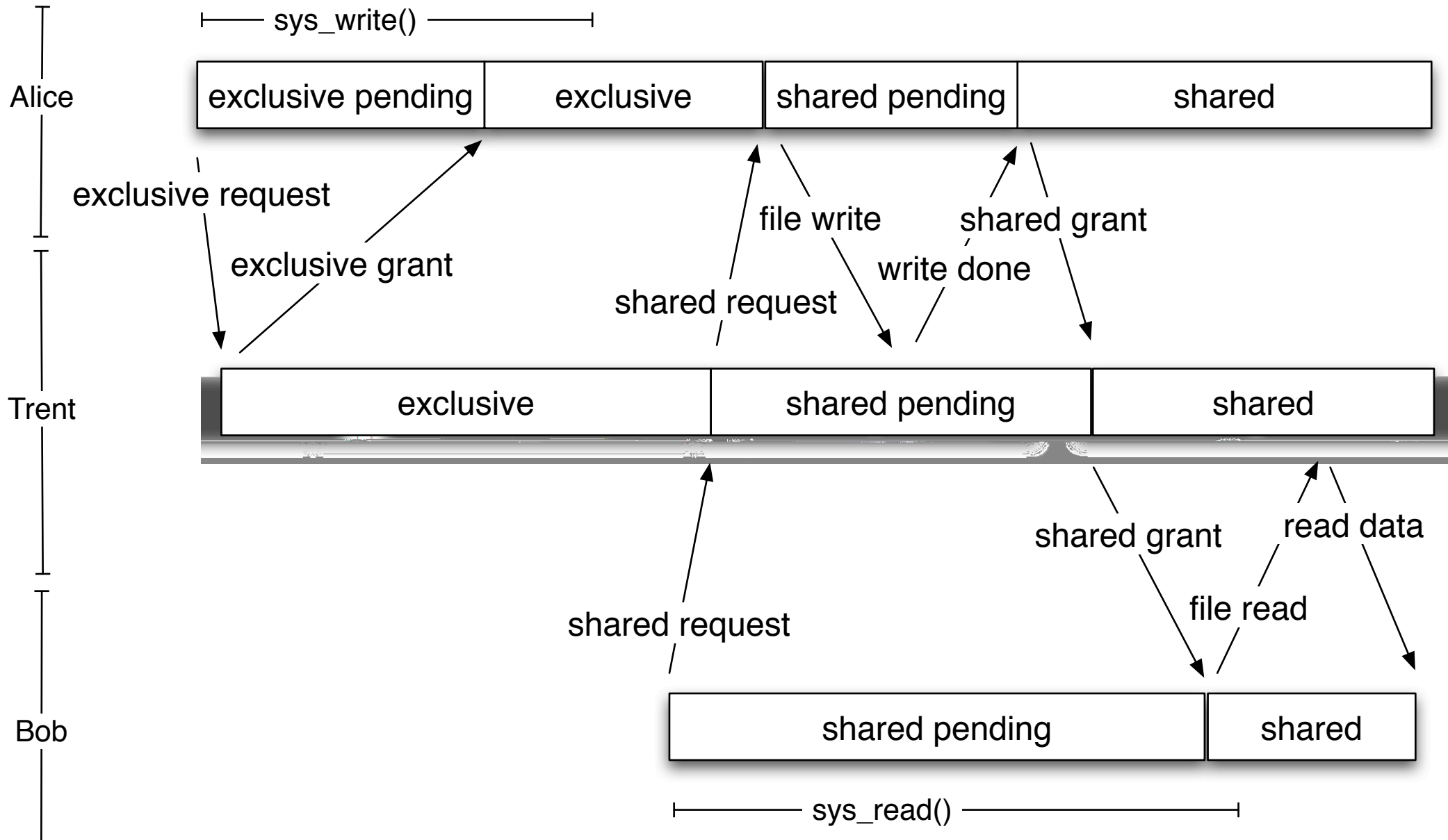
— Network RX





# coherency protocol

(Simplified. No, seriously.)



# security hand-waving

- currently untranslated UID/GID in the clear
- will secure transport (TLS? MAC only?)
- will inspect transactions to enforce policy
- identity and capability metadata on disk and in transactions could enable untrusted clients
- crypto on disk could let clients hide data from crfsd (maybe ecryptfs metadata?)



# TODO: cris-atunity!

BTRFS multi-device support      BTRFS allocation policy      SSD tuning

man pages      incremental backups      wireshark      crfsd UNIX IO

OS X      cache coherency      automated QA      xattrs      failover

BSD      hot spare read load balancing      FS metadata crypto      snapshots

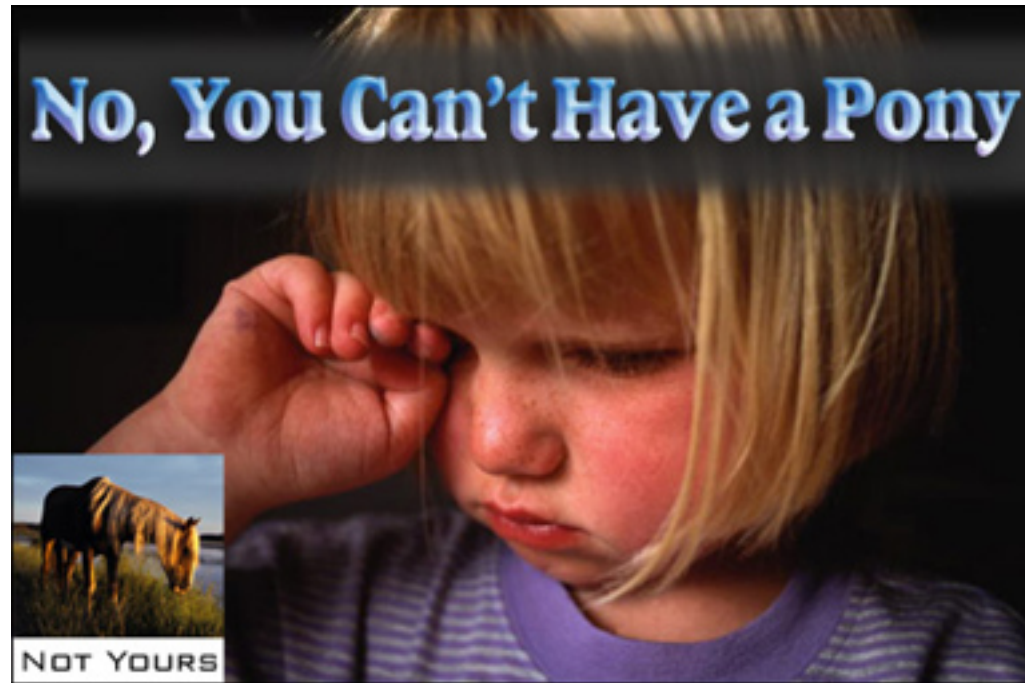
crfsd parallelism      ENOSPC credits      crfsd cache reccounting      howtos

online repair      crfs.ko cache data structure      encrypted user data      FUSE

UID mapping      crfs.ko allocation policy      SMP tuning      tcpdump

WAN tuning      transport crypto      transaction auditing      IETF

BTRFS volume resizing      statistics      binary packages      crfsd config



<http://oss.oracle.com/projects/crfs/>

```
$ wc -l lk/*.ch | tail -1  
7335 total
```

```
$ wc -l crfsd/*.ch | tail -1  
5971 total
```