FOSDEM'25

usb9pfs

network booting without the network

Michael Grzeschik - m.grzeschik@pengutronix.de
Ahmad Fatoum – a.fatoum@pengutronix.de

Pengutronix.

https://www.pengutronix.de

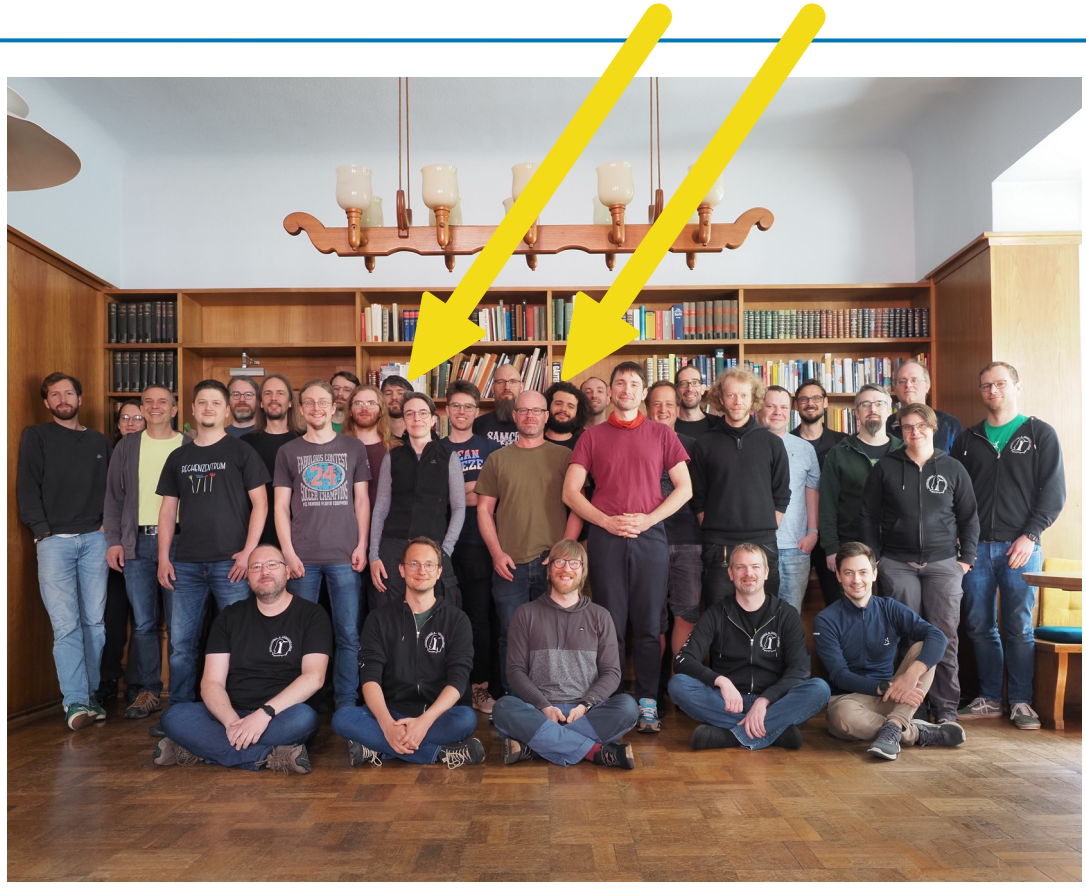
About Us

Michael Grzeschik Ahmad Fatoum

Pengutronix

mgrzeschik a3f

@mgr@nrw.social @a3f@fosstodon.org



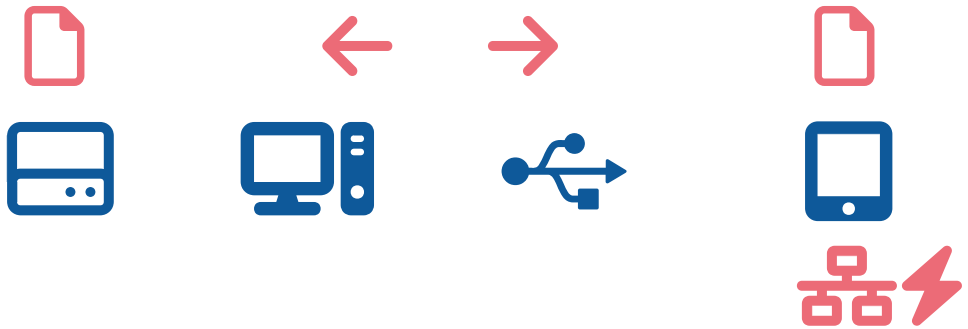
- Kernel and Bootloader Porting
- Driver and Graphics Development
- System Integration
- Embedded Linux Consulting

Show of Hands

- Who is not already regularly network booting their system?
- Why not?
  - No suitable rootfs?
  - Permission/SUID issues?
  - No network interface?
  - Network interface used otherwise?
- Anybody used 9pfs?

Goal

- Transport 9pfs via usb gadget
  - Host is exporting an directory to some gadget



9pfs in the wild

- rootfs mirror via virtio-9p-device in qemu
- 9pfs transports in the kernel (net/9p/trans_*.c)
  - fd (tcp), virtio, rdma, Xen
- TCP exporting servers
  - nfs-ganesha (https://github.com/nfs-ganesha/nfs-ganesha)
  - diod (https://github.com/chaos/diod)

9pfs (9P2000.L) / styx

- File representation of something in some local namespace
- Set of transport request/response functions
  - https://github.com/chaos/diod/blob/master/protocol.md
- Simple operations specified
  - version, flush, walk, read, write, clunk, attach, auth
    lopen, lcreate, symlink, mknod, rename, readlink, getattr,
    setattr, xattrwalk, xattrcreate, readdir, fsync, lock, getlock, link, renameat, unlinkat
- Restrictions:
  - Transport must be reliable (e.g. no udp)
  - Messages must not get transposed (request x → response x)

USB

- Two bulk endpoints (likewise to serial gadget TX/RX)
  - One ep for 9p requests
  - One ep for 9p responses
- Allocate one usb_request per endpoint (simpler)
  - Every transmitted usb tx request waits
    for its corresponding usb rx response to complete
- 9p requests are getting queued in a linked list

usb9pfs

- New transport driver in the linux kernel
  - net/9p/trans_usbg.c
  - in kernel v6.12
- /usr/src/linux/tools/usb/p9_fwd.py
  - for tcp ‹-› usb translation

usb9pfs – Glue (struct p9_trans_module)

```
struct p9_trans_module p9_usbg_trans = {
 .name    = "usbg",
 .create  = p9_usbg_create,   // attach an client to some
                              // transport (e.g. mount)
 .close   = p9_usbg_close,    // dettach
 .request = p9_usbg_request,  // queue an 9p request in to the transport
 .cancel  = p9_usbg_cancel,   // quit all current transactions
 .owner  = THIS_MODULE,
};
```

usb9pfs – Glue (struct usb_function)

```
function->name = "usb9pfs";                // name
function->bind = usb9pfs_func_bind;        // allocate (endpoints, requests)
                                           //   assign descriptors
function->unbind = usb9pfs_func_unbind; // deallocate, unassign
function->set_alt = usb9pfs_set_alt;    // enable usbpfs
function->disable = usb9pfs_disable;    // clear pending transfers
```

Mount example

- gadget side:

```
$ gt load usb9pfs.scheme (libusbgx)
$ mount -t 9p -o trans=usbg,aname=/path/to/fs <device> /mnt/9
```

- host side:

```
$ diod -f -n -d 0 -S -l 0.0.0.0:9999 -e $PWD
$ python $kernel_dir/tools/usb/p9_fwd.py (--path) connect -p 9999
$ python $kernel_dir/tools/usb/p9_fwd.py list
 Bus | Addr | Manufacturer     | Product          | ID        | Path
 --- | ---- | ---------------- | ---------------- | --------- | ----
   2 |   67 | unknown          | unknown          | 1d6b:0109 | 2-1.1.2
   2 |   68 | unknown          | unknown          | 1d6b:0109 | 2-1.1.3
```

Beyond and before the Kernel

- How is the kernel loaded?
  - Preferably via a USB gadget protocol too

    → Port 9pfs in the bootloader!
- Who mounts the rootfs?
  - Non-legacy gadgets need configfs setup from userspace
- How do we generate the rootfs?

barebox: booting from usb9pfs

- Ported 9pfs with Virt I/O and USB gadget transports
- Make good use of barebox file system conveniences:
  - automount enumerated file systems on first access
  - file system driver computes root= for Linux command line

```
barebox@Embest MarS Board i.MX6Dual:/ automount -l
/mnt/mmc2.1           mount mmc2.1
/mnt/nfs              ifup -a1 && mount -t nfs
                      ${global.net.server}:/home/${global.user}/nfsroot/${global.hostname}
                      /mnt/nfs
/mnt/9p/ci_hdrc.0     mount -t 9p -o trans=usbg,
                      aname=/home/${global.user}/nfsroot/${global.hostname}
                      ci_hdrc.0 /mnt/9p/ci_hdrc.0
```

```
barebox@Embest MarS Board i.MX6Dual:/ boot /mnt/9p/ci_hdrc.0
```

rsinit: mounting 9pfs as rootfs

- "a minimalistic single binary init for the initramfs for embedded systems"
- Parses the Kernel command line to:
  - mount dm-verity device
  - setup NFS root
  - mount 9pfs over Virt I/O and USB gadget

```
export RUSTFLAGS='-C link-arg=-s -Clinker=arm-linux-gnueabihf-ld'
cargo build --profile=minimal --target armv7-unknown-linux-musleabihf
tmpdir=$(mktemp -d)
cp target/armv7-unknown-linux-musleabihf/minimal/init $tmpdir/
cd $tmpdir; find . | cpio -o -H newc | gzip > $outdir/rsinit.cpio.gz
```

poky-nfsroot: export OE FS under pseudo

- Unpacking rootfs tarball without sudo problematic
- runqemu-export-rootfs: runs unfsd under pseudo's fake root environment
- poky-nfsroot: update rootfs automatically as soon as new packages are built

```
$ nfs-export-updater --debug reference-base-image nfsroot
INFO: Update from package feeds
INFO: Installing new packages...
INFO: Packages to install: ['kernel-image'].
INFO: Upgrading packages...
```

diod: do not pierce the fakeroot veil

- diod needed patching, so it runs under fakeroot
- Still need to discuss upstream if that's the best way to go about it...

```
$ diod -c /dev/null -f -n -N  -l 0.0.0.0:3048 \
    -e /home/a3f/nfsroot/marsboard
```

It's demo time

Future Outlook

- **barebox**
  - Upstream 9pfs support
- **Linux**
  - Allow more in-flight requests
  - Improve robustness
  - bootconfig as USB gadget configfs alternative?
- **rsinit**
  - Merge platsch(1) functionality (early DRM boot splash)
  - Package for distros and build systems
- **OpenEmbedded-core**
  - Upstream poky-nfsroot
  - Contribute diod-native recipe
  - Upstream runqemu-export-rootfs diod support
- **/usr/src/linux/tools/usb/p9_fwd.py**
  - Improve detection of gadget
- **diod**
  - Upstream patches to make it dumber
- **Labgrid**
  - make usb9pfs a resource?

Interested? Follow the progress at

https://github.com/mgrzeschik/usb9pfs

Questions?