Agenda

- The Linux NFC architecture
- The Linux NFC APIs
- IVI specific uses cases and implementation
The Linux NFC Architecture
NFC 101

- Very short range, very low latency, very cheap
- Tags and devices
- 3 modes: Reader, p2p or card emulation
- 4 tags family, all linked to a specific manufacturer.
- NFC Forum specified
- NDEFs carry NFC payloads
Design Objectives

- HW independence.
- POSIX NFC APIs.
- Consistent behavior and simple APIs.
- Open development process.
- NFC for non Android platforms.
- Kernel/User space split.
The overall picture

User space

Kernel space

NFC Hardware

NFC Drivers

NFC Core

AF_NFC Sockets

NFC Netlink

neard

D-Bus API

application

application
Neard - The NFC daemon

- Tag specific handling (R/W).
- Transport protocols on top of LLCP.
- Adapter and targets management.
- NDEF parsing.
- Handover.
- D-Bus APIs.
- Plugin based.
The Linux NFC APIs
Design Objectives

- System API
  - Provide a standard Linux networking API
  - Provide a netlink based out of band API
- D-Bus API
  - Only expose what the apps need
  - Simple API for a simple and intuitive technology
  - Offload the non NFC bits to the experts (BlueZ, ConnMan)
System API

- A dedicated socket domain – AF_NFC
  - Fully standard UNIX networking API
  - 2 socket protocols: LLCP and RAW
  - Connection less and connection oriented support
- A generic netlink name for NFC
  - Out of band operations
  - Polling, Service Name Lookups, Secure Element handling, NFC events, etc..
D-Bus API

- Simple, consistent and intuitive API
- Focus on data, not on technology
- Minimal exposure to technology details
- Tight interaction with other daemons
- Similarities with BlueZ or ConnMan APIs
NFC app (NDEF agent)

ConnMan (handover agent)

BlueZ (handover agent)

manager

adapter /nfc0

tag /nfc0/tag0

device /nfc0/device0

record /nfc0/device0/record0

adapter /nfc1

device /nfc1/device0
D-Bus API – Manager and Adapters

• Manager is the top level API
  • Handle adapters
  • Handle agents registration
• An adapter represents an NFC controller
  • Polling methods
  • Powered and Polling properties
  • Devices and Tags arrays
    – Detected peers and tags (typically only one single entry)
D-Bus API – Devices and Tags

- Adapters properties
- NDEF records array property
- No read method
  - Read is automatically done at Tag detection time
  - Devices records gets update as data is received
- Data is written to Tags, pushed to Devices
D-Bus API - Records

- One NDEF might contain several records
- Tags and Devices hold an array of Records
- Avoid NFC specific details
- Mostly a list of properties exposing data and meta-data
D-Bus API – Handover Agent

- Split between NFC and WiFi or Bluetooth
- Neard only handles the NFC bits
- Agent role:
  - Implement PushOOB and RequestOOB
  - Register as a handover agent for a specific carrier
  - Initiate pairing/association when needed
D-Bus API – Handover Agent

• Handover reception
  • The peer is the handover initiator
  • Neard asks the agent for OOB data: RequestOOB
  • Neard builds and sends the reply to the peer

• Handover transmission
  • Neard is the handover initiator
  • Neard asks the agent for OOB data: RequestOOB
  • Neard builds and send the handover request frame
  • Neard receives the handover reply from the peer
  • Neard forwards the OOB data reply: PushOOB
D-Bus API – NDEF Agent

- Allows for proprietary NDEF handling
- Agent registers for a certain NDEF type
- Agent implement GetNDEF to retrieve the raw NDEF data
IVI use cases and implementations
Handover focus

- Seamless Bluetooth and WiFi pairing/association
- Bluetooth main device selection
- Binding car seats to headsets
- Wireless charging
Bluetooth pairing implementation

● Upstream BlueZ is a neard handover agent
● D-Bus implementation:
  ● Listen on Adapter.PropertyChanged
  ● When a new device shows up:
    - Device.Push({Type="Handover", Carrier="bluetooth")
  ● http://git.kernel.org/?p=network/nfc/neard.git;a=blob_plain;f=test/bt-handover;hb=HEAD
Implementation status

- Bluetooth pairing implemented
- WiFi handover ready in 3-4 weeks
- Main device selection needs some BlueZ plumbing
- Wireless charging is not an official spec yet
Questions?

• NFC daemon
  http://git.kernel.org/?p=network/nfc/neard.git;a=summary

• NFC kernel
  http://git.kernel.org/pub/scm/linux/kernel/git/sameo/nfc-3.0.git

• Web site
  • https://www.01.org/linux-nfc

• Mailing list
  https://lists.01.org/mailman/listinfo/linux-nfc

• sameo@linux.intel.com