Agenda

- What is NFC
- Existing stacks
- Linux NFC project
- Development tools
- Future developments
What is NFC?
NF What?

- Stands for Near Field Communication
- Short distance communication
- Limited bandwidth
Devices
Tags

- 5 different tag types
- Devices can read and write
- Dedicated usage for a tag
Use cases
Reading tags
Peer to Peer

- Communication between 2 devices
- Data exchange
- Handover
Host Card Emulation

- Device simulates a tag
- Payment, ticketing, authentication
- Emulation through Secure Elements
Existing solutions
Android

- User space stack
- NXP and Broadcom chipsets
- HCI and NCI support
- Maintained by Broadcom and Google
- Support all tag types
- Supports LLCP and HCE modes
libnfc

- Academic LGPL licensed project
- USB and UART devices support
- Community supported
- Hosted on Github
nfcpy

- Written in python
- Nice implementation
- Supported by Sony
- No HCI or NCI support
- USB dongles only
Linux NFC Project
Linux NFC

• Hardware independence
• POSIX NFC APIs
• Licensed under GPL v2
• Kernel and user space separation
• D-Bus APIs
• Maintained by Samuel Ortiz
Architecture diagram

User space

Kernel space

D-Bus API

seeld

neard

Applications

NFC Netlink

AF_NFC sockets

NFC Core

NFC Drivers

NFC Hardware
# Kernel architecture

<table>
<thead>
<tr>
<th>Netlink socket</th>
<th>AF_NFC sockets</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTO_LLCP</td>
<td>PROTO_RAW</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NFC Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCI</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NFC Drivers</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB</td>
</tr>
</tbody>
</table>

| NFC Hardware |
Kernel architecture

- Drivers can register against:
  - The core stack directly
  - The HCI backend
  - The NCI backend
  - The Digital protocol layer
- User space commands go through netlink
- Dedicated socket domain for data transfer
User space architecture
User space architecture

- User space daemon “neard”
- Secure elements daemon “seeld”
- Plugin based
- API Abstraction using D-Bus
- Tag reading & writing
- NDEF parsing
- LLCP protocols
- WiFi and BT handover
Development Tools
Hardware Simulation nfcsim

- Special driver declaring 2 virtual devices
- Supports NFC-DEP protocol only
- Act as a loopback device
  - Everything sent from one device is sent back to the other one.
- Useful to debug the whole stack
nfctool

- List attached devices
- Enable / disable devices
- Start / stop poll
- Set connection parameters
  - Link timeout
  - Receive window
  - Max information unit extension
The sniffer

- Monitor LLCP traffic
- Support decoding for
  - LLCP
  - SNEP
  - NDEF
- Digital Layer frames decoding coming soon
The sniffer in action

```
$ ./tools/nftool/nftool -d nfc0 -n
Start sniffer on nfc0

<< nfc0: local:0x20 remote:0x01
  Connect (CONNECT)
  Service Name: urn:nfc:sn:snap
  Maximum Information Unit Extensions: 2047
  Receive Window Size: 15

>> nfc0: local:0x20 remote:0x04
  Connection Complete (CC)
  Maximum Information Unit Extensions: 2047
  Receive Window Size: 15

<< nfc0: local:0x20 remote:0x04
  Information (I)
  N(S):0 N(R):0
  Simple NDEF Exchange Protocol (SNEP)
  Version: 1.0
  Request: Put
  NDEF Record
    Message Begin: True
    Message End: True
    Chunk Flag: False
    Short Record: True
    ID Length present: False
    Type Name Format: NFC Forum well-known type [NFC RTD] (0x01)
    Type Length: 1
    Payload Length: 14
    Type: 0000: 54 |T|
    Payload:
      0000: 02 65 6E 48 65 66 66 6F 20 77 6F 72 6C 64 |"enHello world"|
```

...
Future Developments

- NCI User channel
- Host Card Emulation
Q & A

- nfc-next kernel
  https://git.kernel.org/cgit/linux/kernel/git/sameo/nfc-next.git

- nfc-next-stable kernel
  https://github.com/tescande/linux-nfc-next-stable

- Daemons
  https://git.kernel.org/cgit/network/nfc/neard.git

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

- IRC
  #linux-nfc on freenode