Linux NFC subsystem

The Near Field Communication (NFC) subsystem is required to standardize the NFC device drivers development and to create an unified userspace interface.

This document covers the architecture overview, the device driver interface description and the userspace interface description.

Architecture overview

The NFC subsystem is responsible for:

- NFC adapters management;
- Polling for targets;
- Low-level data exchange;

The subsystem is divided in some parts. The 'core' is responsible for providing the device driver interface. On the other side, it is also responsible for providing an interface to control operations and low-level data exchange.

The control operations are available to userspace via generic netlink.

The low-level data exchange interface is provided by the new socket family PF_NFC. The NFC_SOCKPROTO_RAW performs raw communication with NFC targets.

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\networking\(linux-master) (Documentation) (networking)nfc.rst`, line 29)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        +--------------------------------------+
        |              USER SPACE              |
        +--------------------------------------+
            ^                       ^
            | low-level             | control
            | data exchange         | operations
            |                       |
            |                       v
            |                  +-----------+
            | AF_NFC           |  netlink  |
            | socket           +-----------+
            | raw                   ^
            |                       |
            v                       v
        +---------+            +-----------+
        | rawsock | <--------> |   core    |
        +---------+            +-----------+
                                    ^
                                    |
                                    v
                               +-----------+
                               |  driver   |
                               +-----------+
```

Device Driver Interface

When registering on the NFC subsystem, the device driver must inform the core of the set of supported NFC protocols and the set of ops callbacks. The ops callbacks that must be implemented are the following:

- start_poll - setup the device to poll for targets
- stop_poll - stop on progress polling operation
- activate_target - select and initialize one of the targets found
- deactivate_target - deselect and deinitialize the selected target
- data_exchange - send data and receive the response (transceive operation)

Userspace interface

The userspace interface is divided in control operations and low-level data exchange operation.

CONTROL OPERATIONS:

Generic netlink is used to implement the interface to the control operations. The operations are composed by commands and events, all listed below:

- NFC_CMD_GET_DEVICE - get specific device info or dump the device list
- NFC_CMD_START_POLL - setup a specific device to polling for targets
- NFC_CMD_STOP_POLL - stop the polling operation in a specific device
- NFC_CMD_GET_TARGET - dump the list of targets found by a specific device
- NFC_EVENT_DEVICE_ADDED - reports an NFC device addition
- NFC_EVENT_DEVICE_REMOVED - reports an NFC device removal
- NFC_EVENT_TARGETS_FOUND - reports START_POLL results when 1 or more targets are found

The user must call START_POLL to poll for NFC targets, passing the desired NFC protocols through NFC_ATTR_PROTOCOLS attribute. The device remains in polling state until it finds any target. However, the user can stop the polling operation by calling STOP_POLL command. In this case, it will be checked if the requester of STOP_POLL is the same of START_POLL.

If the polling operation finds one or more targets, the event TARGETS_FOUND is sent (including the device id). The user must call GET_TARGET to get the list of all targets found by such device. Each reply message has target attributes with relevant information such as the supported NFC protocols.

All polling operations requested through one netlink socket are stopped when it's closed.

LOW-LEVEL DATA EXCHANGE:

The userspace must use PF_NFC sockets to perform any data communication with targets. All NFC sockets use AF_NFC:

```
struct sockaddr_nfc {
       sa_family_t sa_family;
       __u32 dev_idx;
       __u32 target_idx;
       __u32 nfc_protocol;
};
```

To establish a connection with one target, the user must create an NFC_SOCKPROTO_RAW socket and call the 'connect' syscall with the sockaddr_nfc struct correctly filled. All information comes from NFC_EVENT_TARGETS_FOUND netlink event. As a target can support more than one NFC protocol, the user must inform which protocol it wants to use.

Internally, 'connect' will result in an activate_target call to the driver. When the socket is closed, the target is deactivated.

The data format exchanged through the sockets is NFC protocol dependent. For instance, when communicating with MIFARE tags, the data exchanged are MIFARE commands and their responses.

The first received package is the response to the first sent package and so on. In order to allow valid "empty" responses, every data received has a NULL header of 1 byte.