Using Rust for your network management tools

Let the crabs control the packets!

Fernando F. Mancera
Senior Software Engineer
What we’ll discuss today

- Network management
- Library and binary
- Serde
- Zbus
- Nispor & Netlink
- Sockets
- Bindings everywhere!
Network Management

Done easy with NetworkManager and Nmstate

Network management is a process that requires userspace and kernelspace coordination to configure the desired network state.

- NetworkManager is the standard Linux network configuration tool suite
- Nmstate is a library with an accompanying command line tool that manages host networking settings in a declarative manner.
Programming your networking configuration tools give you flexibility.

- We developed our own nmstate library
- `nmstatectl` is built using nmstate
Serde

JSON and YAML are your friends

Serde is a framework for serializing and deserializing Rust data structures efficiently and generically.

- Serde allow our users to define their declarative network state
- Allow us to implement our own Serializer and Deserializer
- There are plenty of serde decorators

Source: [https://serde.rs/](https://serde.rs/)
Serde

JSON and YAML are your friends

```rust
interfaces:
    - name: bond99
      type: bond
      state: up
      ipv4:
        address:
        - ip: 192.0.2.0
        prefix-length: 24
        enabled: true
      link-aggregation:
        mode: balance-rr
      options:
        miimon: '140'
      port:
        - eth3
        - eth2
```

```rust
#[serde(try_from = "NumberAsStdString")]

#[serde(
    skip_serializing_if = "Option::is_none",
    rename = "port",
    alias = "ports"
)]

#[serde(
    skip_serializing_if = "Option::is_none",
    default,
    deserialize_with = "crate::deserializer::option_u64_or_string"
)]

pub enum BondMode {
    #[serde(rename = "balance-rr")]
    /// Deserialize and serialize from/to "balance-rr".
    /// You can use `Integer8` for deserializing to this mode.
    RoundRobin,
```

Source: https://nmstate.io/examples.html#interfaces-bond
Zbus

Hello NetworkManager, how are you doing?

Zbus is a Rust API for D-Bus communication. Safe, high- and low-level.

- NetworkManager provides D-Bus API
- Zbus + zvariant will allow us to communicate with NetworkManager
- Configuring and retrieving NetworkManager profiles and devices
Nispor & Netlink

Providing an unified interface for Linux network state querying

Nispor allow us to query real-time Linux network information using rust-netlink crate.

- Real-time Linux network information is used to perform verification and partial editing.
- Contributed to rust-netlink to extend the supported interfaces
Sockets

Communicating with ovsdb

NetworkManager does not support global ovsdb configuration.

- Nmstate uses the Rust std library for Unix stream sockets.
- We use serde and serde_json libraries to create our own json_rpc library

Source:
https://github.com/nmstate/nmstate/blob/base/rust/src/lib/ovsdb/json_rpc.rs
Bindings boost adoption from other projects/organizations.

- We create C bindings from our Rust library
- Then Python and Golang bindings from the C one
- And now we can still using our integration tests written in Python with our Rust library!
Questions?

Feel free to ask questions! There are not dumb questions :-) 

Contact me at ffmancera@riseup.net or ffmancera@mastodon.social