RISC-V向けOS上でのICMPプロトコルスタック実装

Arch B2 Kota
親: macchan

背景

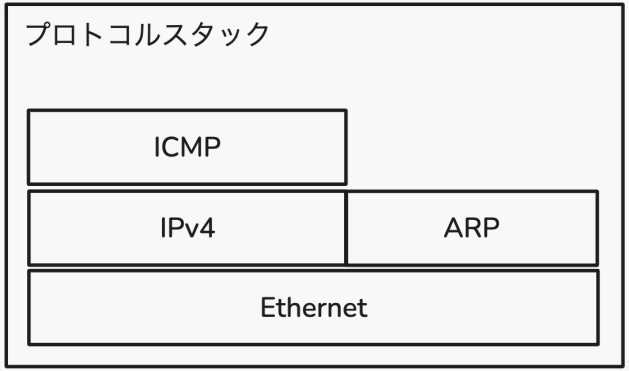
- 大学入学時からトランスポートプロトコルに興味がある
  - 特にQUIC
- IETFに参加した際に、カーネルの知識不足から満足に議論に参加できなかった
- またQUICに関する調査を進める中で低いレイヤーのコードを読む必要が生まれ、背景知識を得る必要が生まれた
- 今期はよりコンピュータアーキテクチャ、ネットワークアーキテクチャの理解を深めるためにICMP Echo Request/Reply送受信（ping）に対応するプロトコルスタックを実装した

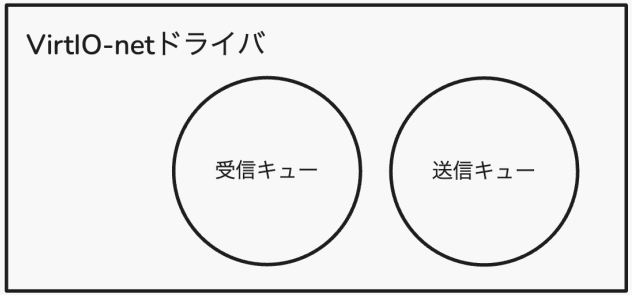
ベースにしたOS

- 「1000行で作るOS」[1]
  - RISC-V 32bits向け
  - OpenSBI[2]を用いてQEMU[3]上で動作する
  - より大規模のOS実装も検討したが時間の制約により小さな実装に止まった
- プロセス切り替え時のメモリ解放など、50行ほど追加機能を拡張した

プロトコルスタックの実装

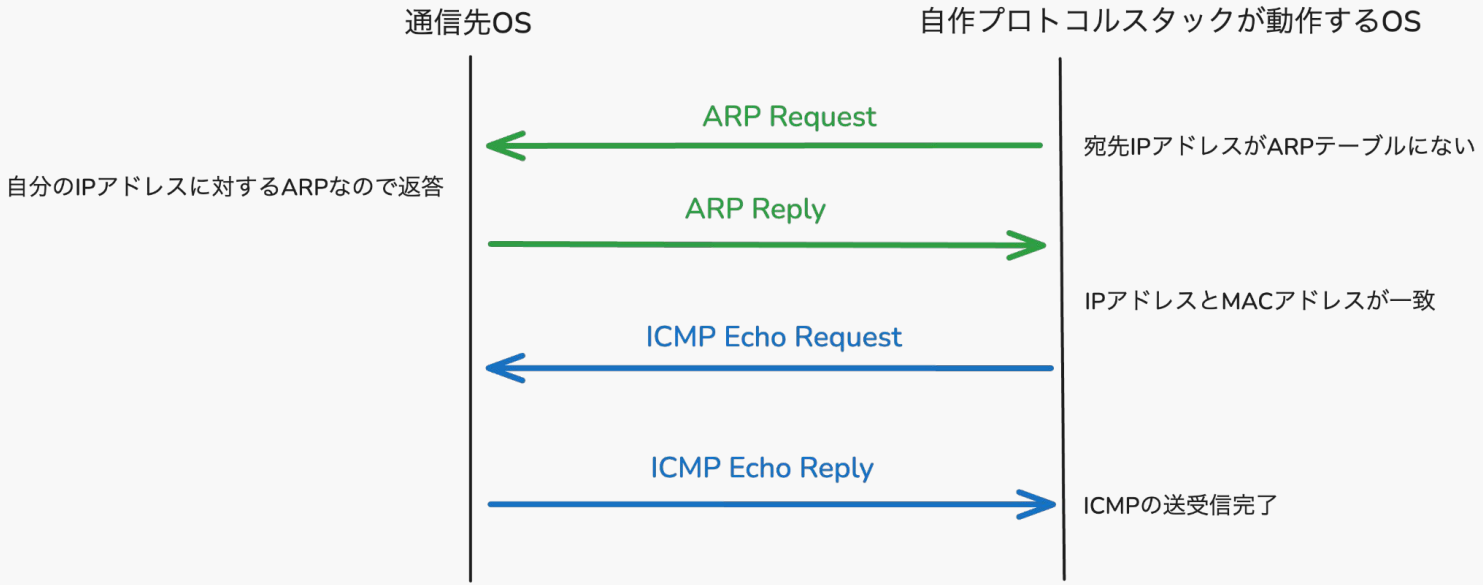
- OS上でICMP送受信の実装を行った
  - 準仮想デバイス virtio-net[4]を利用
  - Ethernet/IP/ICMPとARPに対応する簡易的なプロトコルスタックを実装
- 追加した実装が700行ほど
  - ARP Request/Reply
  - ICMP Echo Request/Reply





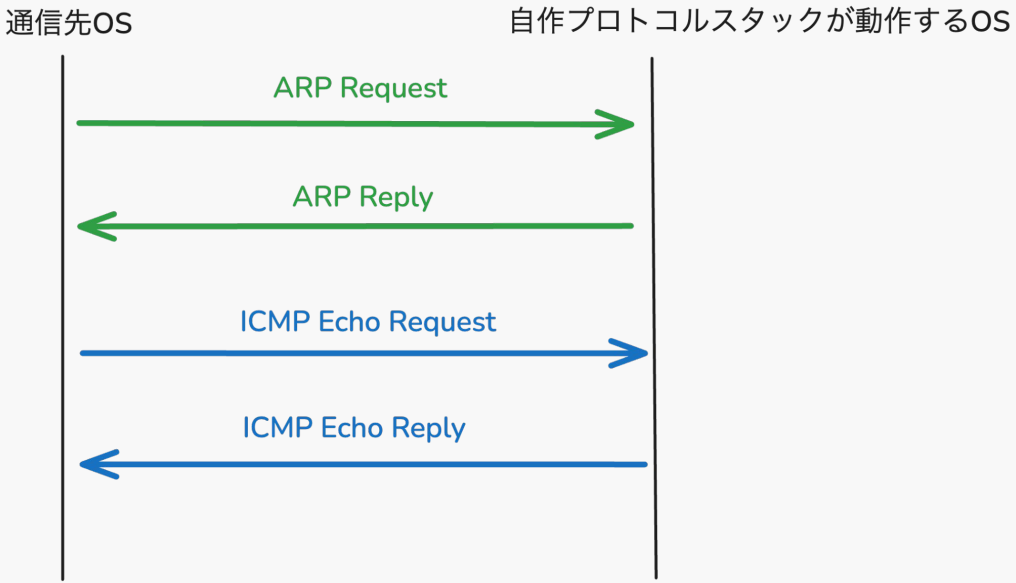
リポジトリ: https://github.com/kota-yata/small-nick-os

自作プロトコルスタックからICMPリクエストを送信する際のシーケンス



- 最初の通信時のみARP Request/Replyのやり取りが発生する
- 2回目以降はすでに対応表が存在するのでICMPの往復だけで通信は完了する

ICMPリクエストを受信する際のシーケンス



- 最初の通信時のみARP Request/Replyのやり取りが発生する
- 2回目以降はすでに対応表が存在するのでICMPの往復だけで通信は完了する

結果

自作プロトコルスタックからのリクエスト送信と応答のパケットキャプチャの結果

Source	Source Port	Destination	Destination Port	Protocol	Length	Info
52:54:00:12:34:56		Broadcast		ARP	42	Who has 192.168.100.10
4e:8b:3e:54:df:02		52:54:00:12:34:56		ARP	42	192.168.100.101 is at
192.168.100.104		192.168.100.101		ICMP	51	Echo (ping) request
192.168.100.101		192.168.100.104		ICMP	51	Echo (ping) reply

↑自作PSからEcho Requestを送信。ARPの送受信後Echo Replyが返ってきた

Source	Source Port	Destination	Destination Port	Protocol	Length	Info
82:2e:89:fe:32:b8		Broadcast		ARP	60	Who has 192.168.100.104? Tell 1
52:54:00:12:34:56		82:2e:89:fe:32:b8		ARP	42	192.168.100.104 is at 52:54:00:
192.168.100.102		192.168.100.104		ICMP	98	Echo (ping) request id=0x0037,
192.168.100.104		192.168.100.102		ICMP	98	Echo (ping) reply id=0x0037,

↑自作PS宛てのEcho RequestにEcho Replyで応答

本実装に関する今後の計画

- TCP/UDP/QUICのプロトコルスタックを設計/実装する
  - ネットワークソケットの実装
  - ステートフルな通信の処理を書く
- 物理NIC（e1000など）でネットワーク処理を書く
  - virtioに関する実装はほとんどが共有メモリの初期化作業だった
  - あまりNICを触っているという気持ちにならなかった
  - 本物のNICのための処理を書いてみたい

参考文献

[1] "Operating System in 1000 Lines." Operating System in 1000 Lines, n.d., https://operating-system-in-1000-lines.vercel.app/ja/.

[2] "RISC-V Open Source Supervisor Binary Interface (OpenSBI)." GitHub, n.d., https://github.com/riscv-software-src/opensbi.

[3] QEMU Project. “QEMU - The Fast Processor Emulator.” QEMU, n.d, https://www.qemu.org.

[4] Tsirkin, Michael S., and Cornelia Huck, editors. Virtual I/O Device (VIRTIO) Version 1.1. OASIS, 20 Dec. 2018, https://docs.oasis-open.org/virtio/virtio/v1.1/csprd01/virtio-v1.1-csprd01.html.