BBSakura Networks

# XDPerf: High-Performance Network Traffic Generator

2026/07/16

Takeru Hayasaka <t-hayasaka@bbsakura.net>

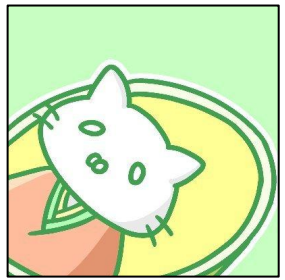
JANOG58 Day2 LT

# Whoami

- 早坂 彪流 (Hayasaka Takeru | @takemioIO)
- Senior Software Enginner
- さくらインターネット に所属

  現在は、BBSakura Networksへ出向中

  モバイルコアや仮想化基盤の運用開発を担当
  - 前職はゲーム会社でゲーム機のファームを書いていた

# The Current Landscape of Network Testing Tools

BBSakura Networks

標準的なパケットジェネレーター（負荷試験ツール）では、

使いやすさと機能性の間でトレードオフを強いられることがよくあります

iperf / iperf3

🙆 TCP/UDPスループットを測定可能、簡単にインストール出来てお手軽！

🙅 L2/L3パケットの処理機能はなし

Cisco TRex (DPDK based)

🙆 Raw L2フレームを起点として、多様なトラフィックを生成可能で高性能！

🙅 参入障壁：複雑なDPDKのセットアップ、OS設定の大幅な変更、

特定のNICハードウェア、およびPythonへの依存関係が必要

# The eBPF Approach & Challenge

BBSakura Networks

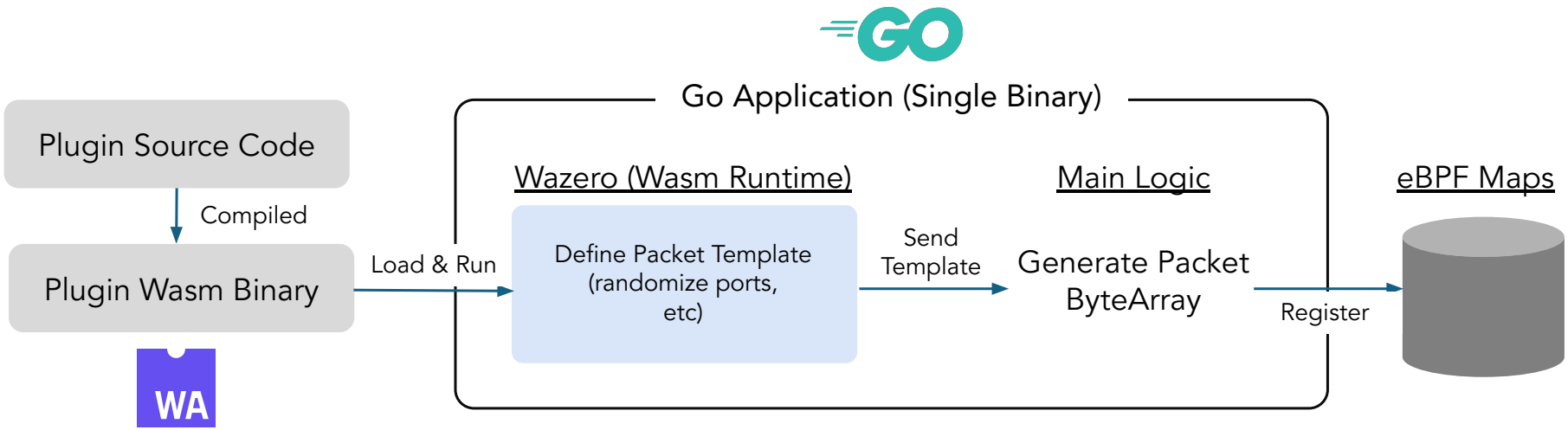
- DPDKほどは性能は要らないので、お手軽でそこそこの性能が欲しい！
  →eBPF / XDPが使えるのでは・・・！？
- eBPF/XDP (eXpress Data Path) Live Frames Mode(BPF_PROG_TEST_RUN)
  - 🙆カーネル空間内で完結する高速パケット処理を可能にするカーネル機能
    DPDKと違って、NICバインドしない点など含め既存の構成と共存しやすい
  - 🤷Live Framesモードでは、単一の静的なパケットを送り続けるだけ
- The Challenge🧐
  - 単一パケットしか送れない仕様では、実際のテスト環境に対応できずに困る！
    multiflow等で様々な変化をつけて多様なトラフィックにしたい!
    (e.g. 送信元IPアドレス、ポート番号、プロトコル、パケット長の違い、Proto種別など)
  - **高い処理性能と柔軟なパケット生成を両立するツールは構築できないか？**

# Our Solution: XDPerf https://github.com/takehaya/xdperf

- WebAssembly (Wasm) plugins
  - **テストシナリオやパケット定義を記述します。**
  - Wasmにコンパイル可能な任意の言語（Rust、TinyGoなど）を開発に使用できます。
- Go Application
  - 単一のGoアプリケーションがWasmランタイム（wazero）を組み込み、eBPFとのやり取りを管理します
  - Wasmから渡されたテンプレートに基づいてパケットのバイト列を生成し、eBPFマップに登録します
  - 実行時にはXDPのプログラムがeBPFマップの中身を見て、動的に投げるべきパケットを処理します
- **実際のパケット送信は、カーネル内のXDPによって処理しつつ、投げたいパケットはWASMで簡単に書くことができます！**

# Our Solution: XDPerf

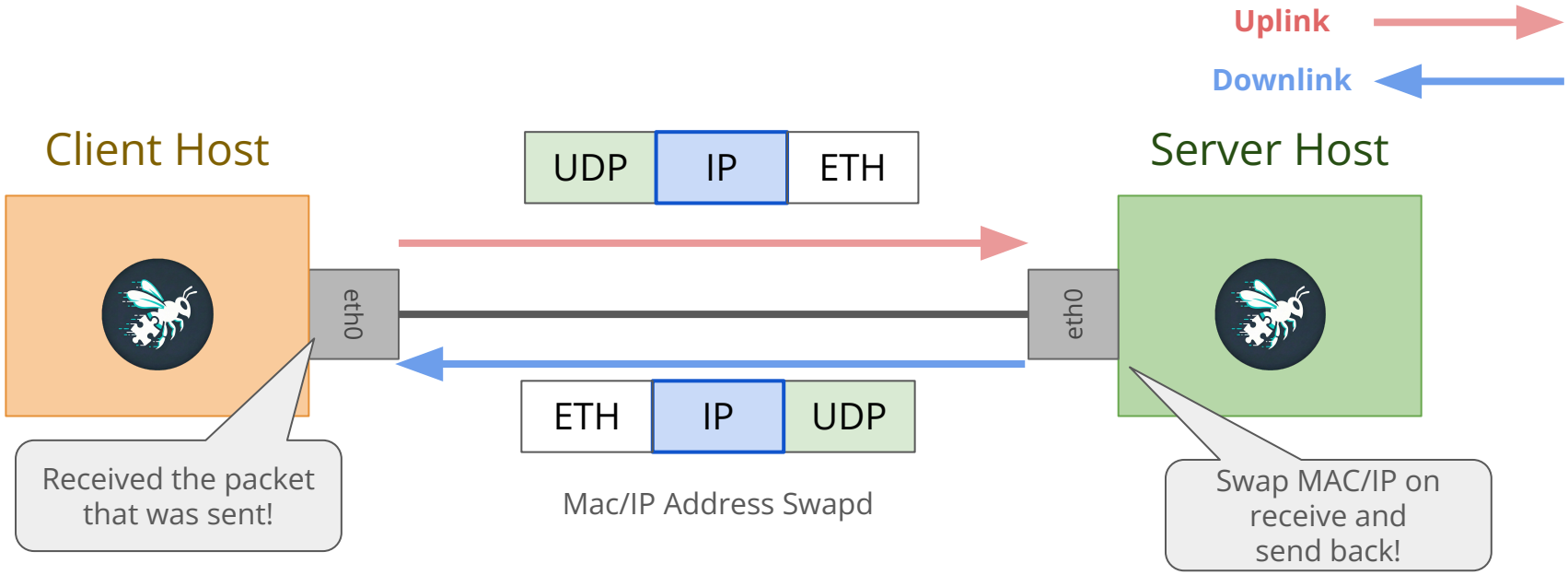
BBSakura Networks



# How To Use

1. One linerでInstall可能
    a. curl -fsSL https://raw.githubusercontent.com/takehaya/xdperf/main/scripts/install_xdperf.sh | sudo bash
2. コマンド一発で負荷をかけれる
    a. sudo xdperf run --device eth0 --plugin simpleudp.go --count 128 --parallelism 24 --infinite --batch-size 64 \ --cfg '{"src_ip":"192.168.1.1","dst_ip":"192.168.1.2","dst_port":10001,"payload_size":22,"is_arp_resolve":false}'
3. さまざまなPluginを使えば IMIXが出来たりGTP-UやSRv6も使えたり...etc

# Example: Round-Trip Test with MAC/IP Address Swapping




```
sudo .xdperf run --device eth0 \
  --plugin simpleudp.go \  --count 128 --parallelism 24 --infinite
--batch-size 64 \ --cfg
'{"src_ip":"192.168.1.1","dst_ip":"192.168.1.2","dst_port":10001,"p
ayload_size":22,"is_arp_resolve":false}'
```

```
sudo xdperf --device eth0
--send=false --recv=true
--swap-resp
```

# 測定環境情報

- kernel: linux kernel 6.15(Client), 6.8(Server)
- CPU: Intel Xeon Platinum 8362 @ 2.80GHz
  - コア構成: 32コア / 64スレッド / 1ソケット / 1 NUMA
- MEM: 64GB
- NIC: Intel E810-C 100GbE (ice, 21.5.9)
- 100GbEでケーブル直結で繋がっている

# Payload Size変化での性能評価

32Core利用で実施

片方向だけ64B, 128Bの時は性能が良い。

恐らく割り込みの量でRoundTrip pkt

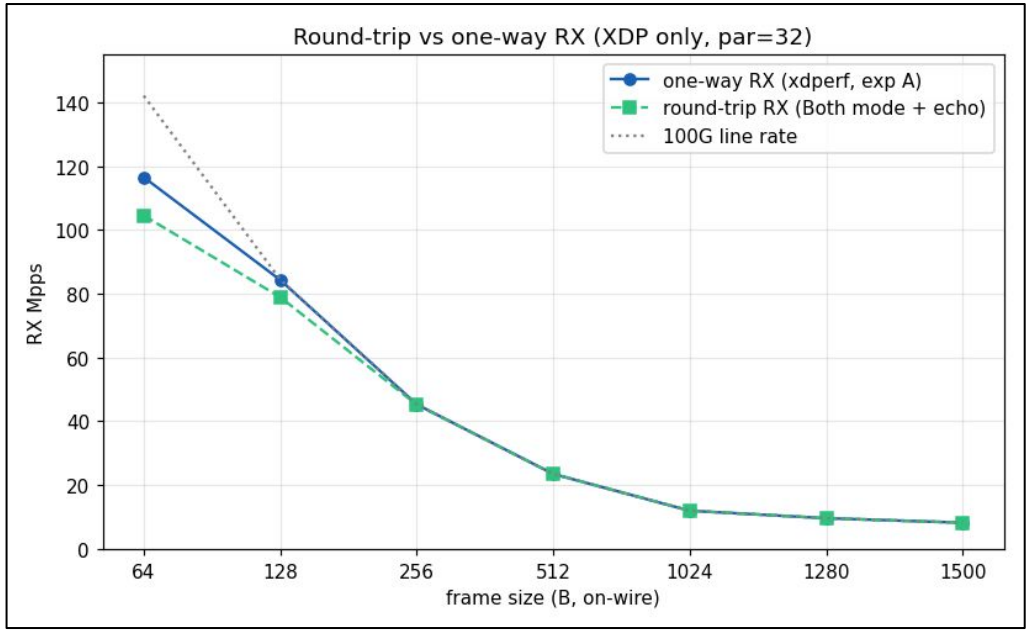
に対する受信性能が強く劣化してるため

**片方向: 116.5Mpps**

**往復: 105.5Mpps**

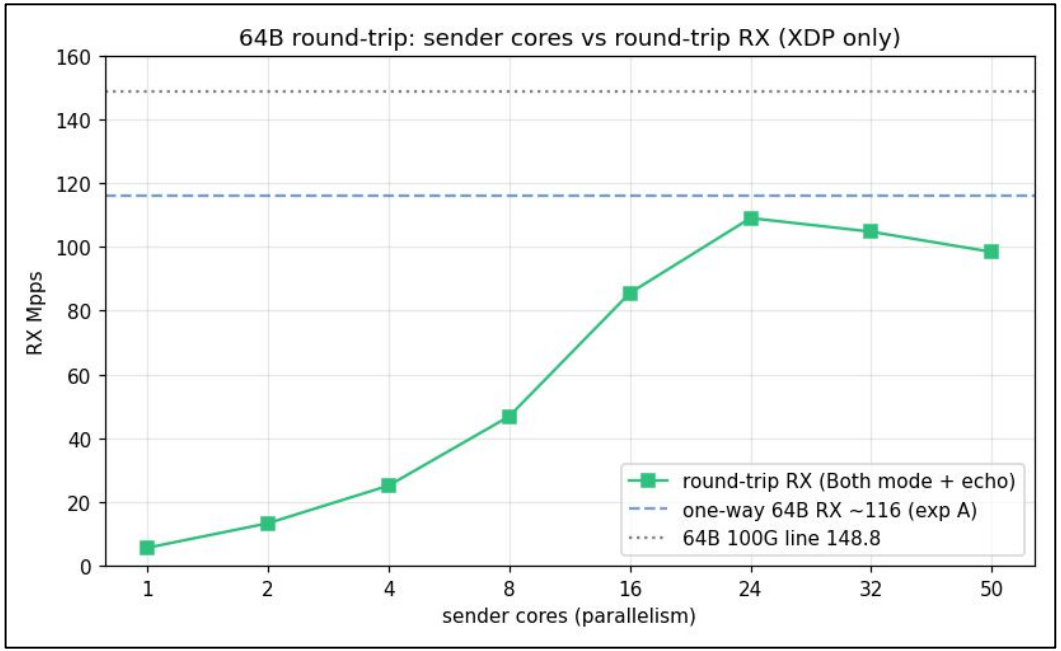
と10Mpps分劣化したと思われる

256B以降は片方向も往復も同様の性能



# Round-Trip時のCoreあたりの性能評価

- shortpkt 64byteで負荷
- 1 Core: 5.56 Mpps
- 2 Core: 13.26Mpps
- …24Core: 109.77 Mpps
- 大体24Coreぐらいで
  XDPerfだと性能が頭打ちに
- 谷になってるのは恐らくRX/TXで
  CPUのリソースをお互いに
  食い合ってるのが理由と思われる
- 148.8Mppsが100Gbps（L2)の理論
  値なのでRTTでは**73%程度**使えてる



# まとめ

- XDPerfというDPDKフリーのな自作負荷試験ツールを紹介しました
  - WASMを使うことでGoやRust等で好きな言語でパケットの構造を書くことで
    任意のパケットを送信できます
  - ショートパケットで片方向RX Mppsで**116.4 Mpps**ほど出ます
- XDPerfはワンライナーで導入できます！
  - pluginの書き方はぜひリポジトリを見てください！ぜひ興味があれば使ってください！
  - **curl -fsSL https://raw.githubusercontent.com/takehaya/xdperf/main/scripts/install_xdperf.sh | sudo bash**
- 宣伝: この発表のXDPerfのDeepDiveを詳しく聞きたくなった人は
  Kubecon Japanコロケイベント(7/28)で、30分ほど話す予定です。そちらもどうぞ:)
  - https://events.linuxfoundation.org/kubecon-cloudnativecon-japan/co-located-events/japan-community-day-schedule/?id=1261494


https://github.com/takehaya/xdperf

# Enabling a Connected Future.

BBSakuraNetworks