# Simple is better

Building fast IPv6 transition mechanisms on Snabb Switch

31 January 2016 – FOSDEM 2016

Katerina Barone-Adesi `kbarone@igalia.com`

Andy Wingo `wingo@igalia.com`

# Snabb Switch

A toolkit for building network functions

High performance, flexible, hackable data plane

# The Tao of Snabb

Simple > Complex

Small > Large

Commodity > Proprietary

# Simple > Complex

How do we compose network functions from smaller parts?

Build inside of network function like composing UNIX pipelines

```
intel10g | reassemble | lwaftr | fragment | intel10g
```

Apps independently developed, linked together at run-time

# Simple > Complex

What is a packet?

```
struct packet {
    unsigned char data[10*1024];
    uint16_t length;
};
```

# Small > Large

Early code budget: 10000 lines

Build in a minute

Constraints driving creativity

# Small > Large

Secret weapon: LuaJIT

High performance with minimal fuss

# Small > Large

Minimize dependencies

1 minute build budget includes LuaJIT and all deps

Deliverable is single binary

# Small > Large

Writing our own drivers, in Lua

User-space networking

- The data plane is our domain, not the kernel’s
- Not DPDK’s either!
- Fits in 10000-line budget

# Commodity > Proprietary

Open source (Apache 2.0)

# Commodity > Proprietary

Open data sheets

Intel 82599 10Gb, soon up to 100Gb

Soon: Mellanox (they agree to release data sheet!)

Also Linux tap interfaces, virtio host and guest

# Commodity > Proprietary

Double down on 64-bit x86 servers

Prefer CPU over NIC where possible

Embrace the memory hierarchy

# Storytime!

“We need to do work on data... but there’s just so much of it and it’s really far away.”

# Storytime!

Modern x86: who’s winning?

Clock speed same since years ago

Main memory just as far away

# HPC people are winning

"We need to do work on data... but there's just so much of it and it's really far away."

Three primary improvements:

- ❧ CPU can work on more data per cycle, once data in registers
- ❧ CPU can load more data per cycle, once it's in cache
- ❧ CPU can make more parallel fetches to L3 and RAM at once

# Networking folks can win too

Instead of chasing zero-copy, tying yourself to ever-more-proprietary features of your NIC, just take the hit once: **DDIO into L3**.

Copy if you need to – copies with L3 not expensive.

Software will eat the world!

# Networking folks can win too

Once in L3, you have:

- wide loads and stores via AVX2 and soon AVX-512 (64 bytes!)
- pretty good instruction-level parallelism: up to 16 concurrent L2 misses per core on haswell
- wide SIMD: checksum in software!
- software, not firmware

# </storytime>

So what about the lwAFTR

# IPv6 transition on Snabb: a lwAFTR

# Why IPv6?

- The IPv4 address space is exhausted

  - IANA top level exhaustion in 2011

  - 4/5 Regional Internet Registries exhausted

  - September 2012 in Europe

  - September 2015 in the US

  - AfriNIC within the next few years

- The internet is still growing
- Moving to IPv6 helps

# IPv6 transition mechanisms

- Users want everything to continue working

  … including IPv4 websites, networked games, etc
- Some user equipment cannot do IPv6
- Several options: NAT64, 464XLAT, DS-Lite...

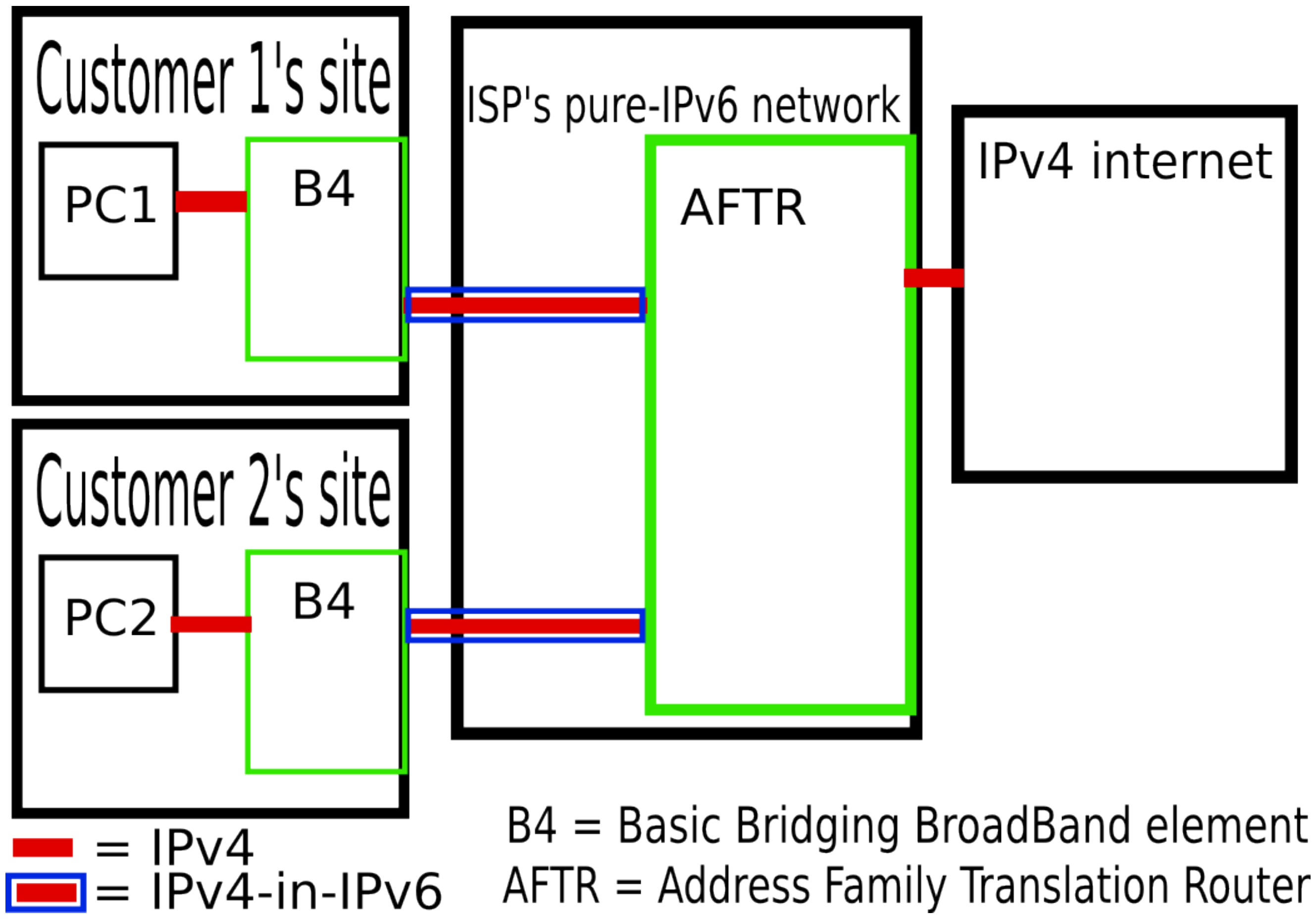
# Why Lightweight 4over6?

- Similar to DS-Lite, but less centralized state
- Share IPv4 addresses between users
- Each user gets a port range
- Allows providers to have a simpler architecture: pure IPv6, not dual-stacked IPv4 and IPv6, in their internal network
- Standardized as RFC 7596 in 2015

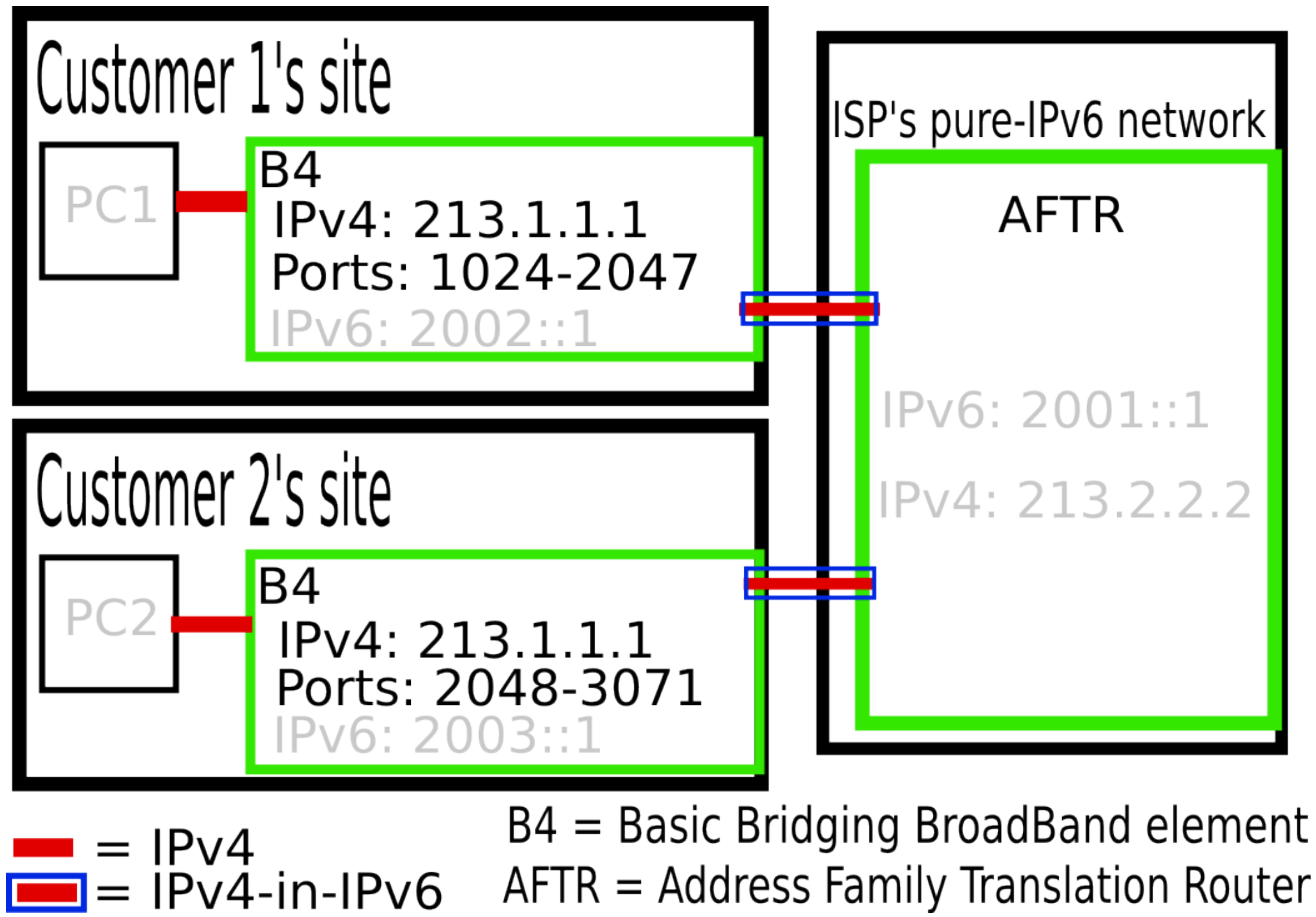
# Two main parts: B4 and AFTR

- Both encapsulate and decapsulate IPv4-in-IPv6
- Each user (subscriber) has a B4
- The network provider has one or more AFTRs, which store per-subscriber (not per-flow) information
- The information: The B4's IPv6 address, IPv4 address, and port range.

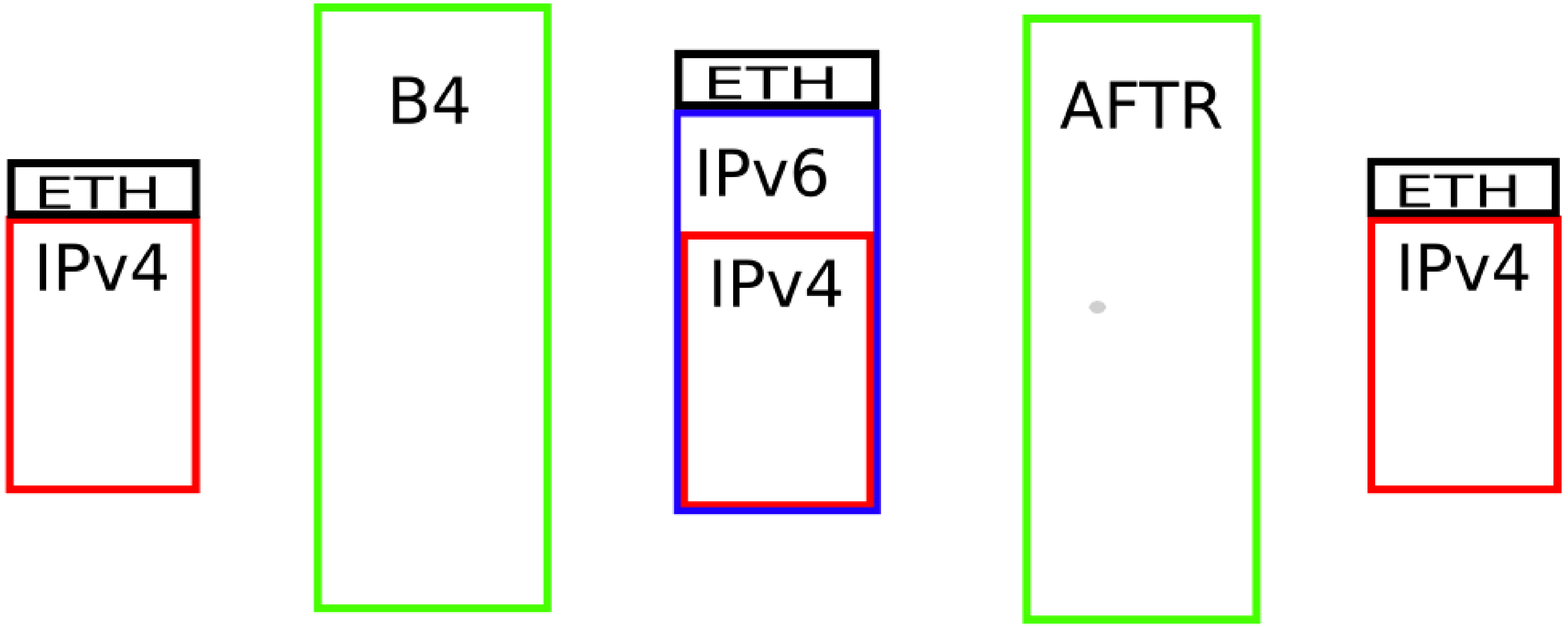
# lw4o6 architecture



# Lw4o6 address sharing



# IPv4 is tunnelled in IPv6




The B4 and AFTR encapsulate and decapsulate

All packets between them are IPv6

# Snabb lwAFTR

- Started July 2015
- Proof of concept data plane October 2015
- It's already usable and fast.
- http://github.com/igalia/snabbswitch/

  - lwaftr* branches

  - Apache License v2

# Performance

- Hardware: two 10-gigabit NICs

  - Intel 82599ES, SFI/SFP+
- Xeon processor: E5-2620 v3 @ 2.40GHz
- Snabb-lwaftr alpha release
- 550-byte packets
- Over 4 million packets/second

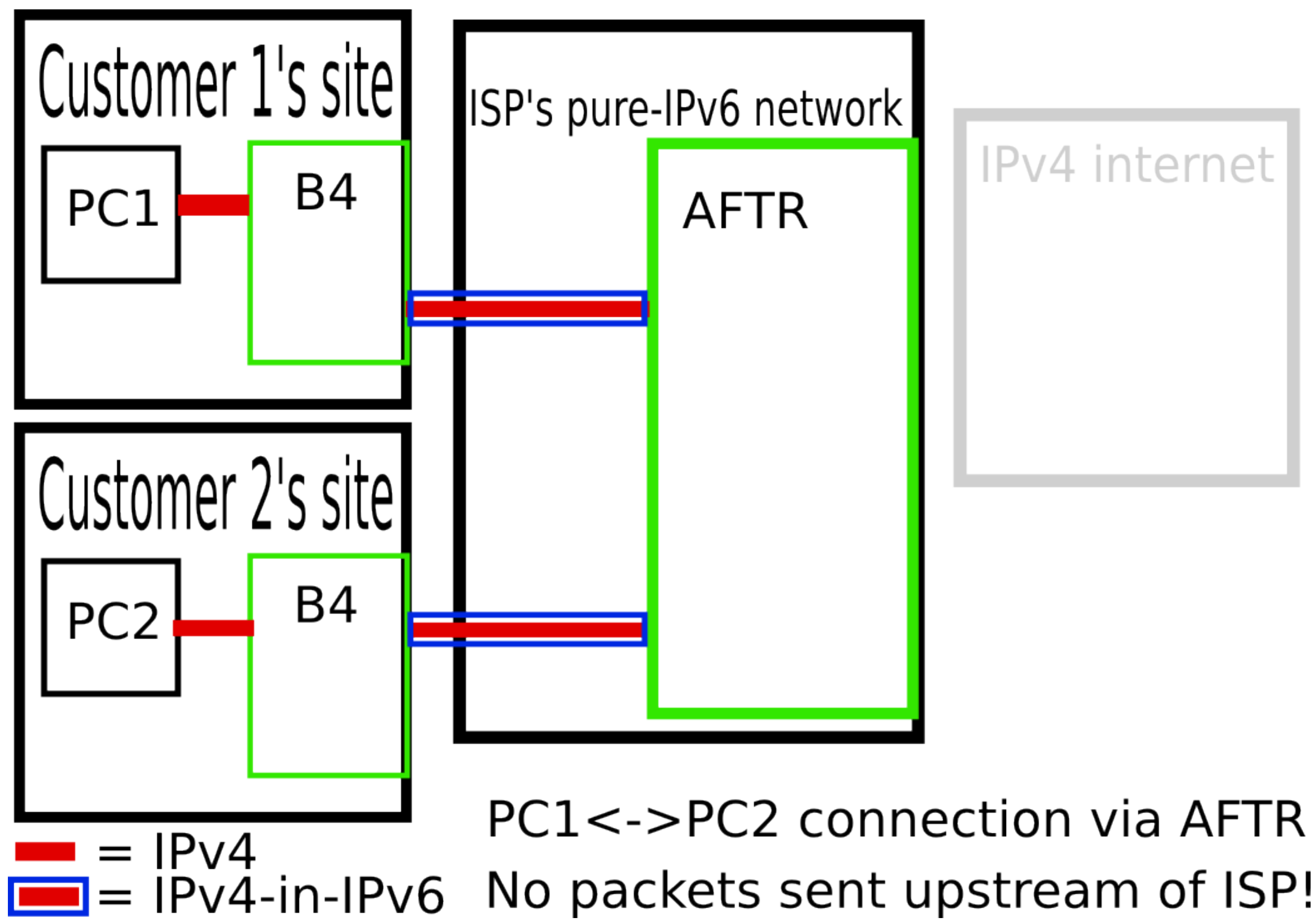
  → over **17 gigabit/second** handled on one core

# Challenges

- Correctly handling ICMP

  - conveying failure information, for instance to an IPv4 host if a failure occurs within the tunnel
- Speed
- Speed with a lot of subscribers
- Correctness
- Hairpinning

# Hairpinning: client-to-client traffic



# And we’re back

# Implementation challenges

Binding table lookup - Port partition

When to hairpin?

Virtualization

Policy

Configuration

# Binding table lookup

Say, Belgium: millions of tunnels

Per-tunnel: IPv4, IPv6 of B4, port set ID

At least 4 + 16 + 2 = 22 bytes

2M entries: 44MB

You can’t fit it in L3.

# Binding table lookup

So always budget for an L3 cache miss – but only one!

4 MPPS in: 250 ns/packet

One cache miss RTT (80 ns) within budget

Many fetches can happen in that RTT

# Binding table lookup: v1

Open-addressed robin-hood hash table with linear probing

Result probably right where we first look for it, otherwise in adjacent memory, might fetch adjacent cache lines

# Binding table lookup: v2

Maximum probe length around 8 for 2e6 entries, 40% occupancy

Stream in all 8 entries at once in parallel

Branchless binary search over those 8 entries

# Binding table lookup: v3

Stream in all 8 entries at once in parallel

❧ for multiple packets in parallel

32 packets at a time: amortized 50ns/ lookup

Worst-case bounds!

# Port partitioning

Different IPv4 addresses can have their ports partitioned in different ways

Need `f(ipv4, port) -> params`

Current solution: partition IPv4 space into ranges with same parameters, use binary search

# Hairpinning

Problem: after decapsulating IPv4 packet, send to internet or re-tunnel back to IPv6?

Answer: Use port partition as quick check, if so do the hairpinning

Yay software

# Virtualization

Want to make a virtualized lwaftr

Missing virtio-net implementation

Work by Virtual Open Systems; thanks!

Usual workload: One Snabb-NFV per interface on the host

Same performance

Yay software!

# Policy

Ingress/egress filtering

Pflua! `https://github.com/Igalia/pflua`

As an app!

# Configuration

Compile binding table from text

Update/control plane TBD

# Future work

Yang

Smaller packets

Integrate ILP binding table fetch

40Gb

# Thanks!

`kbarone@igalia.com`

`wingo@igalia.com`

`https://github.com/SnabbCo/snabbswitch`

`https://github.com/Igalia/snabbswitch`

ps. We are hiring!