PHY Covert Channels: Can you see the Idles?

Chupja - Network covert timing channel

Mateusz Walkiewicz

PHY Covert Channels: Can you see the Idles?
Ki Suh Lee, Han Wang, and Hakim Weatherspoon,
Cornell University, 2014

Communication channels

- Overt - visible
- Covert - transfer hidden information in between legitimate packets
  - storage - e.g. unused protocol headers
  - timing - delay between messages

Chupja

kor. 첩자 - mole, spy

- High-bandwidth
- Robust - < 10% Error rate
- Undetectable - without specialized hardware

First covert channel using PHYsical layer of Ethernet protocol

Uses fine-grained (<1ms) delay to hide communication from statistical analysis done in user- and kernel-space

10Gb Ethernet - IEEE802.3

Frame - 64 bit payload + 2 bit synchronization header

66/64 × 10G = 10.3125 Gbaud - ticks per second

Each bit is ~97 picoseconds wide (9.7 × 10^-11 s)

Gap between packets:

- /I/ - Idle symbol
- 700~800 picoseconds
- not visible by Layer 2 - discarded by hardware

The Idea

Use modified hardware to change length of interpacket gaps

Make those changes too small to be visible by software analysis tools

- System clock has at least 1 µs accuracy - can’t differentiate malicious packets from usual traffic

Use every gap to transfer one bit of covert data

Problems?

- Won’t hardware change the length of the gaps?
- Won’t other traffic on the network thwart our plans?
- What about high traffic?
- Can’t administrators use hardware timestamps to analyse the delays?

Details - length of the packet

Some definitions

- Homogeneous packet stream (HOM) - same size of packets, destination and IPD
- Maximum transfer unit (MTU) - 1500B + 14 Eth Header + 4 Correction = 1518B
- Interpacket delay (IPD) - difference between first bit of successive packets
- Interpacket gap (IPG) - time when /I/ are transferred between packets

Details

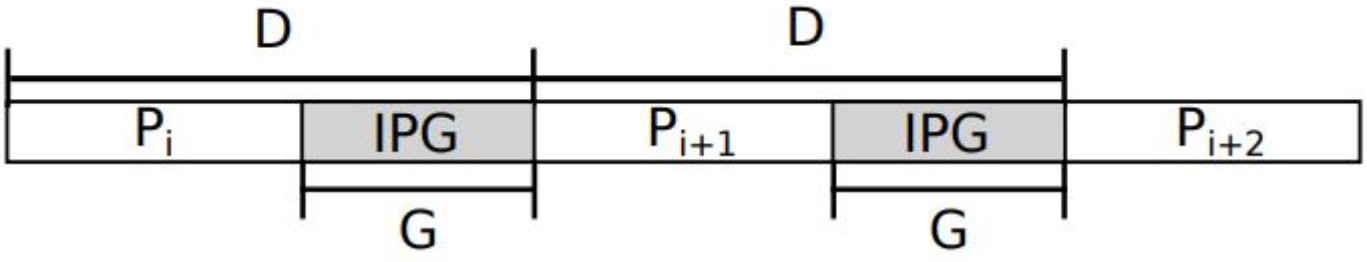
G - number of /I/s

Gi -- gap after i-th packet

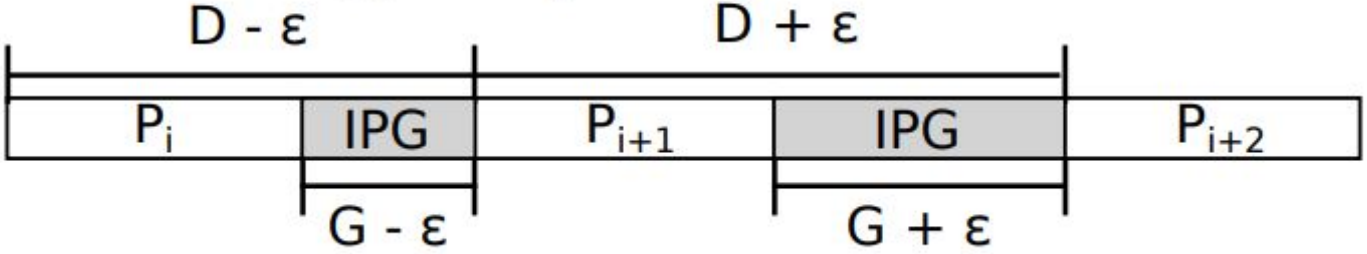
Bi -- i-th bit to transfer

Gi = G−ε if Bi = 0

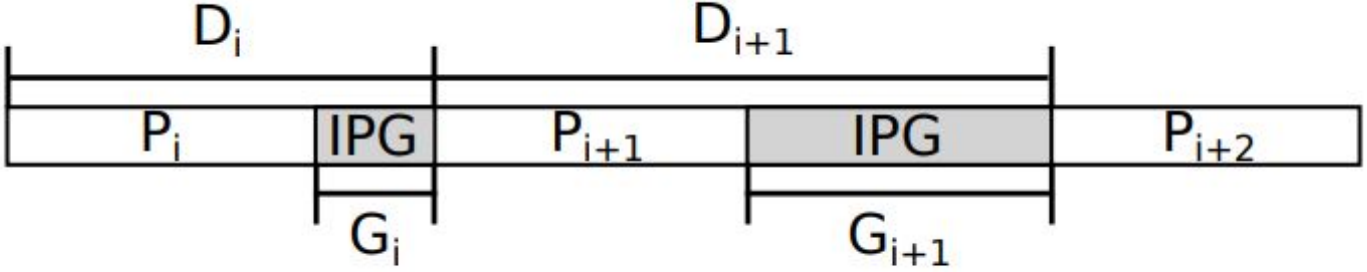
Gi = G+ε if Bi = 1



(a) Homogeneous packet stream



(b) Timing channel: Sender



(c) Timing channel: Receiver

Details - representation of bits

G - number of /I/s in HOM (Gap)

Gi -- gap after i-th packet; Bi -- i-th bit to transfer

$G_i = G - \varepsilon$ if $B_i = 0$

$G_i = G + \varepsilon$ if $B_i = 1$

We want ε to be long enough to be visible by the receiver, but still as short as possible, to prevent the channel from being discovered by the warden

W - Wait time - number of /I/s that indicate a pause in transmission

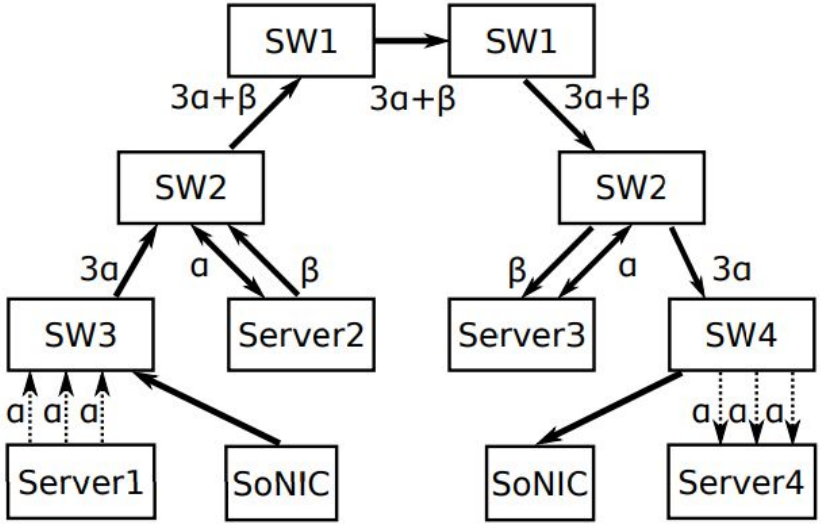
Implementation

Network interface cards were implemented using SoNIC (sonic.cs.cornell.edu), which allows to implement NIC on FPGA boards

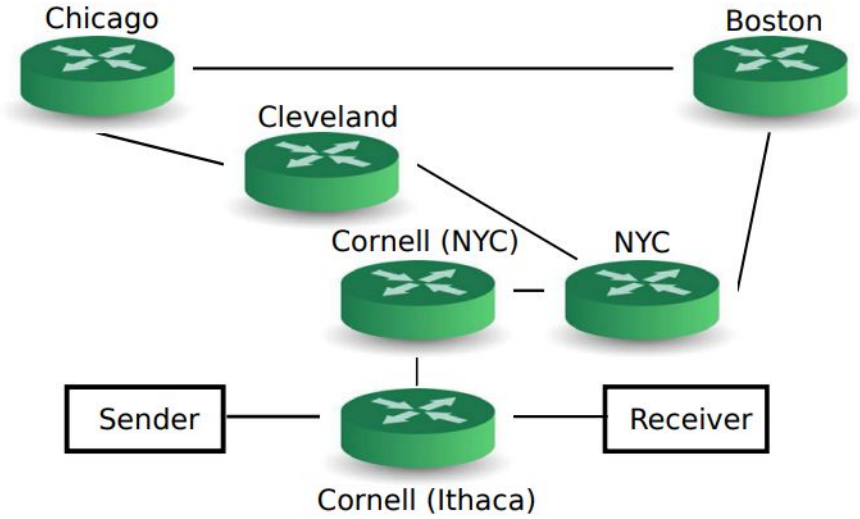
Precise control over hardware - send exact number of /I/s and capture packets with sub-nanosecond timestamping

50 lines of code added to existing codebase

Setups



Small Network
Few Hops, Local area
No or artificial cross-traffic



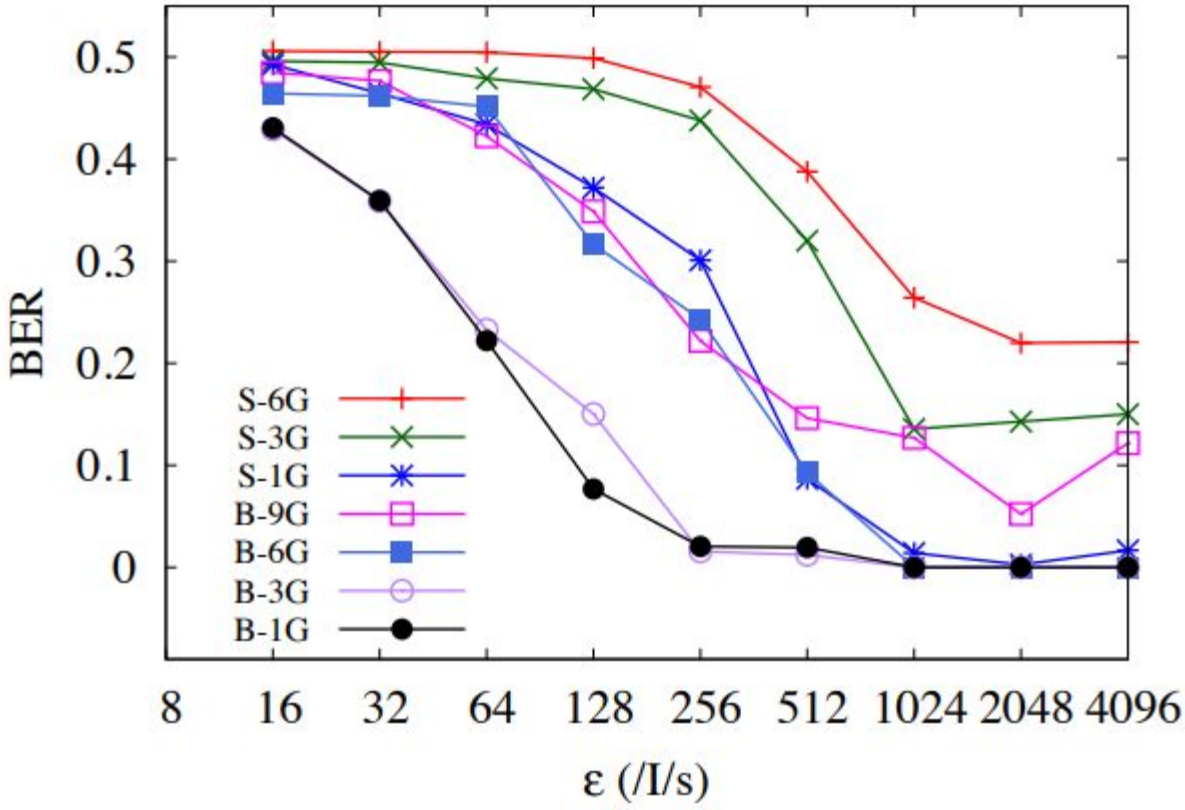
National Lambda Rail
9 routing hops, 4000 km,
High cross-traffic

Results - Error

BER - bit error rate

S - 64B packet

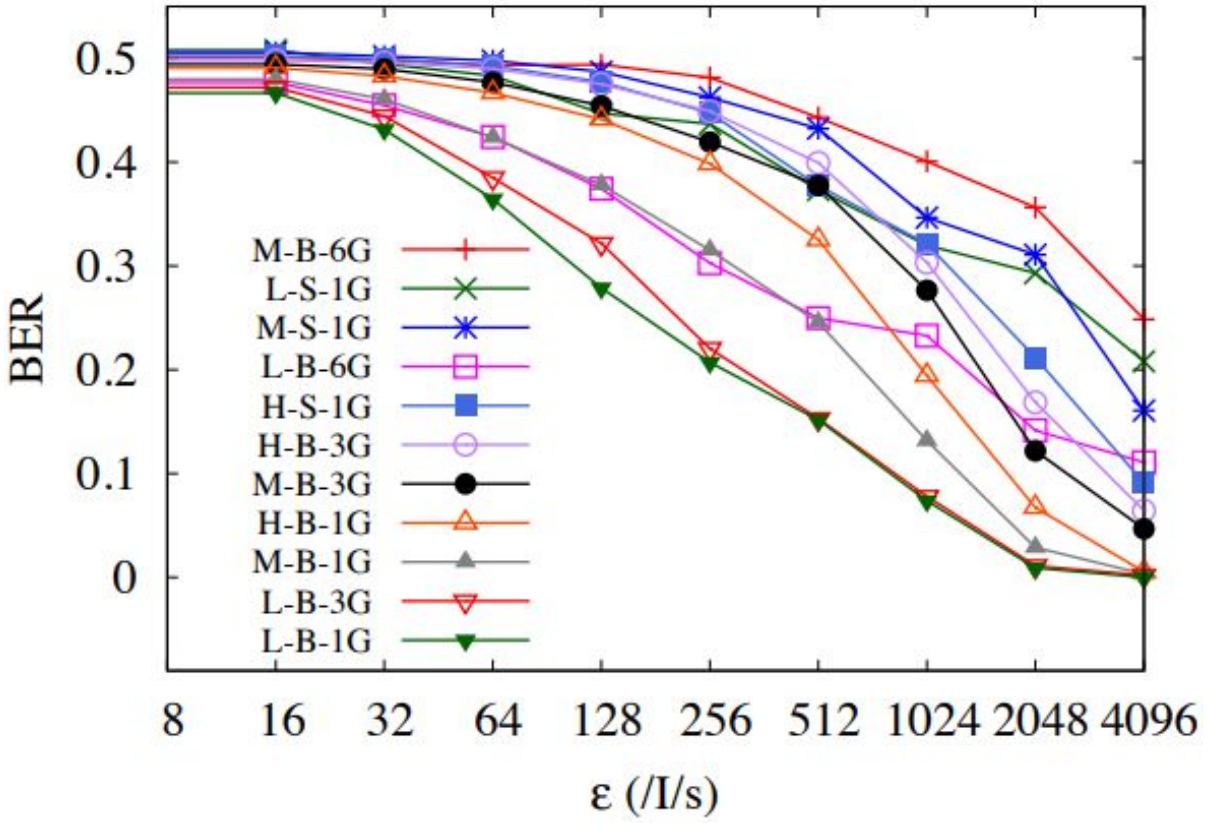
B - 1518B packet



(a) Without cross traffic over a small network

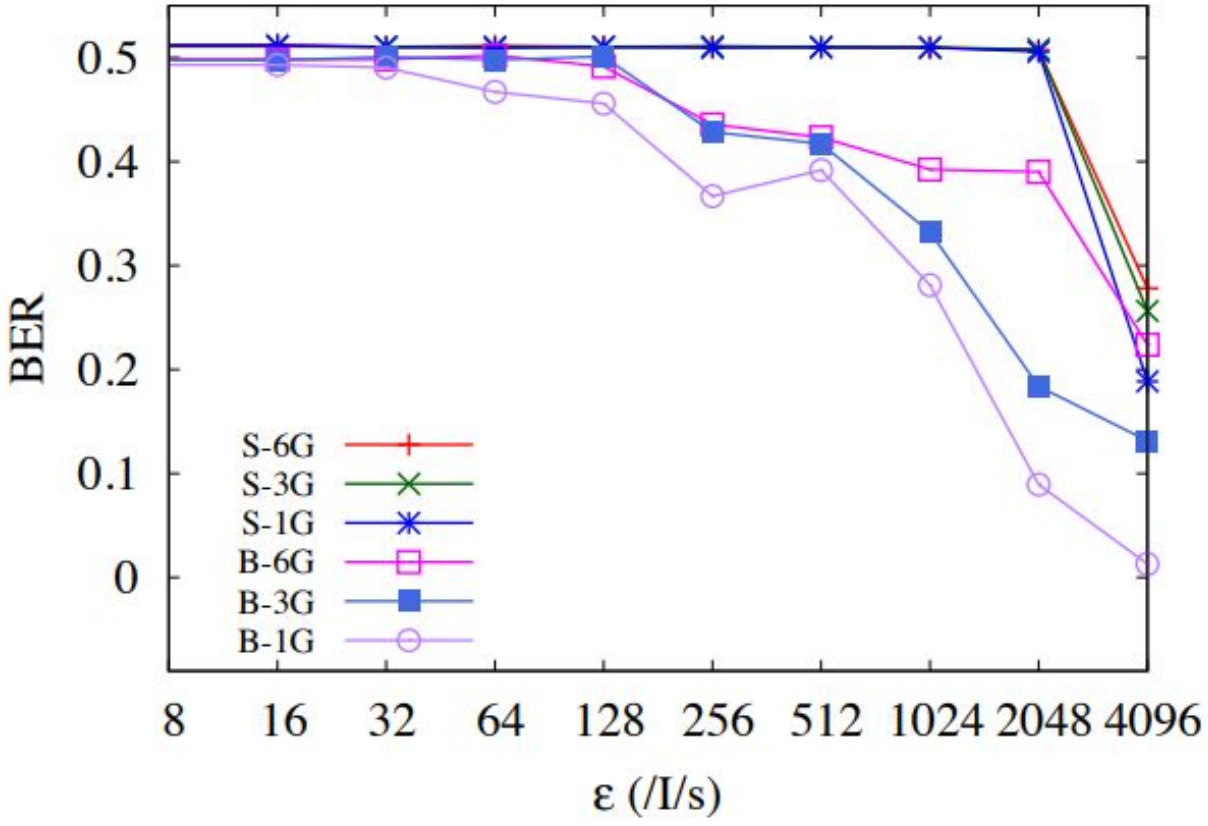
Results - Error

L, M, H - cross-traffic size



(b) With cross traffic over a small network

Results - Error



(c) Over National Lambda Rail

Results - how to pick right ε

$$P(\mu - \varepsilon \leq D \leq \mu + \varepsilon) \geq 0.90$$

D - random variable, delay time, known distribution
μ - mean delay

Results - do switches preserve the gaps?

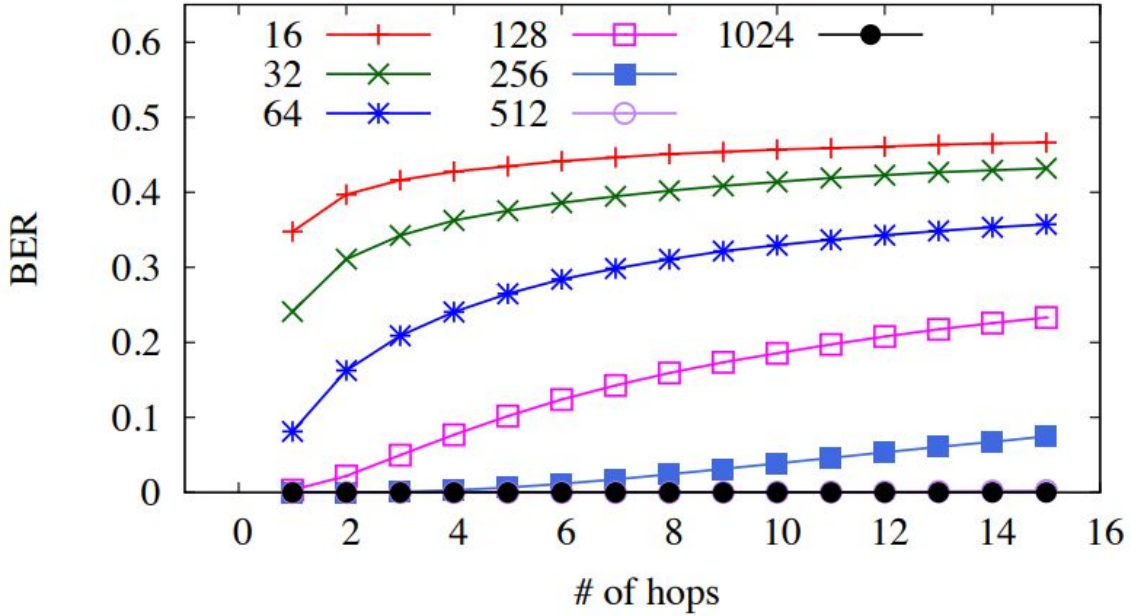
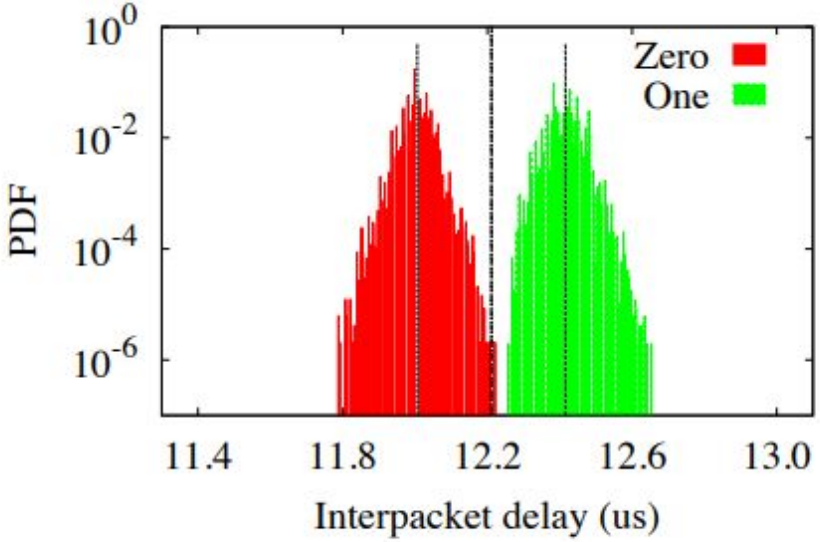
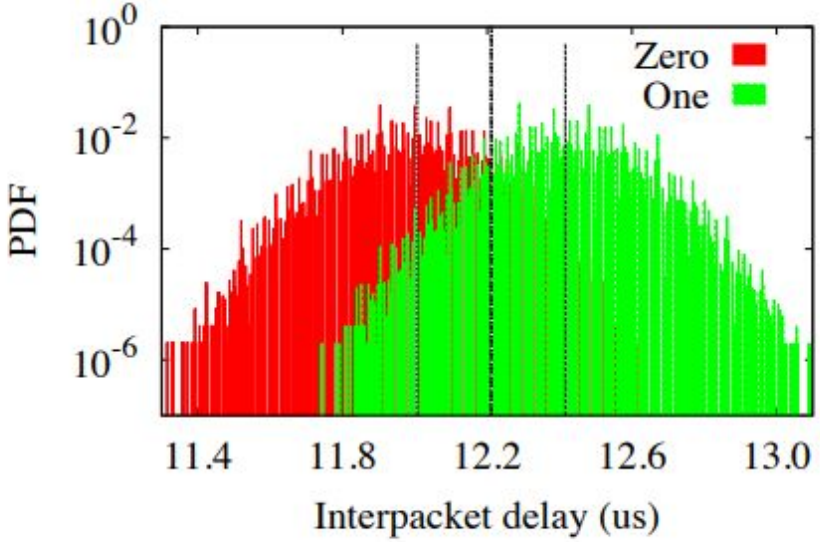


Figure 10: BER over multiple hops of SW1 with various ε values with (1518B, 1Gbps)

Results - do switches preserve the gaps?

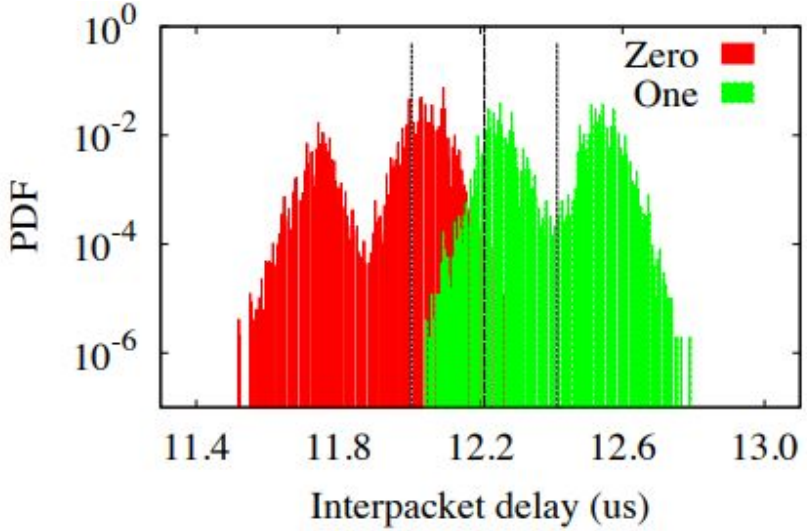


(e) $\varepsilon = 256$ after one SW1

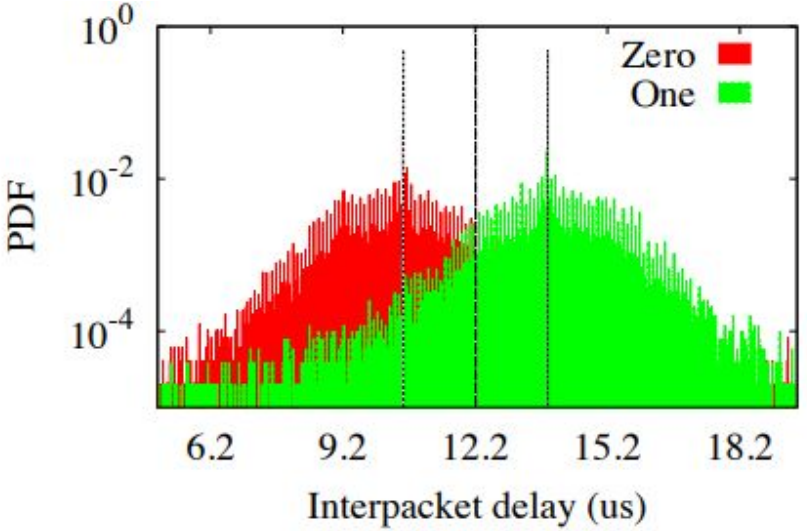


(f) $\varepsilon = 256$ after fifteen SW1

Results - do switches preserve the gaps?



(g) $\varepsilon = 256$ after one SW1 with cross traffic (64B, 1Gbps)



(h) $\varepsilon = 2048$ over NLR

Results - detection

Statistical tests on IPDs

Chupja traffic is very regular - easy to discover if monitored with high resolution

Tested with kernel, zero-copy and hardware timestamping.

Zero-copy - packets are copied to memory shared with user-space, no kernel stack overhead

Hardware - Sniffer 10G NIC - 500ns resolution

Results - detection

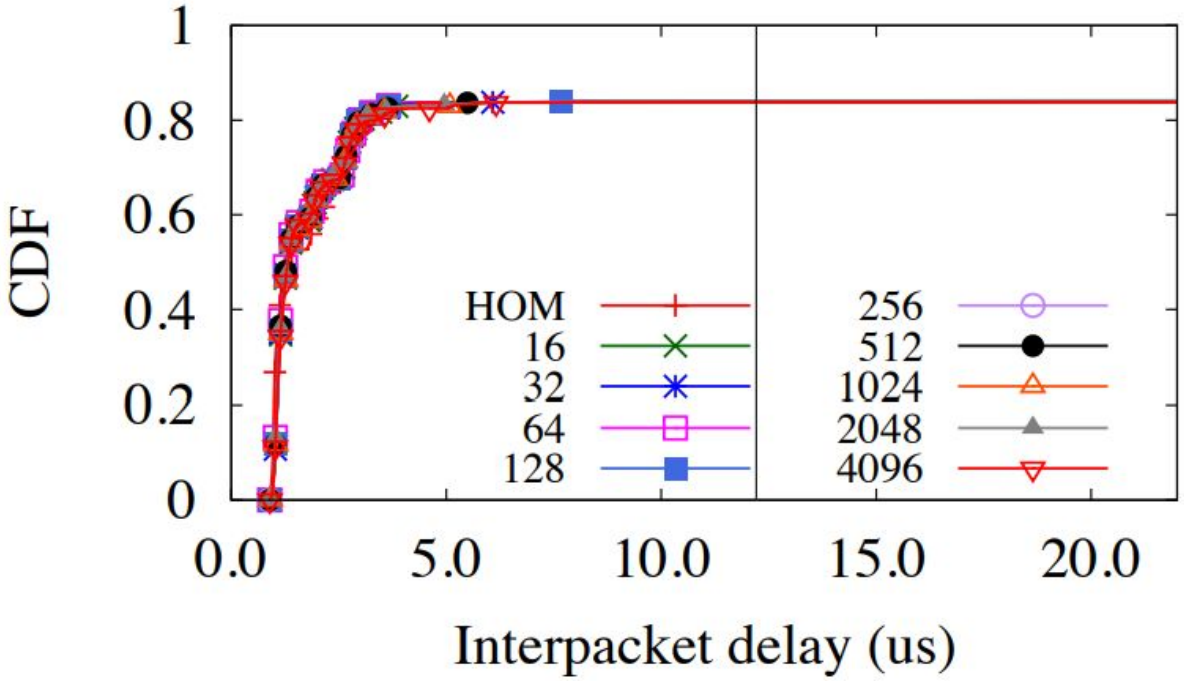
Tests on 1Gbps network with 1518B packets

$\mu = 12.2\mu s$

ε $(/\mathtt{I}/s)$	16	32	64	128	256	512	1024	2048	4096
ns	12.8	25.6	51.2	102.4	204.8	409.6	819.2	1638.4	3276.8

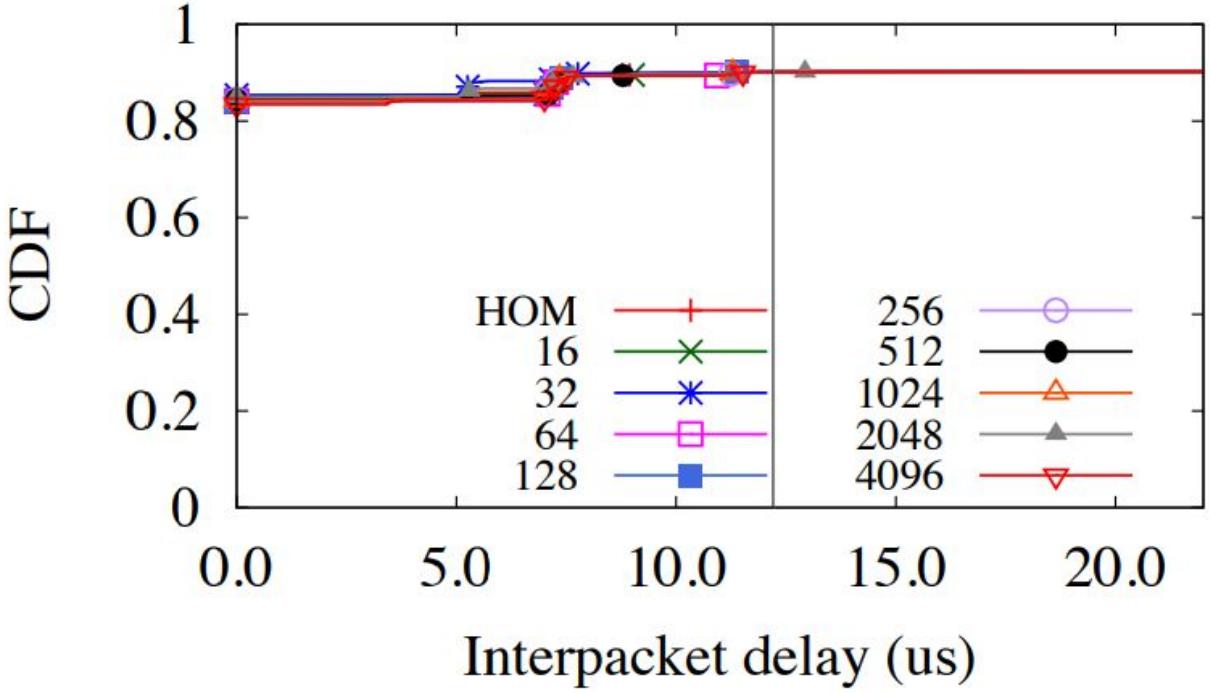
Table 3: Evaluated ε values in the number of $/\mathtt{I}/s$ and their corresponding time values in nanosecond.

Results - detection



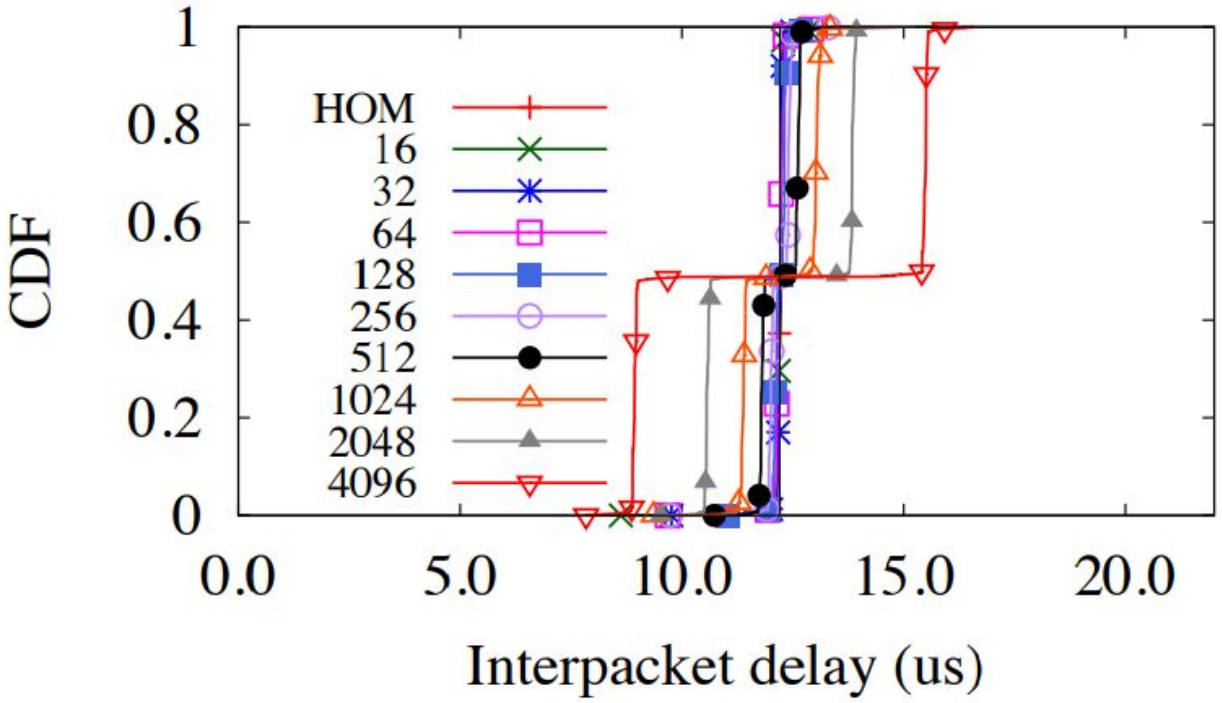
(a) Kernel timestamping.

Results - detection



(b) Zero-copy timestamping.

Results - detection



(c) Hardware timestamping.

Conclusions

- Chupja can provide reliable covert communication channel over the 10Gb Ethernet even if the end-points are far away
- with high-bandwidth - ~81kbps
- Untraceable on software level
- PHY level hardware (scanners or jammers) required to prevent leak of data
- Transfer with small ε (64B packet, 1Gbps, ε <= 128/I/) was undetectable by hardware

Code

In theory: sonic.cs.cornell.edu

Code

In theory: sonic.cs.cornell.edu

In practice:

Download

- SoNIC Hardware Firmware (requires a SoNIC board and Altera Quartus II).
- SoNIC kernel module and user-space apps
    - Beta version (tested on Linux Kernel 2.6.38) (will be available soon)

Since 2014

Thank you!

Questions?