Université Libre de Bruxelles

Implementation of High-Level Cryptographic Protocols using a SoC platform

June 24th, 2015

Quentin Delhaye

Contents

- Internet of things
- More connections, less power, same security
- Work done with Barco Silex

Objectives

- Real life use cases.
- Decrease CPU load.
- Improve performance.

Cryptographic protocols

VPN

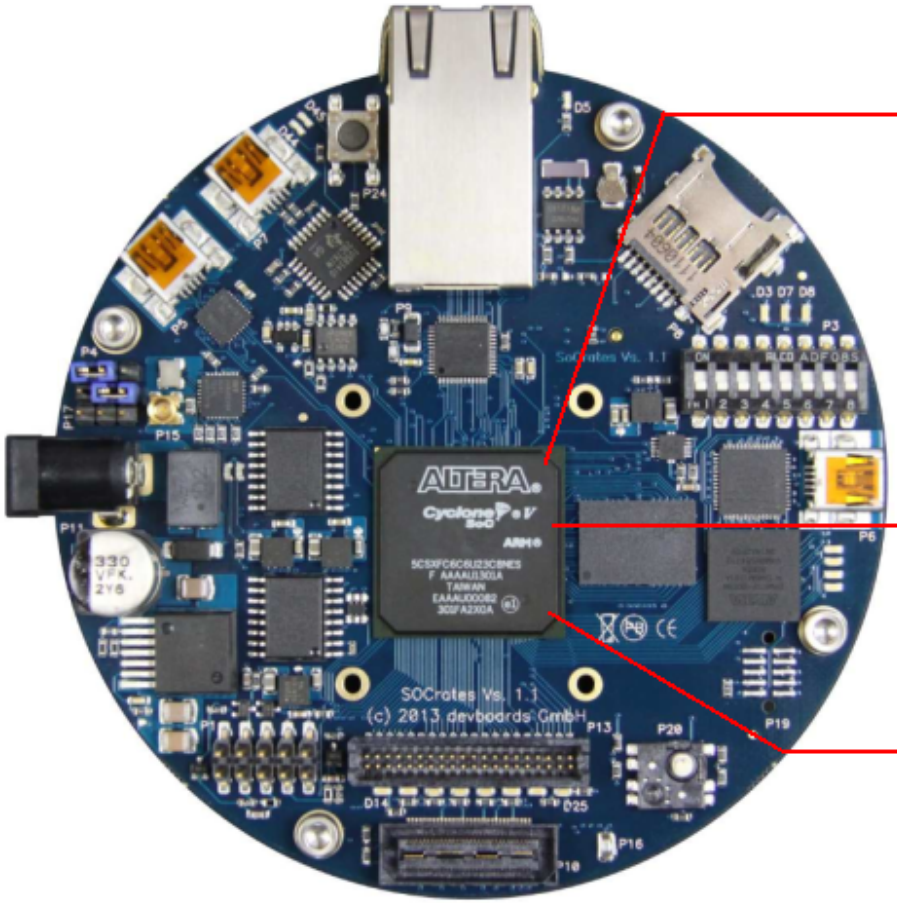
- TLS
- IPsec

Schemes

- AES
- SHA-2
- Diffie-Hellman
- RSA

Contents

SoCrates

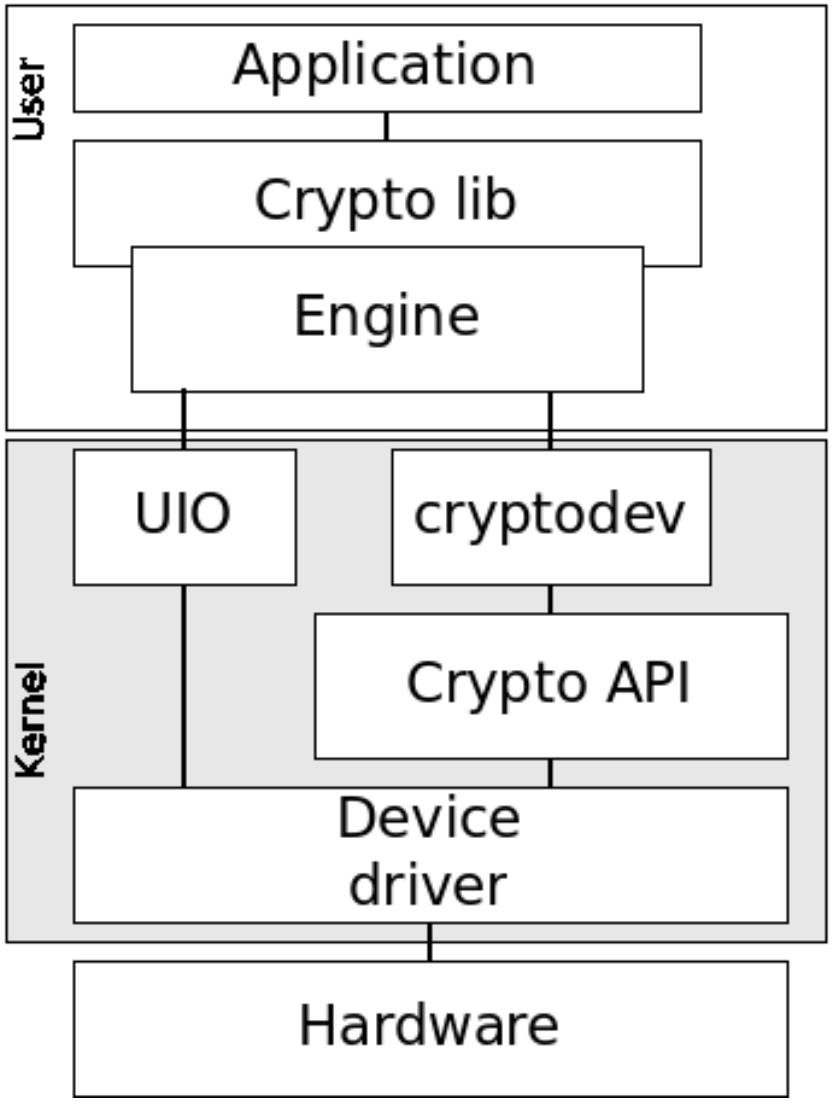


ARM Cortex A9
dual core
800MHz

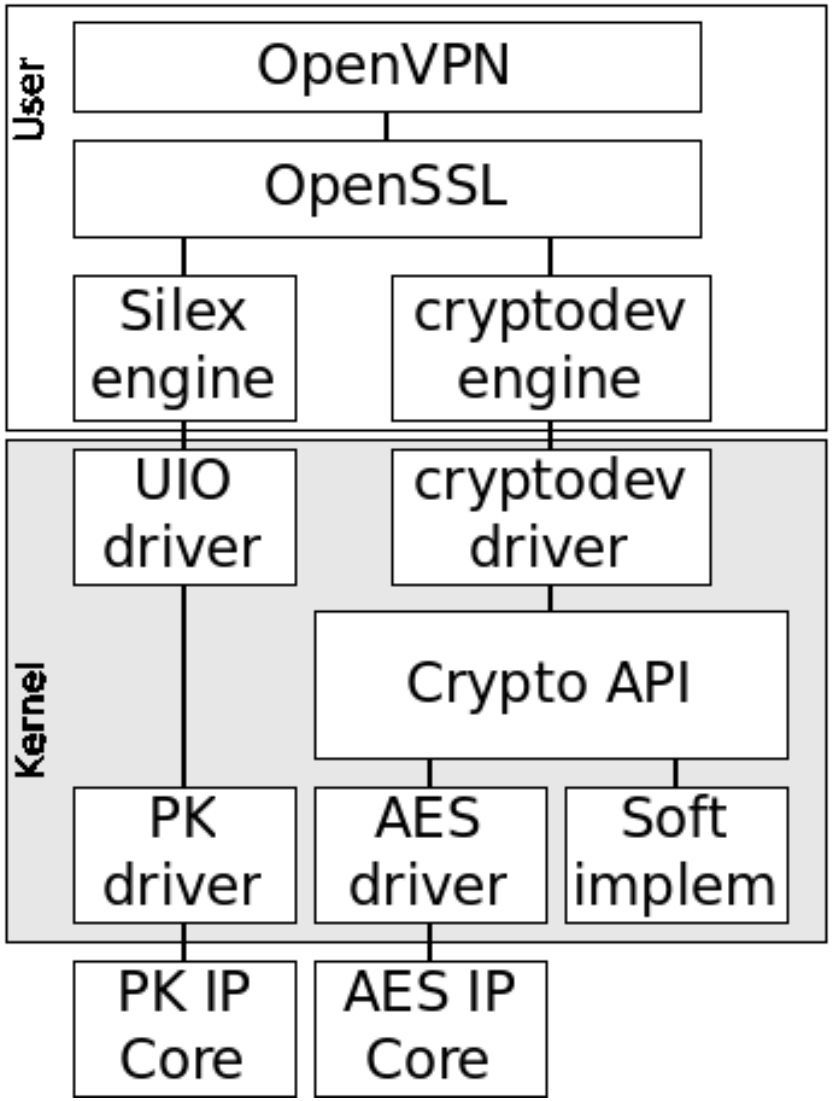
Altera Cyclone V

Barco Silex IP Cores
PK
AES

Linux structure

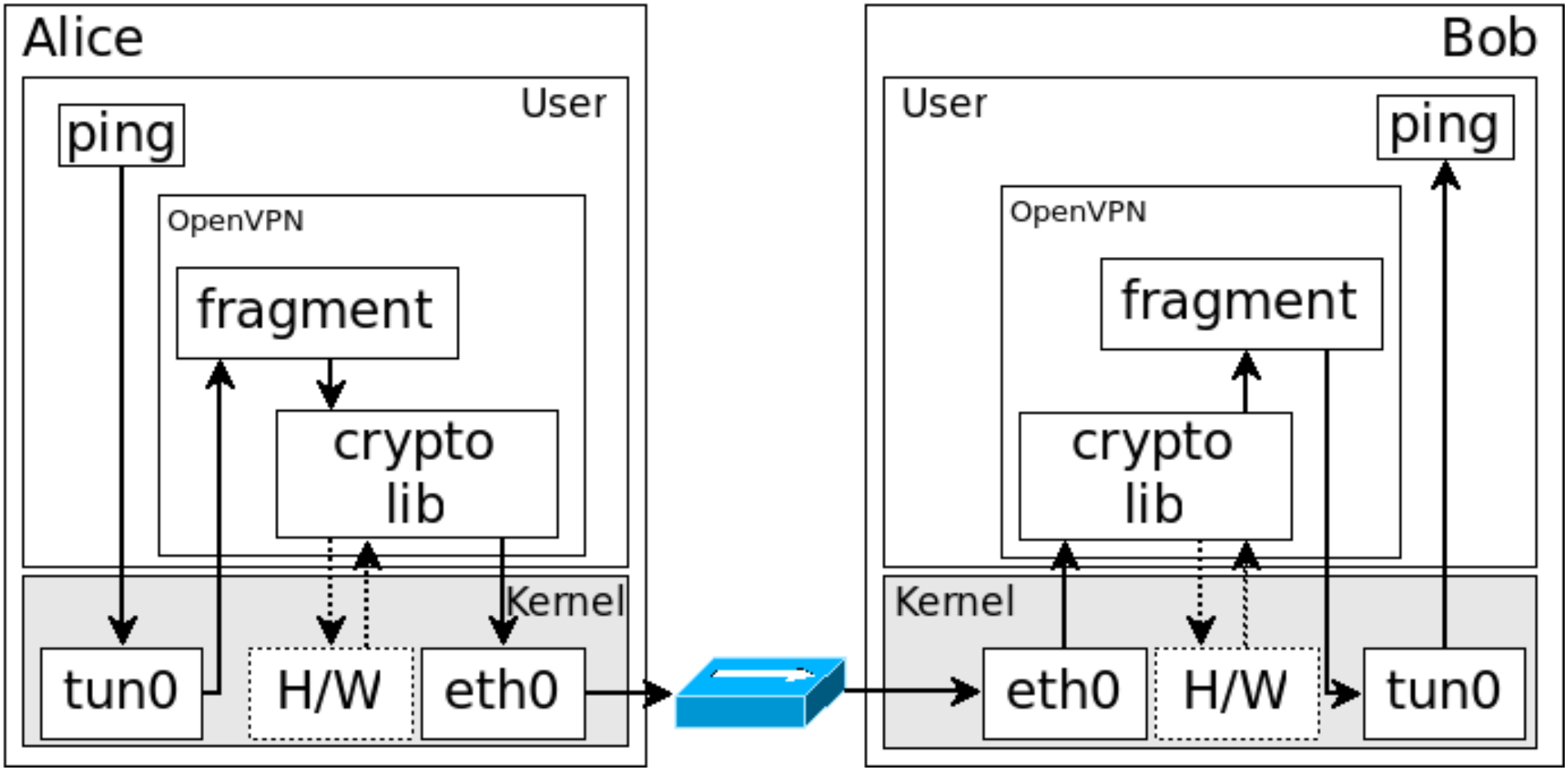


Linux structure (Cont'd)

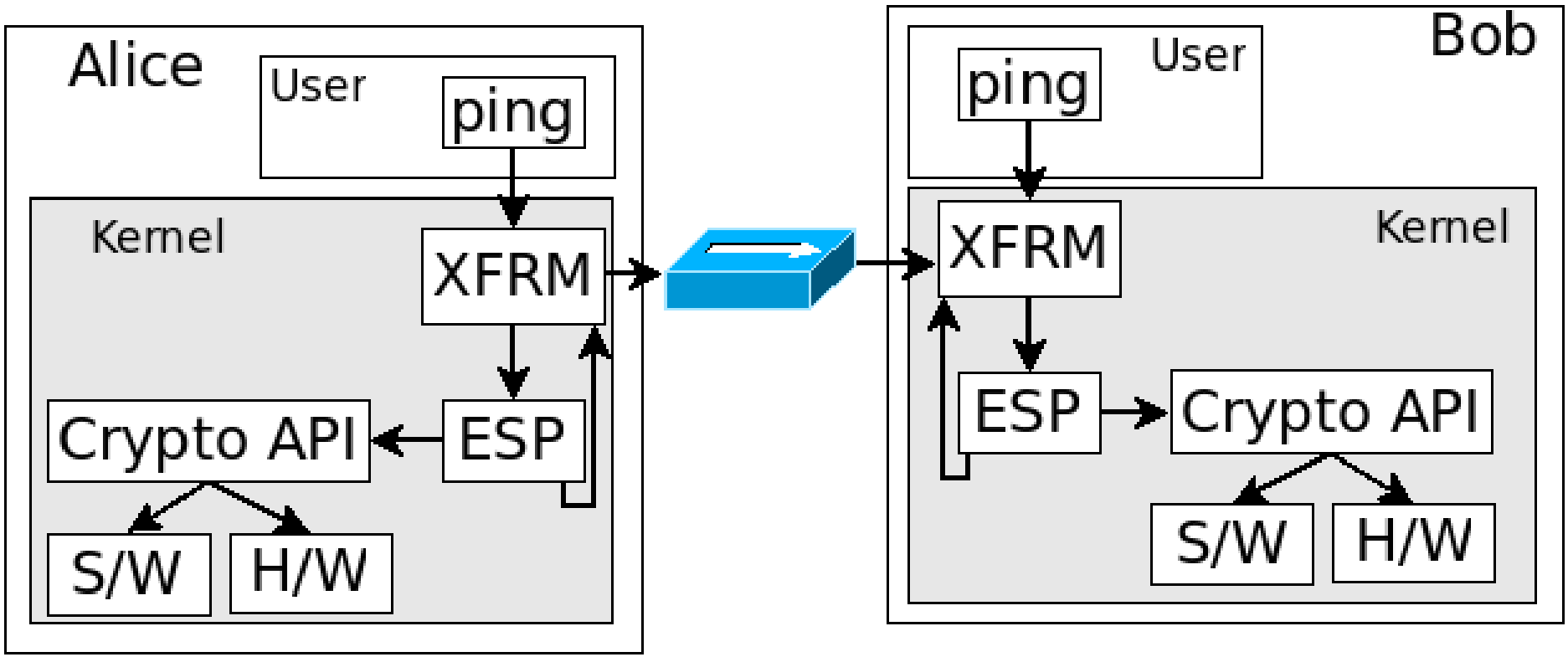


Contents

OpenVPN

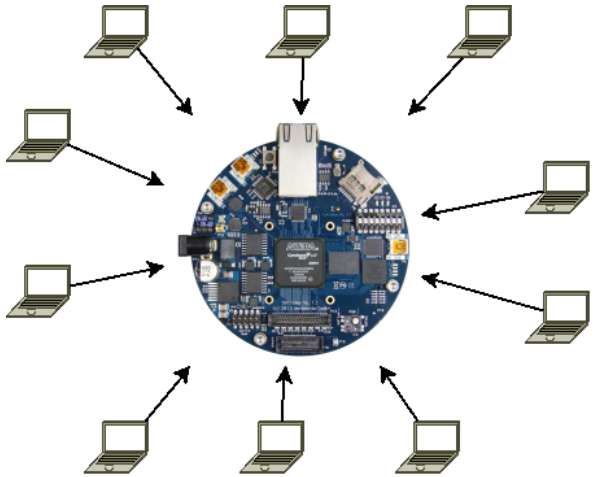

Alice
User
ping
OpenVPN
fragment
crypto
lib
Kernel
tun0
H/W
eth0
Bob
User
ping
OpenVPN
fragment
crypto
lib
Kernel
eth0
H/W
tun0

IPsec


Alice
User
ping
Kernel
XFRM
Crypto API
ESP
S/W
H/W
Bob
User
ping
Kernel
XFRM
ESP
Crypto API
S/W
H/W

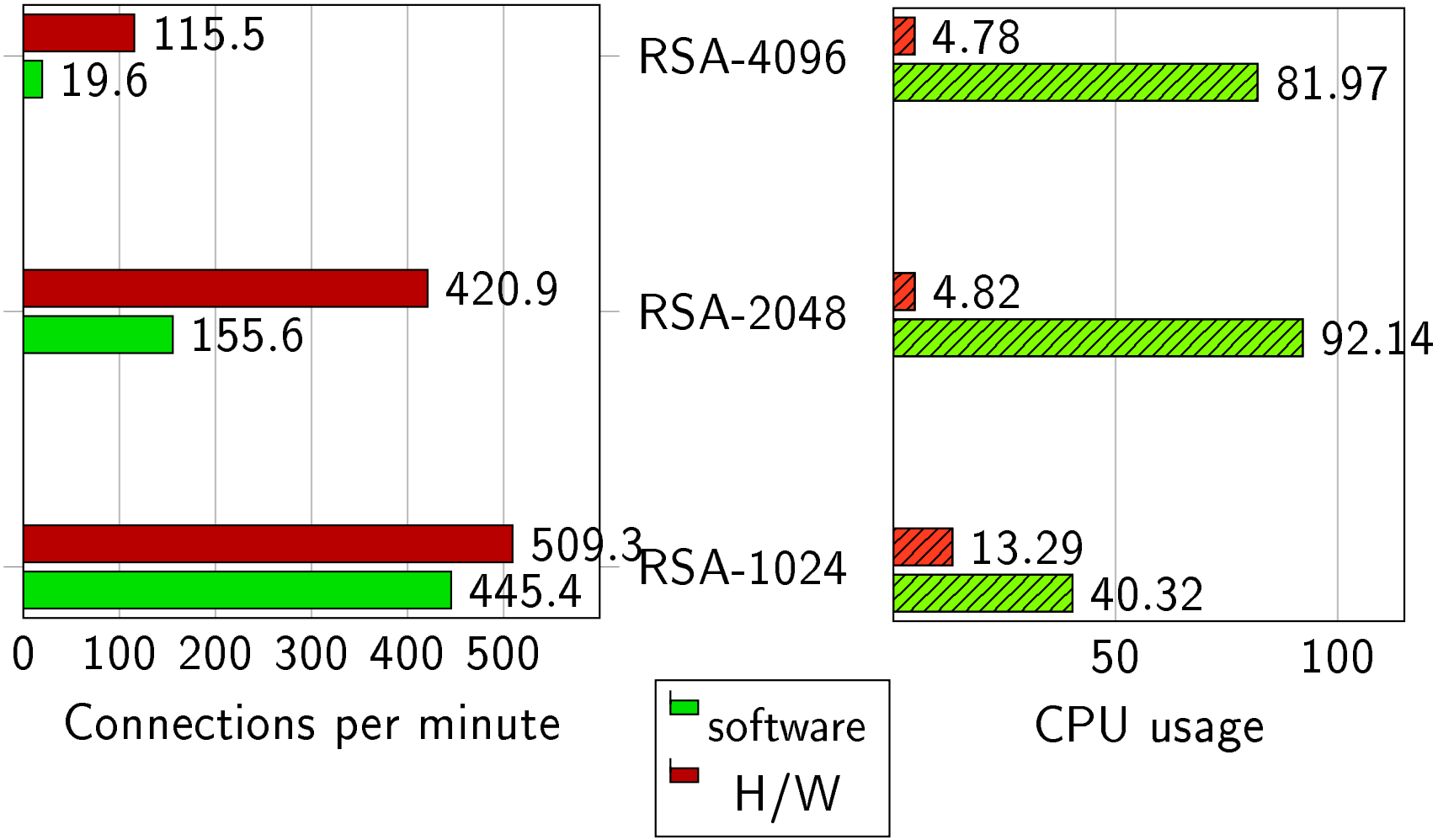
Contents

TLS connections – Context

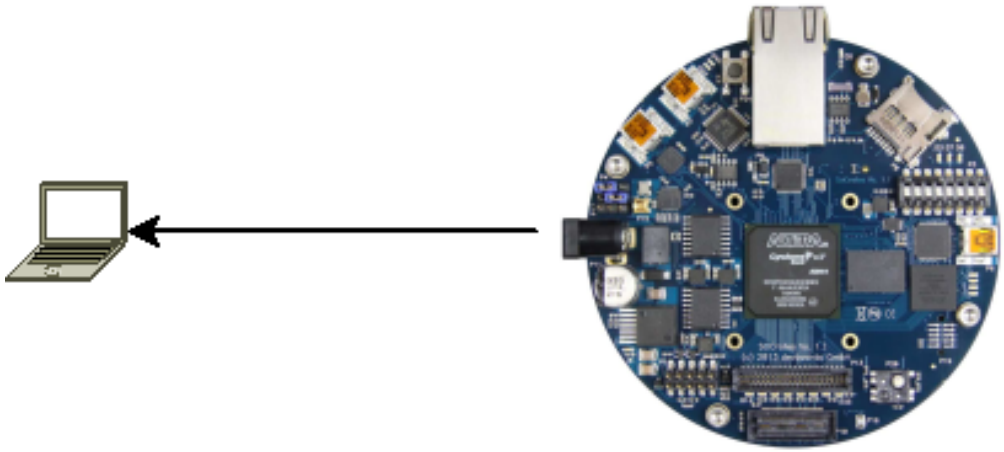


- 1 server, 10 clients
- 1-second connections
- RSA-1024/2048/4096
- OpenVPN

TLS connections – OpenVPN

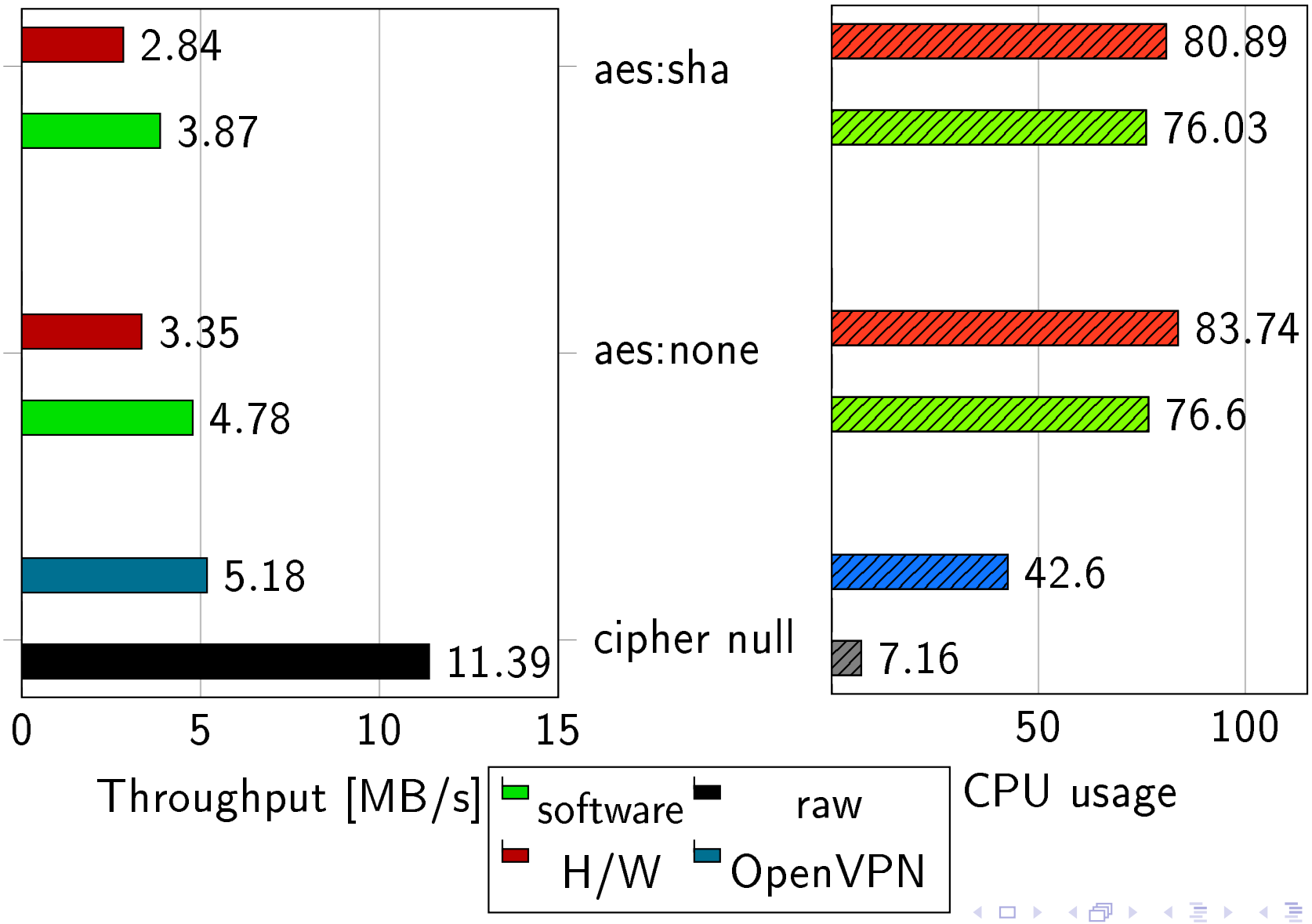


File transfer – Context

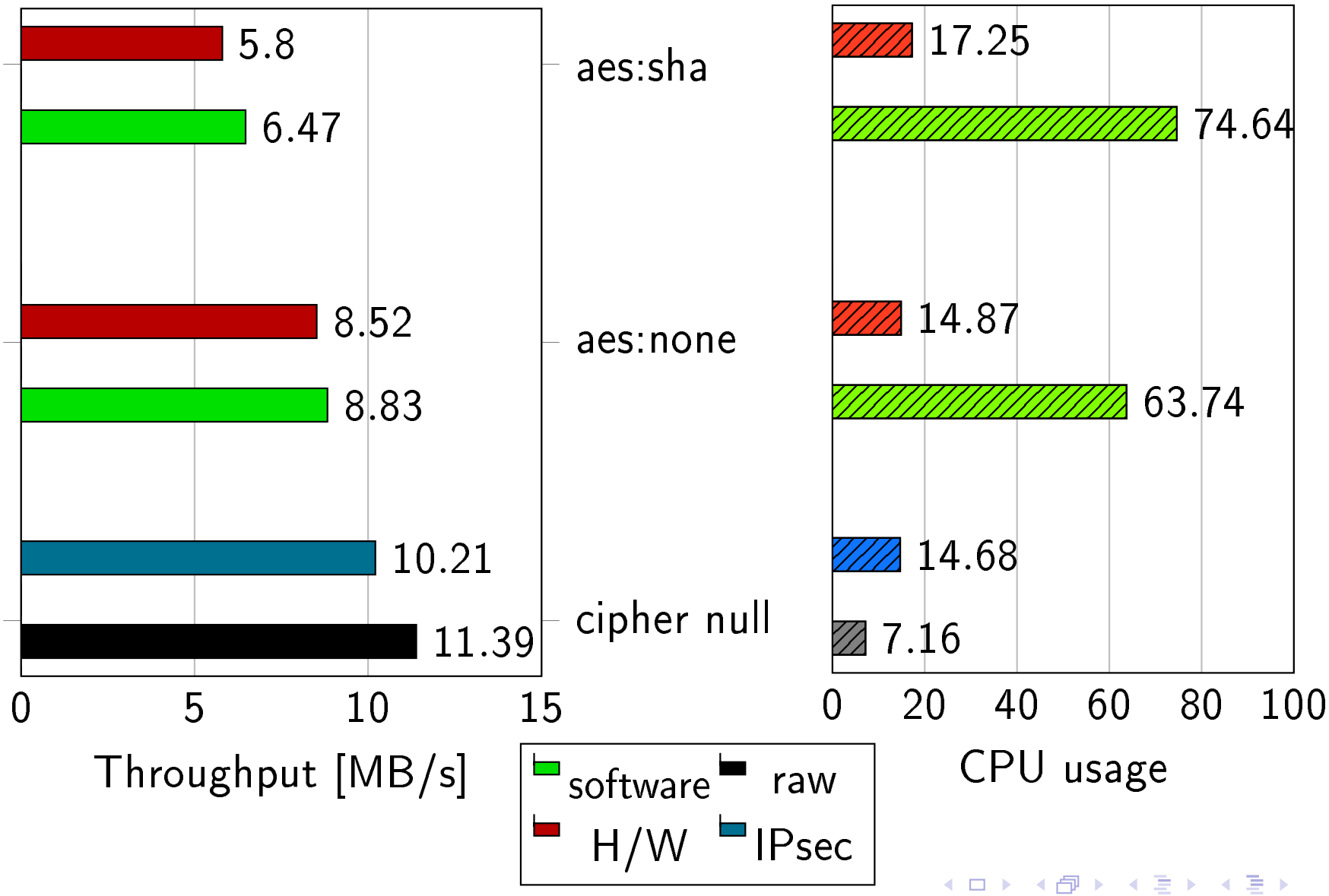


- 128MB file
- AES-256-CBC/SHA-256
- OpenVPN/IPsec

File transfer – OpenVPN



File transfer – IPsec



Results interpretation

- OpenVPN is single-threaded
- OpenVPN software overhead

Conclusion

TLS connections

- connections $\times 6$
- CPU usage $\div 20$

File transfer

- Drop OpenVPN
- Performance -10%
- CPU usage $\div 4$

Conclusion

- Stay in the kernel
- GCM is comming
- Ongoing development
  - Test better hardware
  - Improve the drivers

Software GCM

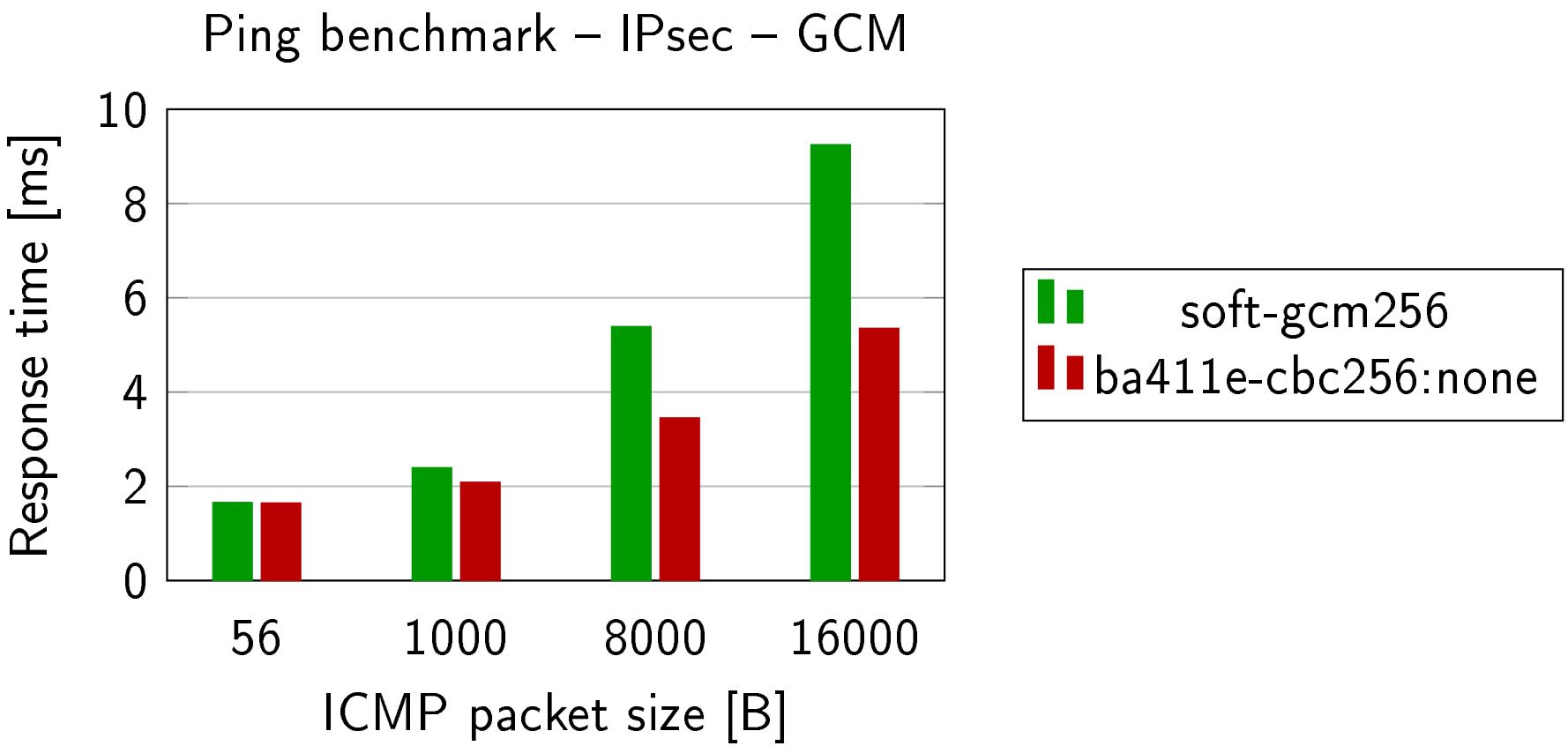


Figure: Software: asm kernel module mode GCM
Hardware: AES IP core mode CBC

OpenVPN file transfer – AES-256-CBC – MAC none

- Hardware top 3:
  1. Kernel memory handling
  2. Context switch
  3. IRQ restore
- Software top 3:
  1. AES encryption
  2. IRQ restore
  3. OpenVPN encryption routine

OpenSSL benchmark