Comparing TCP performance of tunneled and non-tunneled traffic using OpenVPN

Berry Hoekstra | Damir Musulin
OS3
Supervisor: Jan Just Keijser
Nikhef

Outline

- Introduction
- Approach
- Research
- Results
- Conclusion
- Questions?

Introduction

- Virtual Private Networks
  - Secure connection over an insecure network
  - SSL, IPsec, PPTP and L2TP are the most popular VPN solutions
  - Packets are encapsulated into packets on a lower layer
- OpenVPN
  - SSLv3/TLSv1 based VPN solution
  - Able to saturate 100 Mbps
  - Performance issues with 1 Gbps
    - Not much documented research available
  - OpenSSL for encryption
  - TUN/TAP driver for tunneling

Research Question

- *"What are the causes of the network performance loss when using OpenVPN at Gigabit speed?"*

Sub-questions:

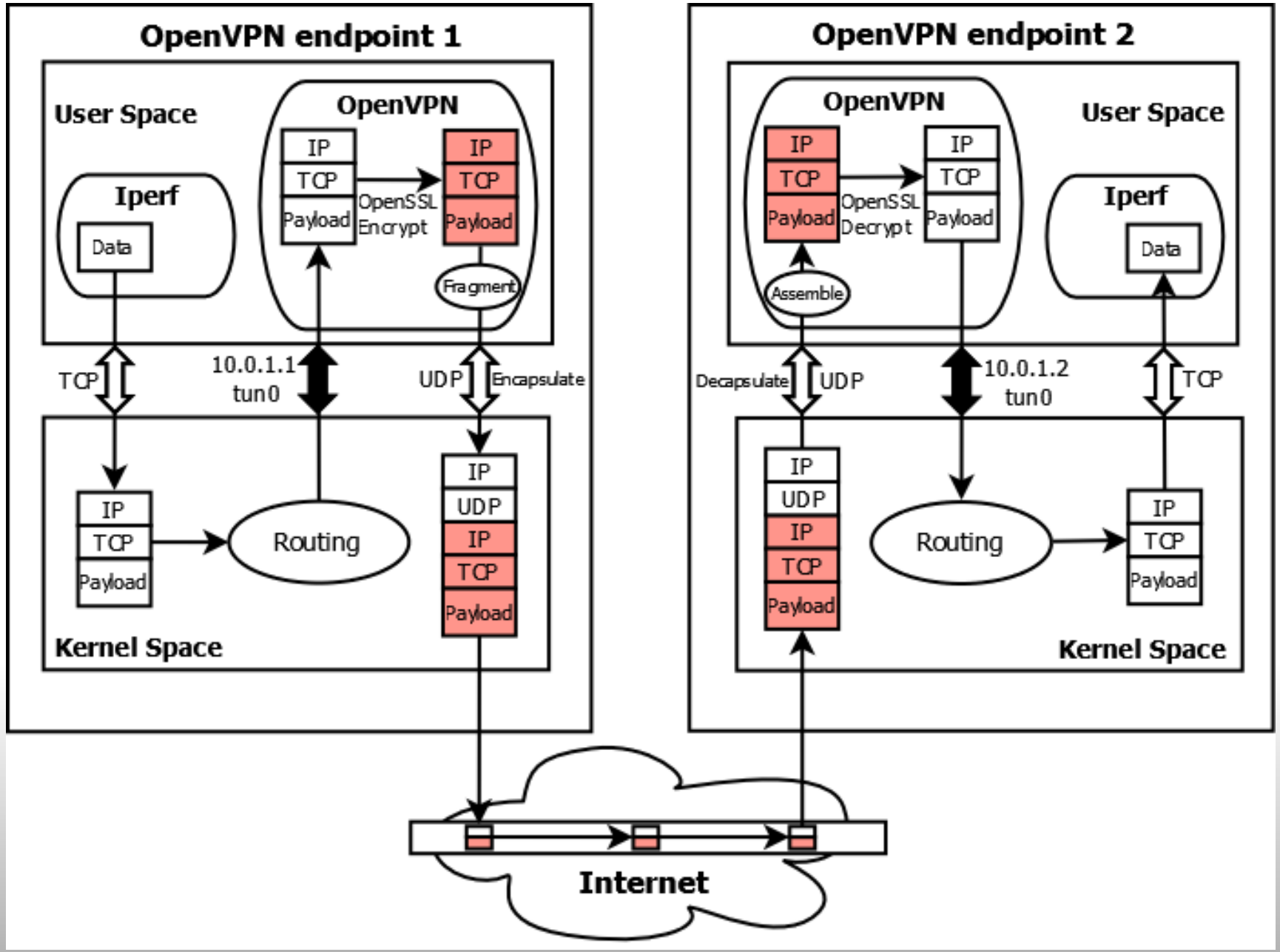
- *What is the effect of using different encryption and authentication methods or parameters in OpenVPN?*
- *Is the same performance hit found on other OpenSSL-based tunnel solutions?*
- *Is the same performance hit found on other operating systems (e.g. FreeBSD)?*
- *What are the possibilities to mitigate slow OpenVPN network performance?*

Problem definition

- Unable to saturate 1 Gbps over a VPN tunnel
  - Even with no encryption and signing with default settings

- Suspected culprits:
  - Inefficient cryptographic functions
  - OS context switching
  - TUN/TAP driver overhead
  - Context switching

OpenVPN packet flow

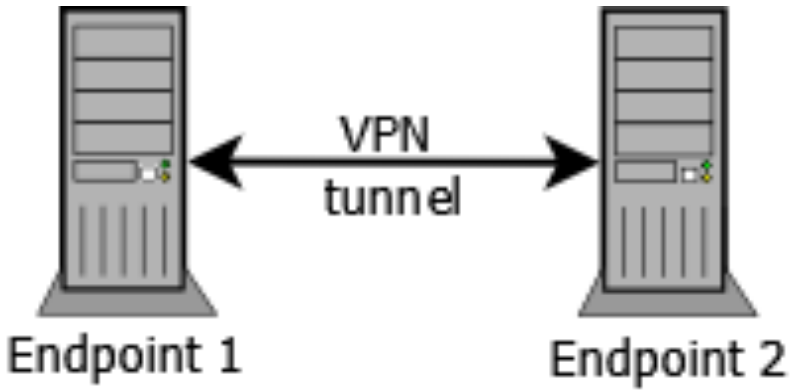


Methodology (1)

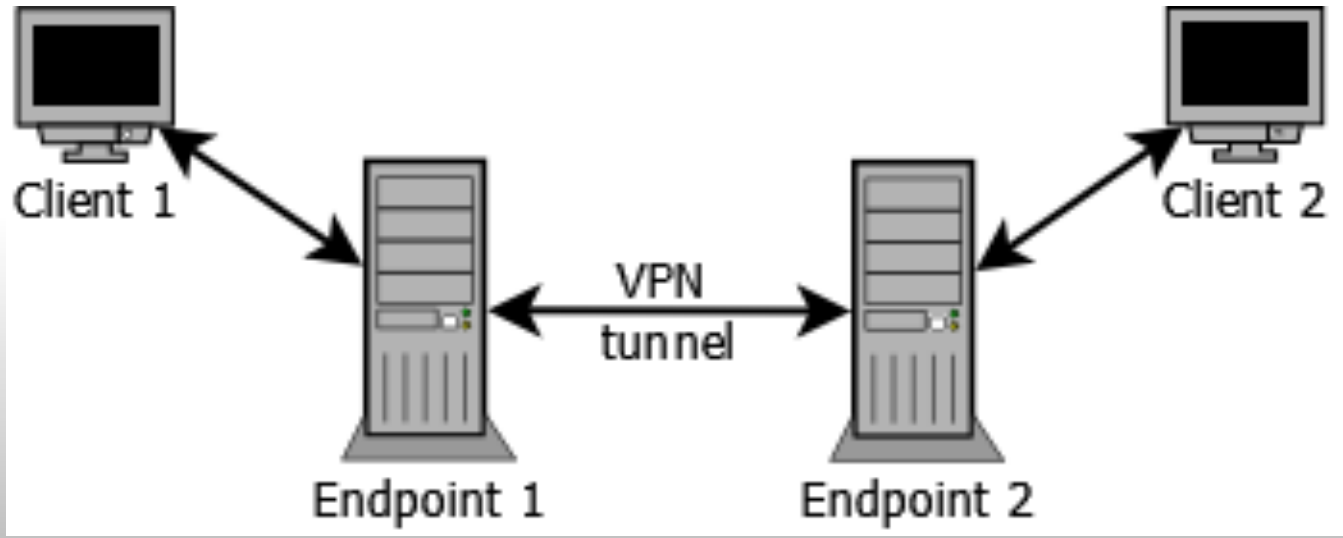
- Perform throughput measurements using Iperf
  - Using a control script
  - On different infrastructures
  - Perform OpenSSL speed tests
- What are the effects in throughput when:
  - Using different parameters
  - Using a different OpenSSL version
  - On a different infrastructure
  - On a different operating system
- Compare against similar VPN solutions
  - Vtun
- Source code analysis
  - OpenVPN / OpenSSL functionality
  - TUN/TAP driver

Lab setup

- Dell R210
- Intel Xeon L3426
  - 4/8 cores @ 1.87 GHz
- 8GB memory
- 2x Broadcom NIC



Setup 1: Endpoint to endpoint



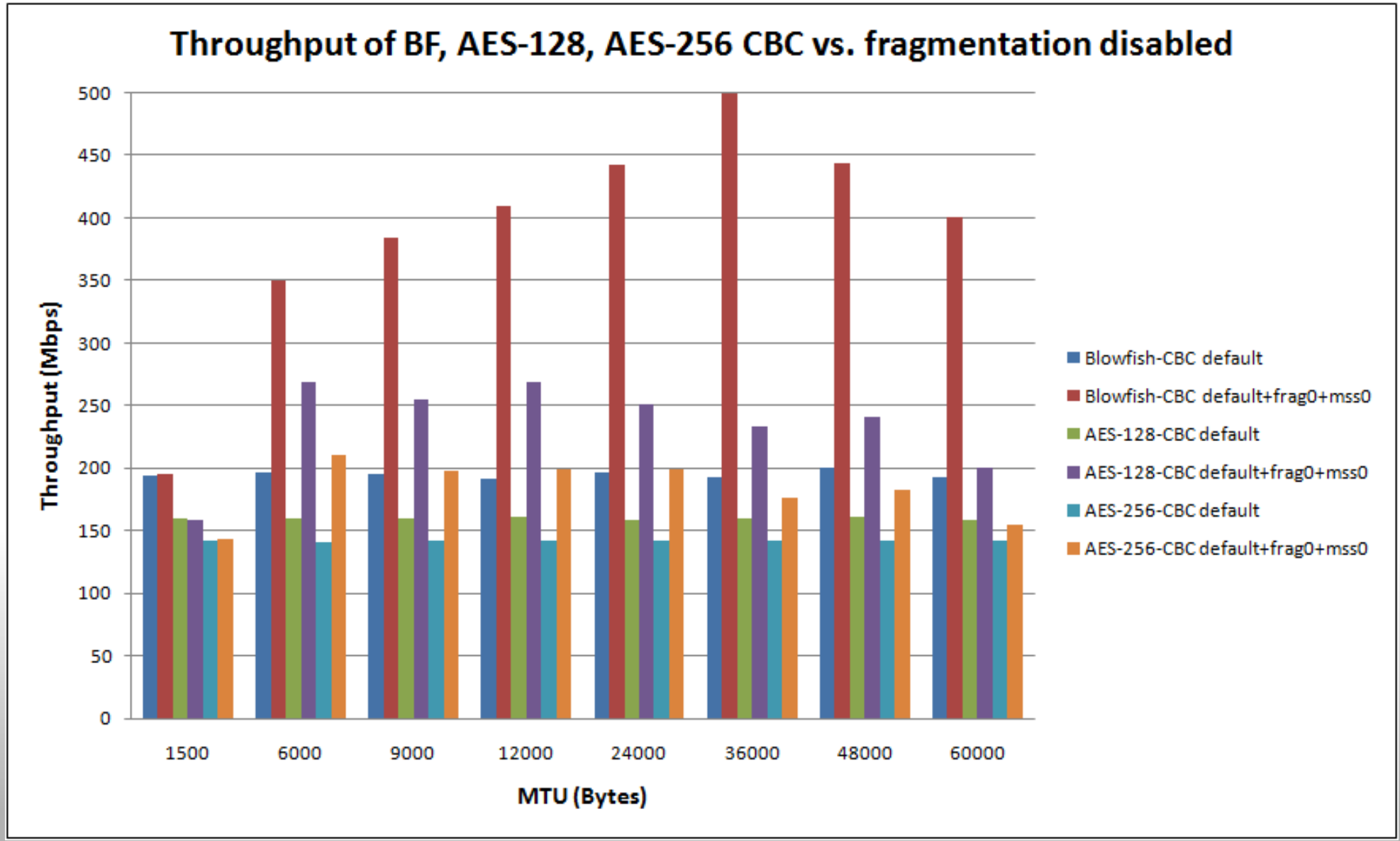
Setup 2: Client to client

Methodology (2)

- Ciphers
  - Blowfish-128-CBC (default)
  - AES-128-CBC
  - AES-256-CBC
- HMAC signing
  - SHA-1 vs. MD5
- Increasing TUN MTU sizes
  - Increases the block size towards OpenSSL
  - Encryption is done more efficient
- OpenVPN fragmentation options
  - Disabled, fragmentation is done at kernel level
  - Increases throughput! (between endpoints)

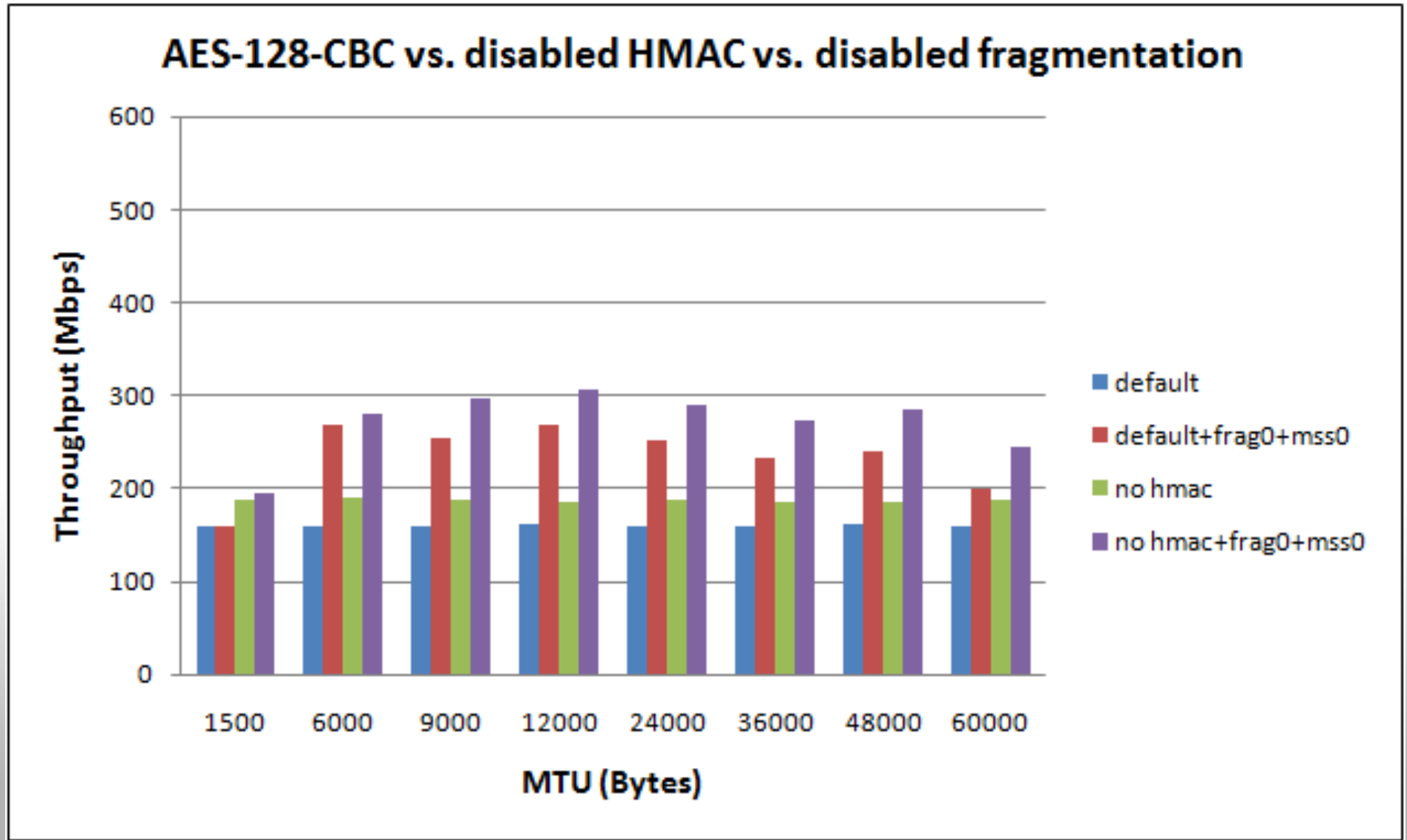
Results (1)

Different ciphers: BF +150%, AES +30%-80%



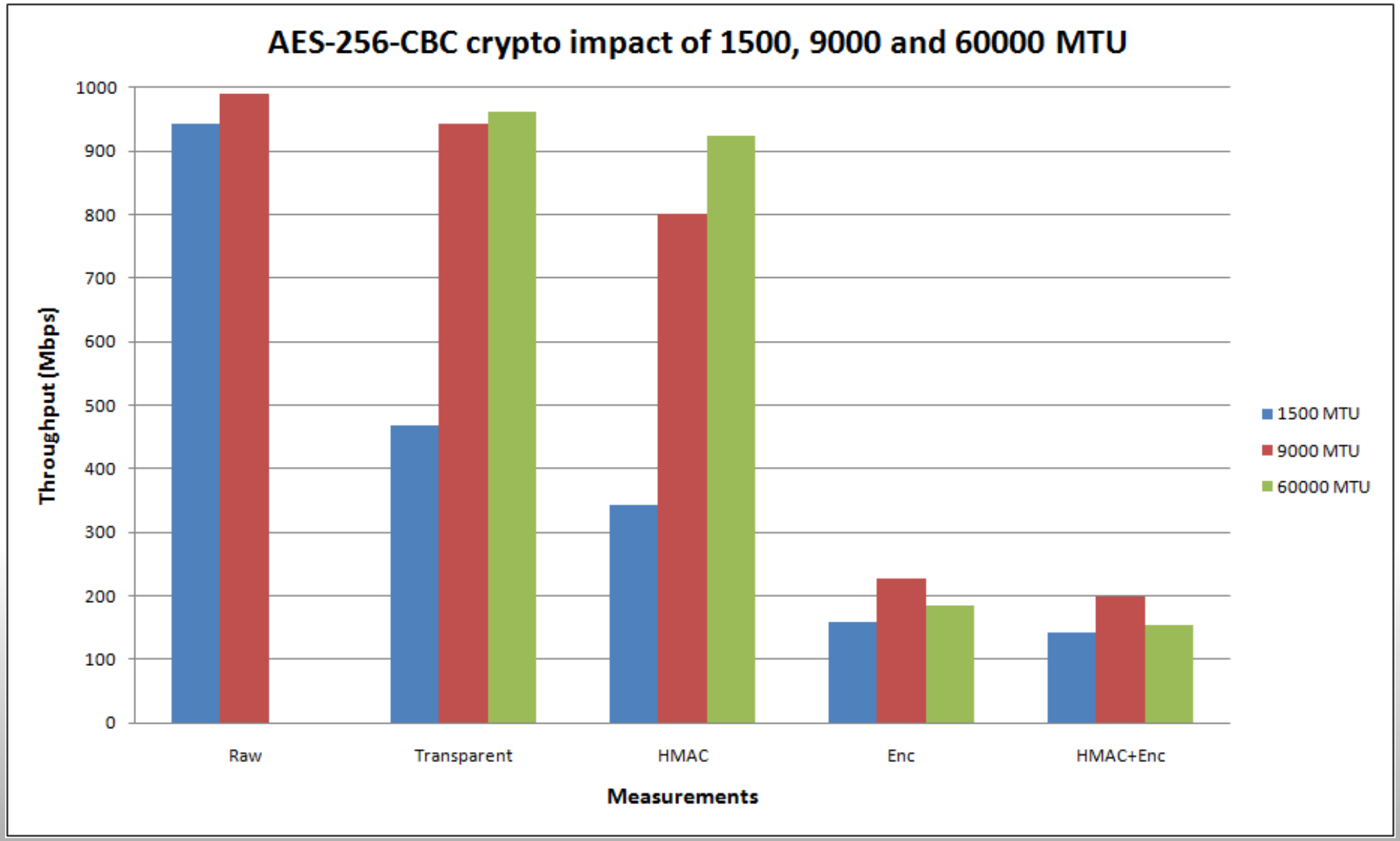
Results (2)

HMAC disabled +10%-20%
Fragmentation disabled + ~40%



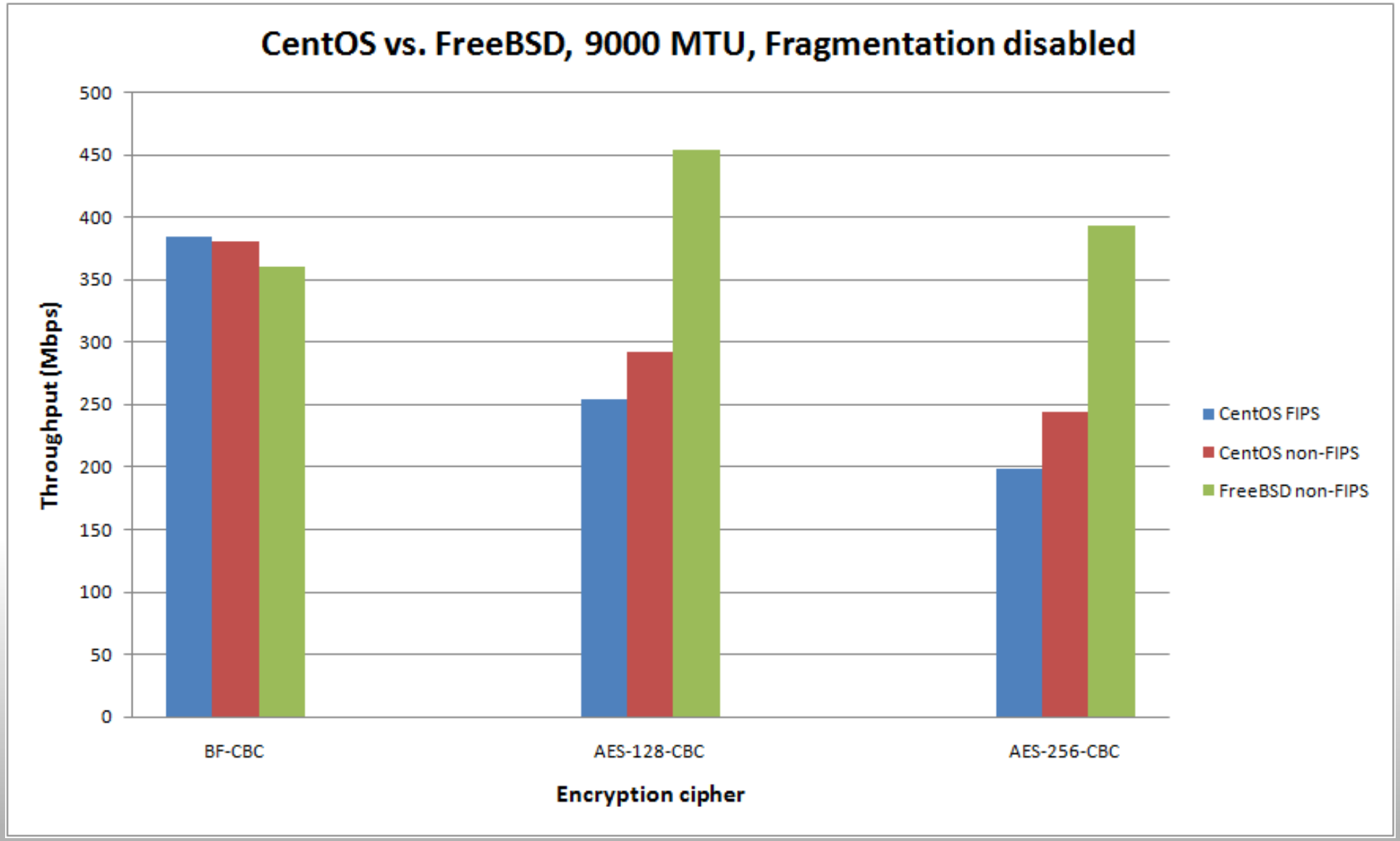
Results (3)

Crypto impact on AES-256-CBC



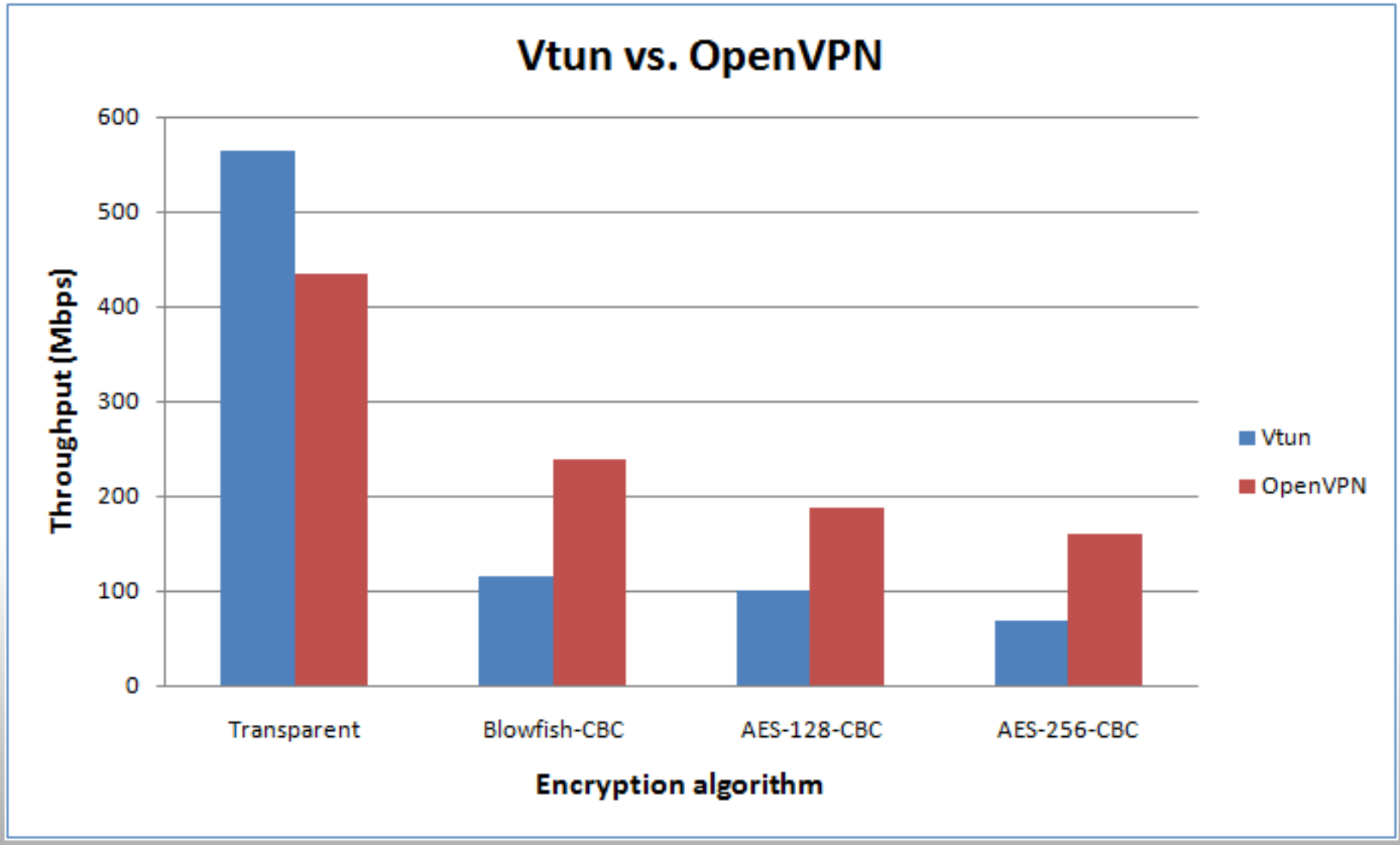
Results (4)

CentOS vs. FreeBSD: +50%-60%!



Results (5)

OpenVPN vs. other OpenSSL solution: Vtun



Conclusions (1)

- OpenSSL is not capable to encrypt at 1 Gbps
  - BF-128 **~500**, AES-128 **~800**, AES-256 **~700** Mbps
- OpenVPN results show inefficient handling
  - Even with the internal fragmentation disabled
  - BF-128 **~400**, AES-128 **~200**, AES-256 **~155** Mbps
- OpenVPN needs high TUN MTU values for most efficient handling
- TUN/TAP driver plays a role in causing more overhead
  - Context switching
  - Mitigated by running in kernel space like IPsec

Conclusions (2)

- Tunnel performance can be optimized
  - Only on endpoint to endpoint setups
  - Hard to improve performance on routed setup
    - Clients deliver packets with a small MTU to endpoints
- Fragmentation options matters
  - Only for endpoint to endpoint setups
- FreeBSD shows a throughput increase of ~80%
  - Due to inefficient FIPS version of OpenSSL on CentOS
    - Fixed in OpenSSL 1.0.0 (default in Fedora)
  - Against CentOS, FreeBSD still outperforms with 50% to 60%
    - Using the same OpenSSL version

Conclusions (3)

"What are the causes of the network performance loss when using OpenVPN at Gigabit speed?"

- There is a relation between the OpenSSL version and OpenVPN throughput
- Encryption routines of OpenVPN are inefficient
- OpenVPN fragmentation options cause a lot of overhead
  - Calculation, reassemble, and sequence no. administration
- Different performance measured on different operating systems
- OpenVPN source code contains a lot of branching
  - if {..} else {..} if {..} else {..} if {..} else {..} if {..} else {..}
  - Performance hit on CPU

Future work

- Hardware acceleration
  - AES-NI instruction set
  - Graphics cards
  - Cryptographic cards
- Kernel Mode Linux
  - Eliminate context switching
- TAP-Win32 driver
- Profiler
  - Low level Linux performance counters
  - Steap learning curve
- CPU affinity
  - No multi-socket hardware available
- 10 Gbps performance measurements
  - TCP Tuning is needed to get near-linespeed
  - Look into UDP offloading

Questions?