Software Benchmarking of the 2nd round CAESAR Candidates

Ralph Ankele[1], Robin Ankele[2]

[1]Royal Holloway, University of London, UK [2]University of Oxford, UK

September 27, 2016

Directions in Authenticated Ciphers - Nagoya, Japan

Motivation[1]

Use Case 1: Lightweight applications (resource constrained environments)

Use Case 2: High-performance applications

- critical: efficiency on 64-bit CPUs (servers) and/or dedicated hardware
- desirable: efficiency on 32-bit CPUs (small smartphones)
- desirable: constant time when the message length is constant
- message sizes: usually long (more than 1024 bytes), sometimes shorter

Use Case 3: Defense in depth

[1] CAESAR usecases on CAESAR mailing list (16. July 2016) by Dan J. Bernstein: `https://groups.google.com/forum/#!topic/crypto-competitions/DLv193SPSDc`

Overview

1. Classification of the 2^{nd} round CAESAR Candidates
2. Software Optimizations
3. Benchmarking Framework
4. Results
5. Conclusions

Classification of the 2nd round CAESAR Candidates

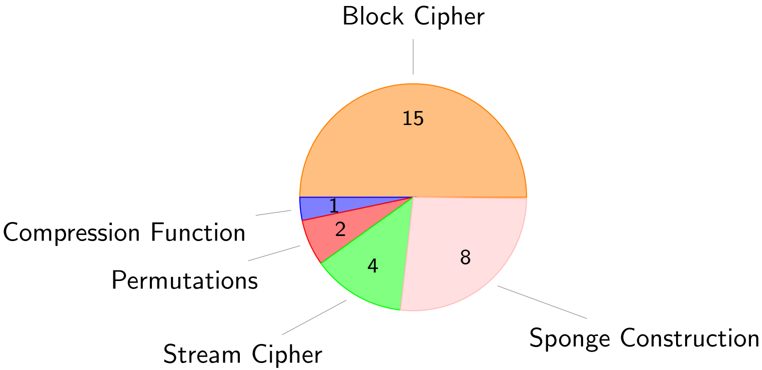
1. Classification of the 2nd round CAESAR Candidates
2. Software Optimizations
3. Benchmarking Framework
4. Results
5. Conclusions

CAESAR competition

CAESAR Round 2 candidates

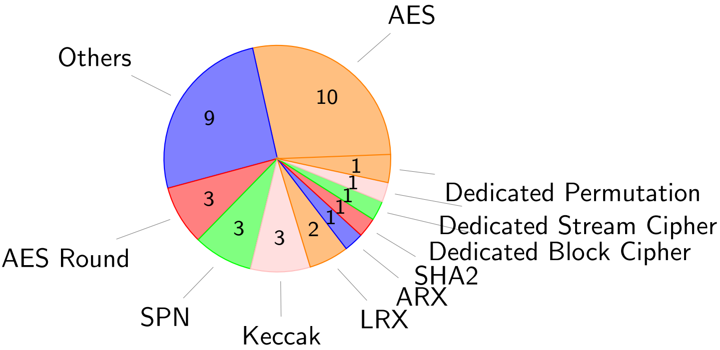
ACORN	AEGIS	AES-COPA	AES-JAMBU	AES-OTR
AEZ	Ascon	CLOC	Deoxys	ELmD
HS1-SIV	ICEPOLE	Joltik	Ketje	Keyak
MORUS	Minalpher	NORX	OCB	OMD
PAEQ	POET	PRIMATEs	SCREAM	SHELL
SILC	STRIBOB	Tiaoxin	TriviA-ck	π-Cipher

Type

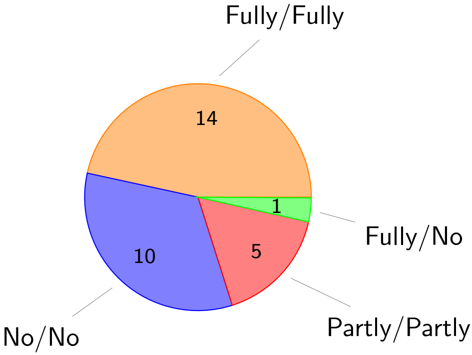


Block Cipher
15
1
Compression Function
2
Permutations
4
Stream Cipher
8
Sponge Construction

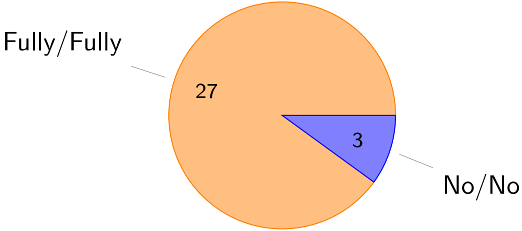
Underlying Primitive



Parallel Encryption/Decryption

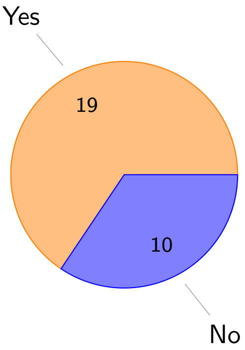


Online Encryption/Decryption

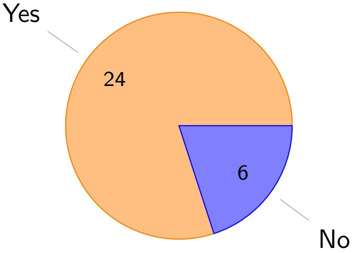


Encryption of a message block M_i only depends on message blocks $M_1 \dots M_{i-1}$.

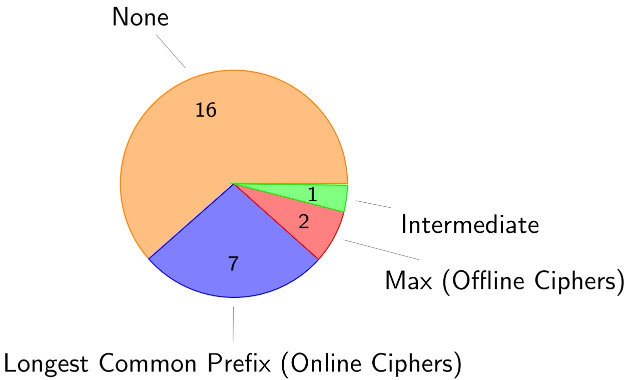
Inverse Free



Security Proof



Nonce-Missuse Resistance

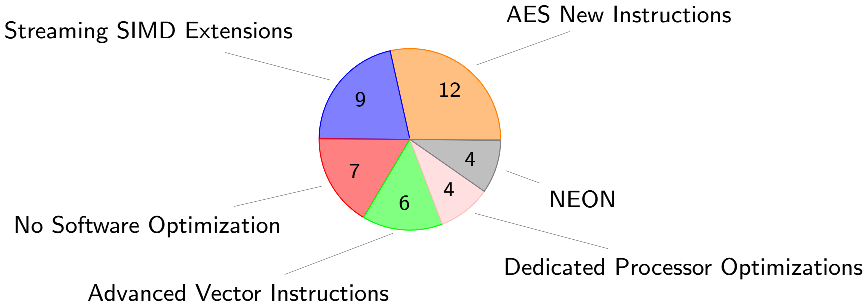


Longest common prefix: an adversary can observe the longest common prefix of messages for repeated nonces

Max: the repetition of nonces only leak the ability to see a repeated message

Software Optimizations

Software Optimizations



AES-New Instructions

Instructions

- Introduced with Intel® 2010 Westmere microarchitecture
- Consists of 6 new instructions that are implemented in hardware
- Four instructions for encryption/decryption (*i.e.* `AESENC`, `AESENCLAST`, `AESDEC`, `AESDECLAST`)
- Two instructions for the keyschedule (*i.e.* `AESKEYGENASSIST`, `AESIMC`)

Performance

- 10 times faster for parallel modes (*i.e.* CTR)
- 2-3 times faster for non-parallel modes (*i.e.* CBC)

Security

- Improved security against side channel attacks [Gue12]

AES-New Instructions

Instructions

- Introduced with Intel® 2010 Westmere microarchitecture
- Consists of 6 new instructions that are implemented in hardware
- Four instructions for encryption/decryption (*i.e.* `AESENC`, `AESENCLAST`, `AESDEC`, `AESDECLAST`)
- Two instructions for the keyschedule (*i.e.* `AESKEYGENASSIST`, `AESIMC`)

Performance

- 10 times faster for parallel modes (*i.e.* CTR)
- 2-3 times faster for non-parallel modes (*i.e.* CBC)

Security

- Improved security against side channel attacks [Gue12]

AES-New Instructions

Instructions

- Introduced with Intel® 2010 Westmere microarchitecture
- Consists of 6 new instructions that are implemented in hardware
- Four instructions for encryption/decryption (*i.e.* `AESENC`, `AESENCLAST`, `AESDEC`, `AESDECLAST`)
- Two instructions for the keyschedule (*i.e.* `AESKEYGENASSIST`, `AESIMC`)

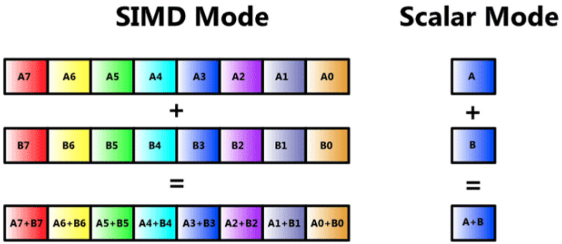
Performance

- 10 times faster for parallel modes (*i.e.* CTR)
- 2-3 times faster for non-parallel modes (*i.e.* CBC)

Security

- Improved security against side channel attacks [Gue12]

Streaming SIMD Extensions



Instructions

- Vector-mode operations that enables parallel execution of one instruction on multible data
- 16 · 128-bit registers (xmm0-15)
- Expanded over Intel® processor generations to include SSE2, SSE3/SSE3S and SSE4

Image: `https://software.intel.com/sites/default/files/37208.gif`

Advanced Vector Extensions

Instructions

- Introduced with Intel® SandyBridge microarchitecture
- Extends SSE 128-bit registers with 16 new 256-bit registers (ymm0-15)
- Support of three-operand non-destructive operations (two-operand instructions *e.g.* A = A + B are replaced by three-operand instructions *e.g.* A = B + C)
- AVX2 instructions expand integer vector types and vector shift operations

Performance

- AVX is 1.8 times faster than fastest SSE4.2 instructions [Len14]
- AVX2 is 2.8 times faster than fastest SSE4.2 instructions [Len14]

Advanced Vector Extensions

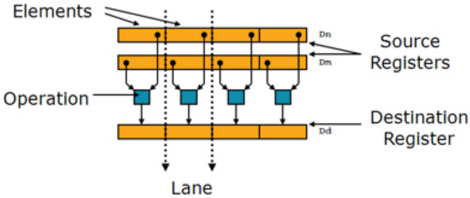
Instructions

- Introduced with Intel® SandyBridge microarchitecture
- Extends SSE 128-bit registers with 16 new 256-bit registers (ymm0-15)
- Support of three-operand non-destructive operations (two-operand instructions *e.g.* $A = A + B$ are replaced by three-operand instructions *e.g.* $A = B + C$)
- AVX2 instructions expand integer vector types and vector shift operations

Performance

- AVX is 1.8 times faster than fastest SSE4.2 instructions [Len14]
- AVX2 is 2.8 times faster than fastest SSE4.2 instructions [Len14]

NEON



Instructions

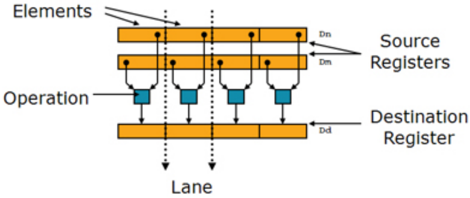
- Advanced SIMD instructions for ARM processors avaliable since CORTEX-A microarchitecture
- $32 \cdot 64$-bit registers (dual view $16 \cdot 128$-bit registers)

Performance

- 2-8 times performance boost [neo]

Image: `http://www.arm.com/assets/images/NEON_ISA.jpg`

NEON



Instructions

- Advanced SIMD instructions for ARM processors avaliable since CORTEX-A microarchitecture
- $32 \cdot 64$-bit registers (dual view $16 \cdot 128$-bit registers)

Performance

- 2-8 times performance boost [neo]

Image: `http://www.arm.com/assets/images/NEON_ISA.jpg`

Benchmarking Framework

1. Classification of the 2^{nd} round CAESAR Candidates
2. Software Optimizations
3. Benchmarking Framework
4. Results
5. Conclusions

High Resolution Methods for CPU Timing Information

High Resolution Timers

- HPET (High Precision Event Timer)
- QueryPerformanceCounter
- `time()` and `clock()` posix functions
- TSC (Timer Stamp Counter)

Timer Stamp Counter

- 64-bit Machine State Register containing the number of cycles since last reset
- `RDTSC` instruction to read out
- Use `CPUID` instruction against out-of-order execution
- Our framework uses `RDTSCP` [Pao10] which is an optimised `RDTSC` + `CPUID`

High Resolution Methods for CPU Timing Information

High Resolution Timers

- HPET (High Precision Event Timer)
- QueryPerformanceCounter
- `time()` and `clock()` posix functions
- TSC (Timer Stamp Counter)

Timer Stamp Counter

- 64-bit Machine State Register containing the number of cycles since last reset
- `RDTSC` instruction to read out
- Use `CPUID` instruction against out-of-order execution
- Our framework uses `RDTSCP` [Pao10] which is an optimised `RDTSC` + `CPUID`

Benchmarking Framework

SUPERCOP [Ber16]

- **S**ystem for **U**nified **P**erformance **E**valuation **R**elated to **C**ryptographic **O**perations and **P**rimitives
- Uses Timer Stamp Counter as Timer (with `RDTSC`)

BRUTUS [Saa16]

- Small codebase, rapid testing cycle
- Uses `clock()` as Timer

Our Framework

- Simple with only focus on Authenticated Encryption schemes
- Optimized Timer Stamp Counter (*i.e.* `RDTSCP`) for accurate timing measurements [Pao10]
- Reduction of noise using single user mode, averaging and median

Benchmarking Framework

SUPERCOP [Ber16]

- **S**ystem for **U**nified **P**erformance **E**valuation **R**elated to **C**ryptographic **O**perations and **P**rimitives
- Uses Timer Stamp Counter as Timer (with `RDTSC`)

BRUTUS [Saa16]

- Small codebase, rapid testing cycle
- Uses `clock()` as Timer

Our Framework

- Simple with only focus on Authenticated Encryption schemes
- Optimized Timer Stamp Counter (*i.e.* `RDTSCP`) for accurate timing measurements [Pao10]
- Reduction of noise using single user mode, averaging and median

Benchmarking Framework

SUPERCOP [Ber16]

- **S**ystem for **U**nified **P**erformance **E**valuation **R**elated to **C**ryptographic **O**perations and **P**rimitives
- Uses Timer Stamp Counter as Timer (with `RDTSC`)

BRUTUS [Saa16]

- Small codebase, rapid testing cycle
- Uses `clock()` as Timer

Our Framework

- Simple with only focus on Authenticated Encryption schemes
- Optimized Timer Stamp Counter (*i.e.* `RDTSCP`) for accurate timing measurements [Pao10]
- Reduction of noise using single user mode, averaging and median

Measurement Setup

MacBook Pro Early 2011
Intel® Core i5-2415M
SandyBridge

Dell Latitude E7470
Intel® Core i5-6300U
SkyLake

- Compiler:
  - clang compiler version 6.1.0 (clang-602.0.53)
  - gcc compiler version 5.4.0 (5.4.0-6ubuntu1-16.04.2)
- Compiler flags: `-Ofast -fno-stack-protector -march=native`
- Operating System in Single User mode to get rid of noise (*e.g.* context switches)
- Calculate the median of 91 averaged timings of 200 measurements [KR11]

Benchmarking Settings and Real-World Usecases

Table: Real-world use case settings for our benchmarking.

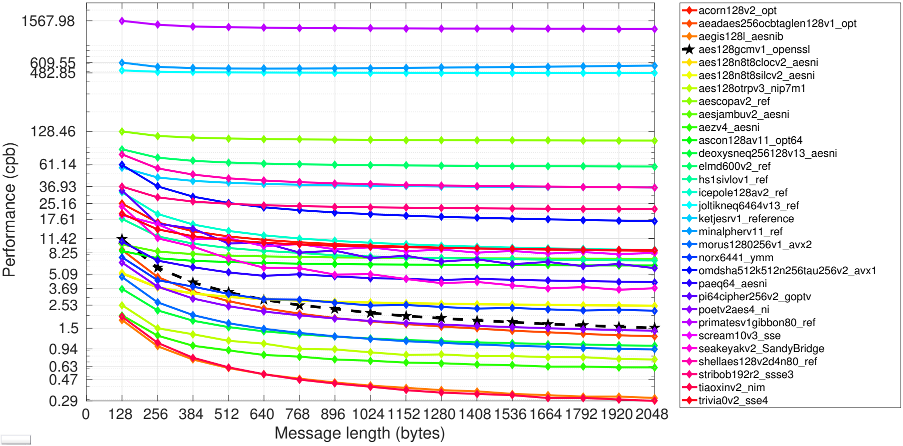
Message Size	Associated Data Size[2]	Comments
1 byte	5 byte	one keystroke (*e.g.* SSH)
16 bytes	5 byte	small payload
557 byte	5 byte	average IP packet size[3]
1.5 kB	5 byte	ethernet MTU, TLS
16 kB	5 byte	max TCP packet size
1 MB	5 byte	file upload

[2] `http://netsekure.org/2010/03/tls-overhead`
[3] `http://slaptijack.com/networking/average-ip-packet-size`

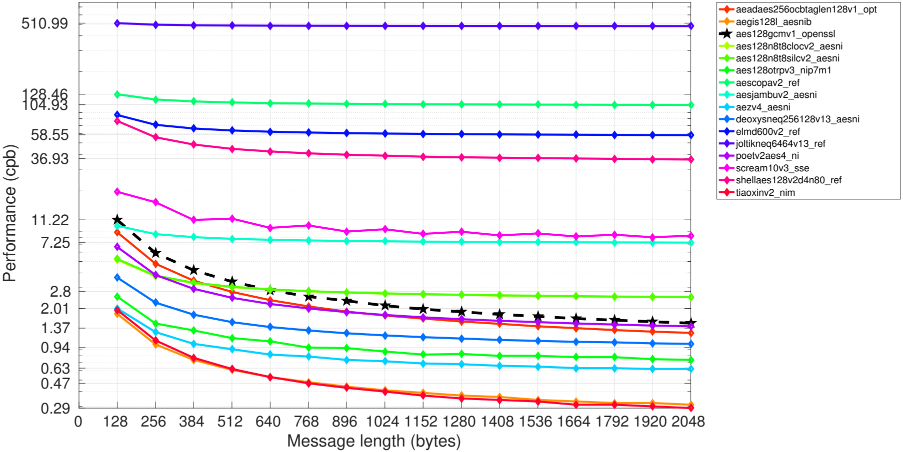
Results

1. Classification of the 2nd round CAESAR Candidates
2. Software Optimizations
3. Benchmarking Framework
4. Results
5. Conclusions

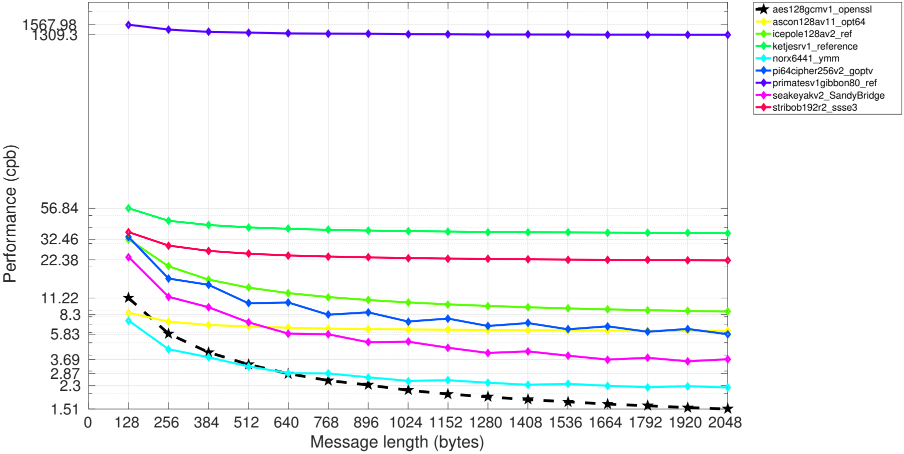
Comparison of all CAESAR 2nd Round Candidates



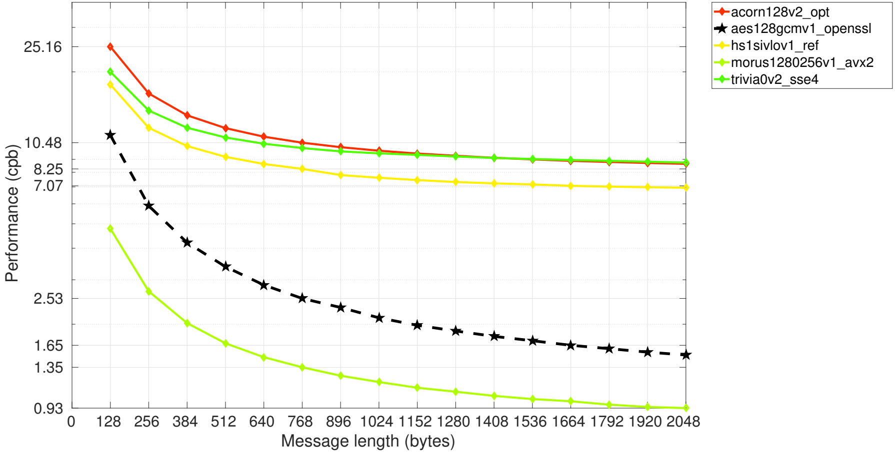
Comparison of all Block Cipher based schemes



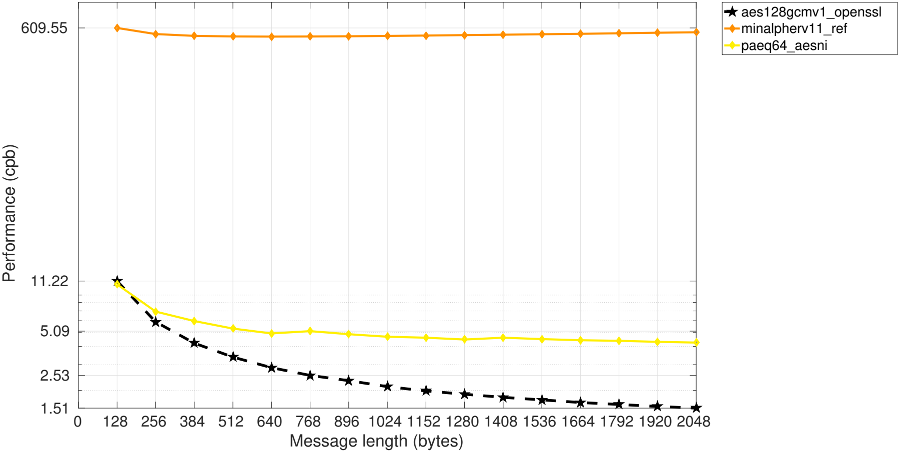
Comparison of all Sponge based schemes



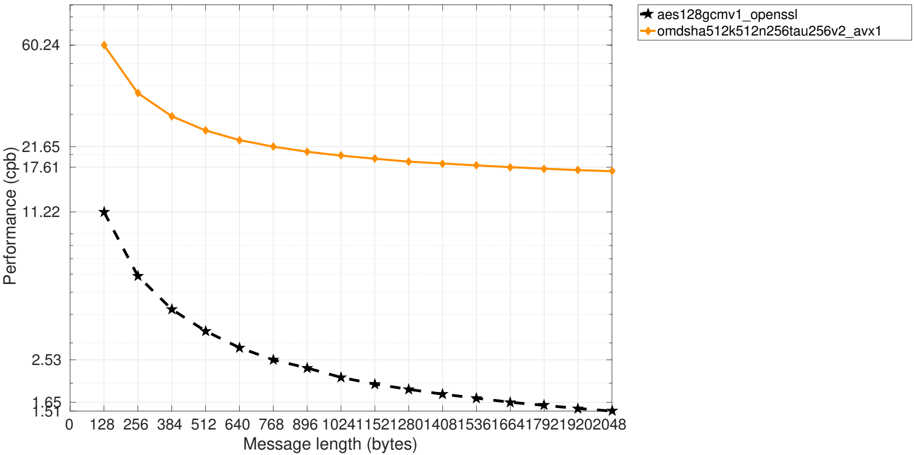
Comparison of all Stream Cipher based schemes



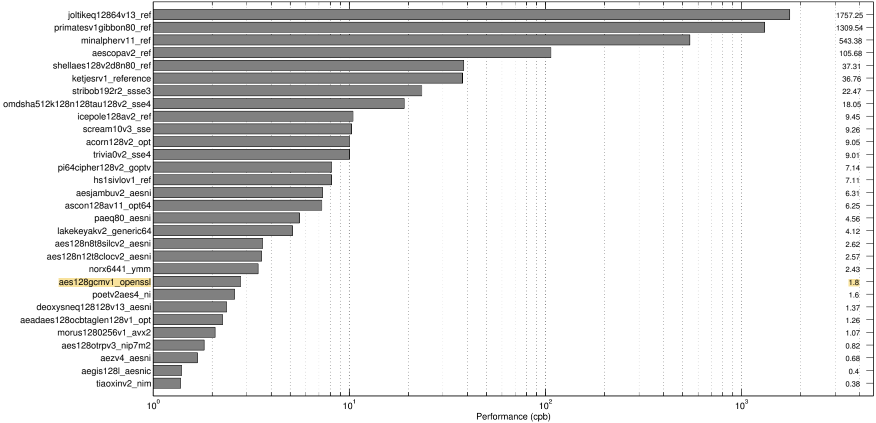
Comparison of all Permutation based schemes



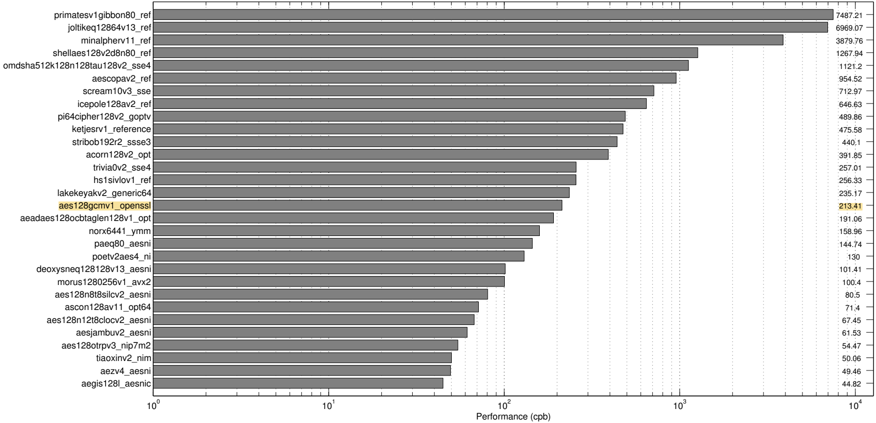
Comparison of all Compression Function based schemes



Comparison in the TLS setting



Comparison in the SSH setting



Currently fastest cipher (Software)

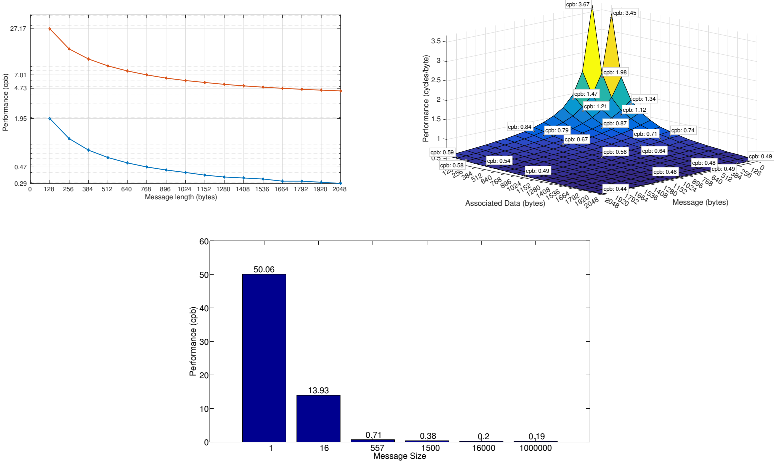


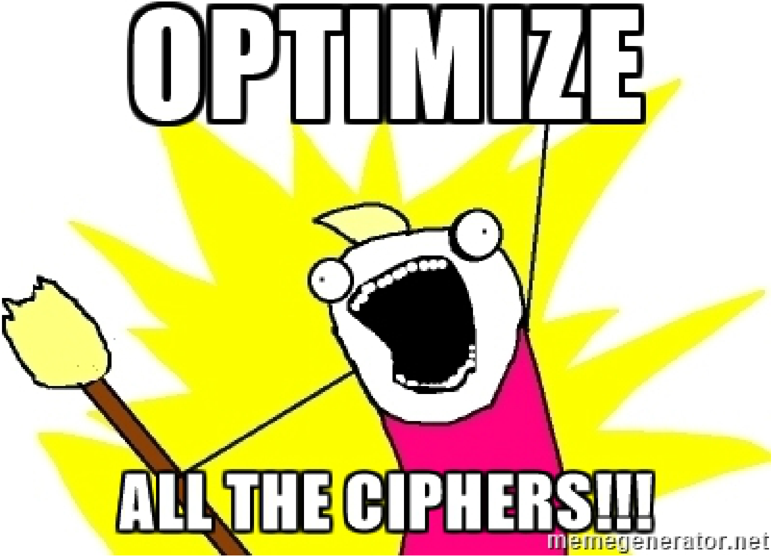
Figure: Tiaoxin v2.0 (SSE and AES-NI optimized)

Conclusions

Conclusions

- New framework to benchmark Authenticated Encryption ciphers
  - Very simple, only focus on AE ciphers
  - Timer Stamp Counter (with optimized `RDTSCP` instruction)
  - Reduction of noise during measurements
- Comparison of CAESAR 2^{nd} round Candidates
  - TLS setting
  - SSH setting
- 23 out of 30 ciphers offer at least one optimization

Further Work



Questions?

Thank you for your attention!

References I

Daniel J. Bernstein.
Supercop.
https://bench.cr.yp.to/supercop.html, 2016.

Shay Gueron.
Intel® advanced encryption standard (aes) new instructions set.
Technical report, Intel Corporation, 2012.

Ted Krovetz and Phillip Rogaway.
The Software Performance of Authenticated-Encryption Modes, pages 306–327.
Springer, 2011.

Gregory Lento.
Optimizing performance with intel® advanced vector extensions.
http://www.intel.com/content/www/us/en/benchmarks/performance-xeon-e5-v3-advanced-vector-extensions-paper.html, 2014.

Neon.

References II

Gabriele Paoloni.
How to benchmark code execution times on intel® ia-32 and ia-64 instruction set architectures.
Technical report, Intel Corporation, 2010.

Markku-Juhani Saarinen.
The BRUTUS automatic cryptanalytic framework.
Journal of Cryptographic Engineering, 6(1):75–82, 2016.