WOLFSSL · TECHNICAL BENCHMARK

wolfSSL vs MbedTLS

Intel、ARM（Cortex-A / Cortex-M）、RISC-Vを対象としたベンチマーク

対象

本資料では、wolfCryptが提供する暗号アルゴリズム群全体をMbedTLSと比較し、4種類のプラットフォームで同一条件のもと測定しました。

対象プラットフォームは、

- Intel x86_64
- Raspberry Pi 5（ARMv8-A Cortex-A76）
- ベアメタル STM32H563 Cortex-M33
- Microchip PolarFire SoC（RISC-V U54）

です。

さらに、MbedTLSでは提供されていない**耐量子暗号（Post-Quantum Cryptography）**や拡張アルゴリズムについても比較対象としています。

使用したライブラリは、

- wolfSSL v5.9.1
- MbedTLS 3.6.6

（2026年6月時点）です。

ベンチマーク方法
すべてのプラットフォームにおいて、両ライブラリをソースコードからビルドし、同一条件で測定しました。
wolfSSLは各プラットフォームで最高性能となるよう、

- CPUアーキテクチャ専用アセンブリ
- Single Precision（SP）演算

を有効にした構成を使用しています。
一方、MbedTLSは各環境で利用可能な標準（Stock）構成の中で最も高速な設定を使用しました。
デスクトップ環境では、

- wolfCrypt：wolfcrypt/benchmark
- MbedTLS：programs/test/benchmark

を利用しています。
MCU環境では、同一の暗号処理を実行し、オンチップのDWT（Data Watchpoint and Trace）サイクルカウンタで実行時間を測定しました。
測定条件は、

- ブロックサイズ：1024バイト
- 各アルゴリズムにつき約1秒間測定

です。

はじめに

wolfSSLとMbedTLSは、一見するとよく似たライブラリです。

どちらも組込み用途向けに設計された、小型で移植性の高いC言語製TLS／暗号ライブラリだからです。

しかし、実際に性能を測定すると、その差は非常に大きいことが分かります。

wolfCryptでは、x86、ARM、RISC-Vといった主要CPU向けに、性能が重要となるほぼすべての暗号処理について手書きアセンブリ実装を提供しています。

その結果、

セキュアブート
デバイスアテステーション
TLS通信

などで性能を左右する暗号処理は、MbedTLSの移植性重視のC実装に比べて何倍もの速度で実行できます。

さらにwolfSSLは性能だけでなく、対応アルゴリズムの範囲も大きく広がっています。

MbedTLSには搭載されていない

CNSA 2.0準拠の完全な耐量子暗号スイート
ML-KEM
ML-DSA
LMS
XMSS
SLH-DSA

までサポートしています。

以下で示す数値は、カタログ値ではありません。

4種類の実機プラットフォーム上で、両ライブラリをソースコードからビルドし、速度重視の最適化を適用したうえで、各アルゴリズムについて同一条件で測定した実測値です。

Intel i9-11950H (x86_64)

wolfCryptは

- --enable-intelasm
- --enable-sp=yes,asm

を指定してビルドしています。

これにより、

- AES-NI
- AVX2
- SHA拡張命令
- Single Precision x86_64アセンブリ

が利用されます。

MbedTLSは、

- AES-NI
- MBEDTLS_HAVE_ASM

を有効にした標準Release構成で測定しました。

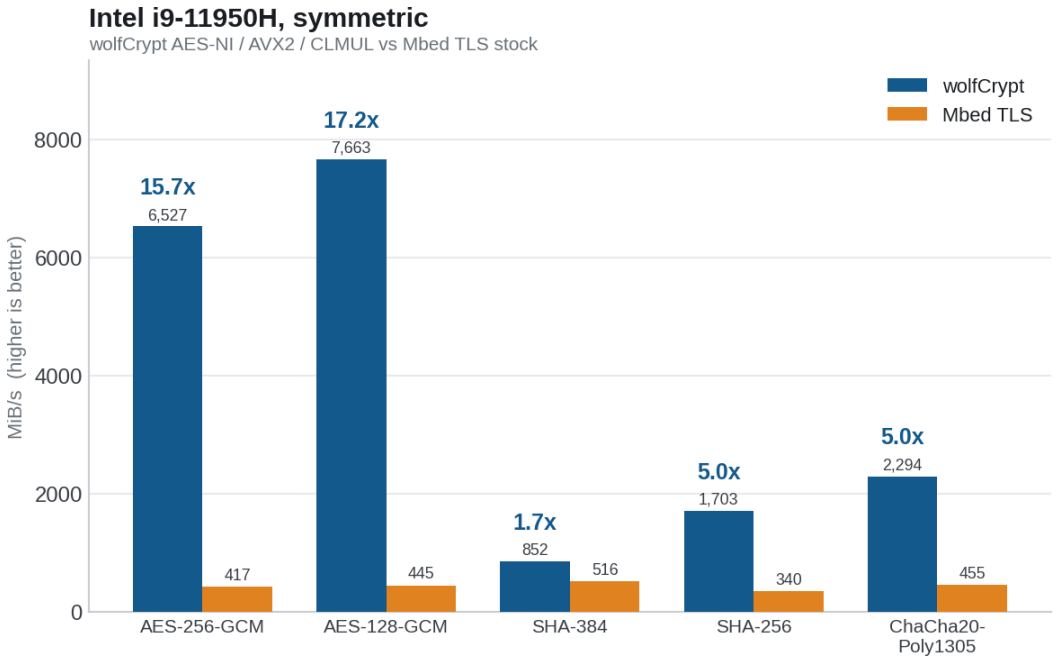


Figure 1. Intel i9-11950H, symmetric throughput (MiB/s).

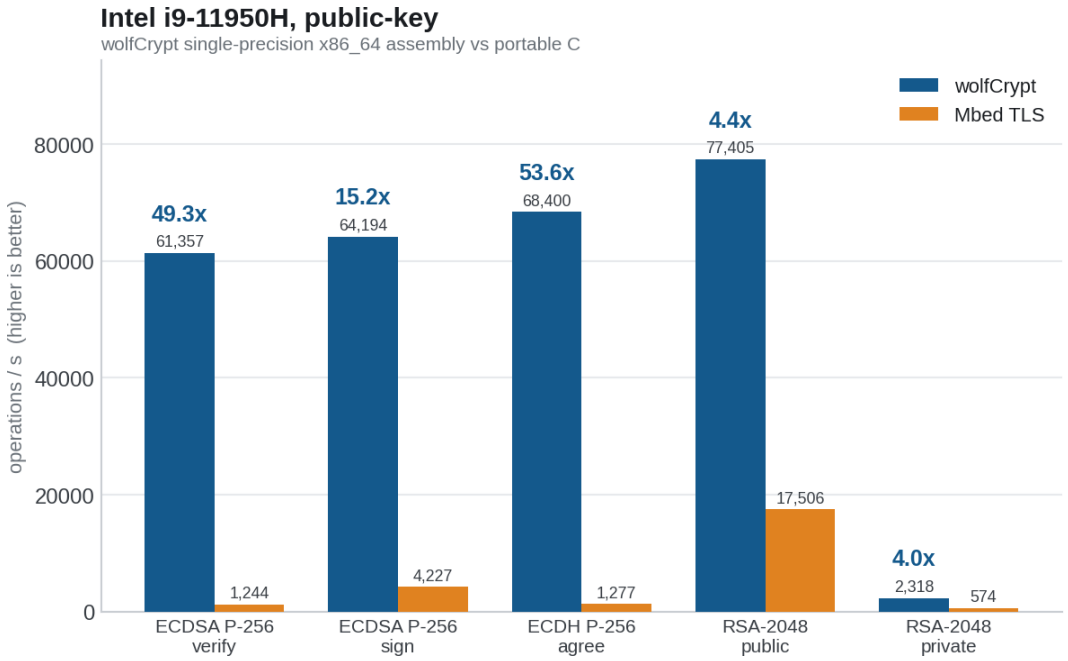


Figure 2. Intel i9-11950H, public-key operations per second.

Operation	wolfCrypt	MbedTLS	wolfSSL Advantage
AES-256-GCM	6527 MiB/s	417 MiB/s	**15.7x**
AES-128-GCM	7663 MiB/s	445 MiB/s	**17.2x**
SHA-384	852 MiB/s	516 MiB/s *	**1.7x**
SHA-256	1703 MiB/s	340 MiB/s	**5.0x**
ChaCha20-Poly1305	2294 MiB/s	455 MiB/s	**5.0x**
ECDSA P-256 sign	64,194/s	4,227/s	**15.2x**
ECDSA P-256 verify	61,357/s	1,244/s	**49.3x**
ECDH P-256 agree	68,400/s	1,277/s	**53.6x**
RSA-2048 public	77,405/s	17,506/s	**4.4x**

* MbedTLS's benchmark only measures SHA-512, not SHA-384; they share the same core and run at the same speed, so it is the fair SHA-384 comparison.

Intel環境では、最も大きな性能差が現れるのはAES-GCMと公開鍵暗号です。

AES-GCMでは、wolfCryptはAES-NIとCarry-less Multiply（CLMUL）による高速GHASH実装を使用しているのに対し、MbedTLSはテーブルベースのGCM実装であるため、**15〜17倍**の性能差が生じています。

公開鍵暗号でも、wolfCryptのSingle Precisionアセンブリ実装とMbedTLSの移植性重視のC実装との差により、ECDHでは**最大53倍**の性能差が確認されました。

ARM Raspberry Pi 5 (Cortex-A76 @ 2.4 GHz)

wolfCryptは

- `--enable-armasm`
- `--enable-sp=yes,asm`

を指定してビルドしています。

configure ログから、ARMv8 の暗号拡張命令が有効になっていることを確認しています。

AES-256-GCM は **1.34 サイクル／バイト**で動作しており、これはハードウェア AES と PMULL 命令を利用した結果です。

参考までに、ソフトウェアのみで AES を実装した場合は約 25 サイクル／バイト程度となります。

MbedTLS は同じ 3.6.6 のソースコードを Raspberry Pi 上でビルドして測定しています。

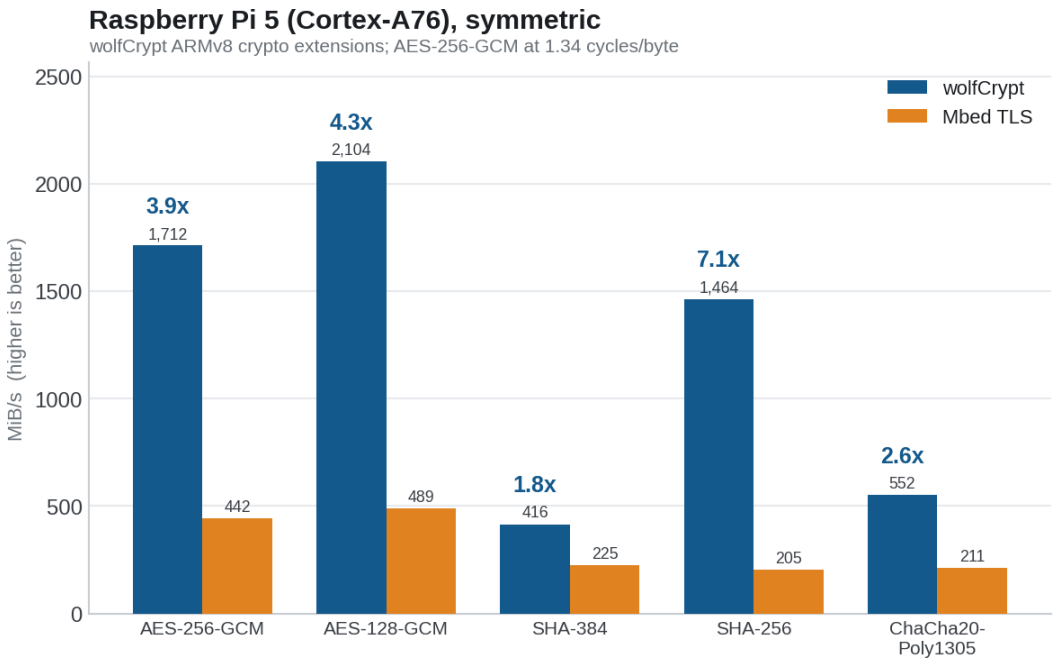


Figure 3. Raspberry Pi 5, symmetric throughput (MiB/s).

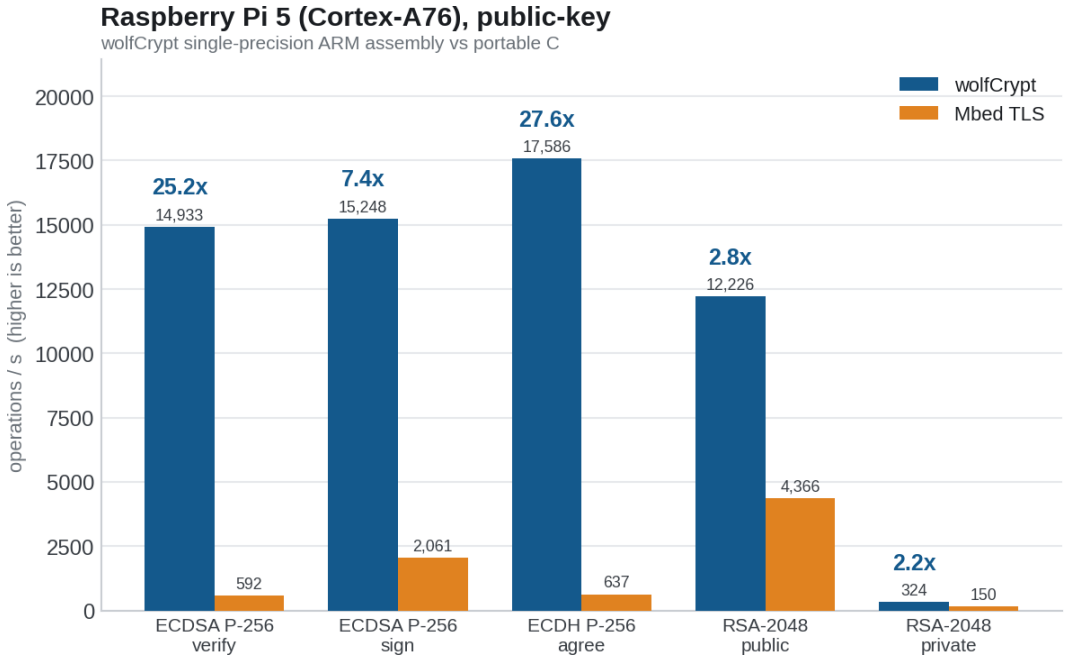


Figure 4. Raspberry Pi 5, public-key operations per second.

Operation	wolfCrypt	MbedTLS	wolfSSL Advantage
AES-256-GCM	1712 MiB/s	442 MiB/s	**3.9x**
AES-128-GCM	2104 MiB/s	489 MiB/s	**4.3x**
SHA-384	416 MiB/s	225 MiB/s *	**1.8x**
SHA-256	1464 MiB/s	205 MiB/s	**7.1x**
ECDSA P-256 verify	14,933/s	592/s	**25.2x**
ECDH P-256 agree	17,586/s	637/s	**27.6x**
RSA-2048 public	12,226/s	4,366/s	**2.8x**

ARM 環境では Intel ほど差は大きくありません。これは、Raspberry Pi 上の MbedTLS も ARMv8 の AES 命令を利用できるためです。

一方、x86 環境では、MbedTLS の GCM 実装はソフトウェア GHASH に依存するため、Intel 環境の方が性能差がより大きく表れています。

STM32H563 (Cortex-M33 @ 250 MHz, bare metal)

両ライブラリとも Cortex-M33 向けにクロスコンパイルし、実機上でオンチップの **DWT（Data Watchpoint and Trace）サイクルカウンタ**を用いて性能を測定しました。

wolfCrypt は、

- AES
- SHA-256
- ChaCha20
- Poly1305

については **Thumb-2 アセンブリ実装**を使用し、

さらに、

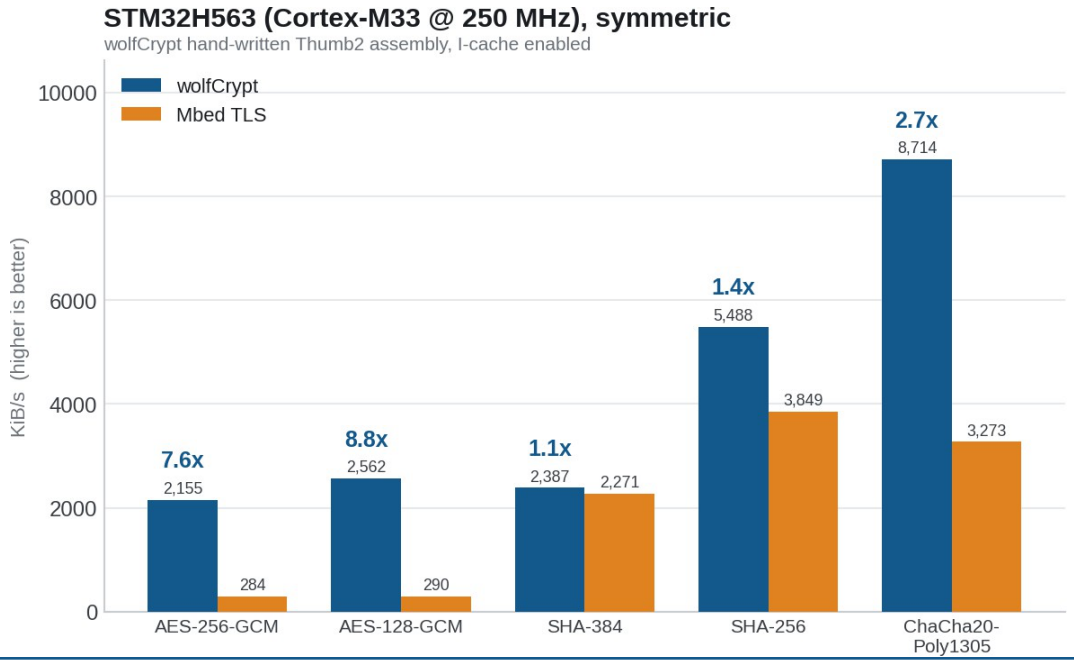
- ECC
- RSA

など公開鍵暗号については **SP（Single Precision）Cortex-M アセンブリ**を使用しています。

なお、この測定では **STM32 のハードウェア暗号アクセラレータは一切使用していません。**

Cortex-M33 コアには AES 命令や SHA 命令が存在しないため、両ライブラリともソフトウェア実装による比較となります。また、実際の STM32H5 ファームウェアと同様に**命令キャッシュ（Instruction Cache）を有効**にしています。命令キャッシュを無効にすると、フラッシュメモリは 5 ウェイトステートで動作するため、すべての測定結果がおよそ **3 倍遅く**なります。

Figure 5. STM32H563, symmetric throughput (KiB/s).



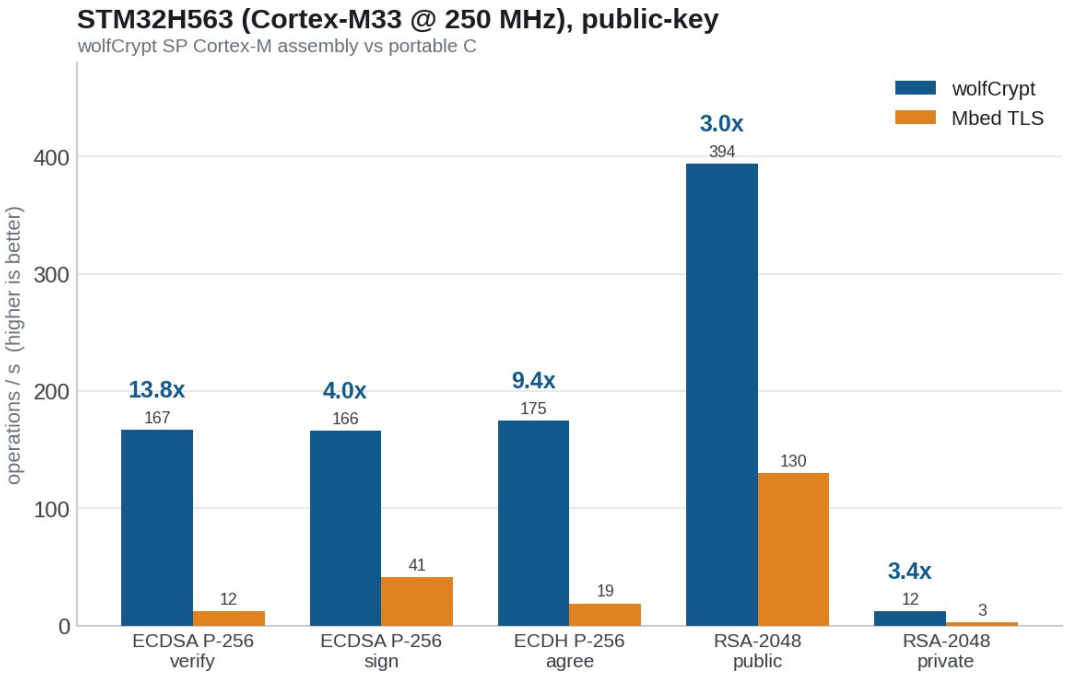


Figure 6. STM32H563, public-key operations per second.

Operation	wolfCrypt	MbedTLS	wolfSSL Advantage
AES-256-GCM	2155 KiB/s	284 KiB/s	**7.6x**
AES-128-GCM	2562 KiB/s	290 KiB/s	**8.8x**
AES-128-CBC	5175 KiB/s	2832 KiB/s	**1.8x**
ChaCha20-Poly1305	8714 KiB/s	3273 KiB/s	**2.7x**
SHA-256	5488 KiB/s	3849 KiB/s	**1.4x**
SHA-384 / SHA-512	2387 KiB/s *	2271 KiB/s	**1.05x**
ECDSA P-256 verify	167/s	12.1/s	**13.8x**
ECDH P-256 agree	175/s	18.6/s	**9.4x**
RSA-2048 public	394/s	130/s	**3.0x**

* SHA-384／SHA-512 には、この MCU 特有の注意点があります。

SHA-512（SHA-384 も同じコアを使用）は 64 ビット演算を行います。wolfCrypt の Thumb-2 による完全展開（Fully Unrolled）SHA-512 アセンブリは約 **12.6KB** あり、STM32H563 が搭載する **8KB の命令キャッシュ**より大きくなっています。そのため、フラッシュメモリ上から実行すると命令キャッシュが頻繁に入れ替わり（キャッシュスラッシング）、性能は **1858KiB/s** まで低下します。これは C 実装よりも遅い値です。

しかし、この同じアセンブリコードを **SRAM（RAMFUNCTION）へ配置すると、フラッシュのウェイトステートがなくなるため、性能は 2387KiB/s** となり、C 実装を上回ります。

この制約は STM32H563 のような小さな命令キャッシュを持つ MCU 特有のものです。

Intel や Raspberry Pi のように 48KB〜64KB 程度の L1 命令キャッシュを持つ CPU では、このアセンブリ実装がキャッシュ容量に制限されることはありません。そのため、それらの環境では SHA-512 の性能差がより大きく現れています。

本資料では、このアセンブリ本来の性能を示すため、**SRAM 配置時の測定値**を掲載しています。

十分なキャッシュ容量や TCM（Tightly Coupled Memory）を備えたプロセッサでは、特別な配置をしなくてもアセンブリ実装が最も高速になります。

STM32での結果

STM32H563では、wolfCryptは**すべてのアルゴリズムでMbedTLSを上回る性能**を示しました。

対称暗号では、

- AES-CBC：約 1.8 倍
- AES-GCM：約 7〜9 倍
- ChaCha20-Poly1305：約 2.7 倍

高速です。

これらはいずれも、命令キャッシュ内に収まるよう最適化された Thumb-2 アセンブリ実装によるものです。

さらに、セキュアブートやデバイスアテステーションで重要となる公開鍵暗号では、

- ECDSA Verify：約 13.8 倍
- ECDH：約 9.4 倍

という大きな性能差が確認されました。

これは wolfCrypt の SP Cortex-M アセンブリ実装による効果です。

RISC-V Microchip PolarFire SoC (SiFive U54-MC @ 600 MHz)

wolfCrypt は U54 向けに性能重視で最適化された構成でビルドしています。

使用した構成は、

- `--enable-riscv-asm`

を有効にし、

以下について **RV64 手書きアセンブリ**を利用しています。

- AES
- SHA-2
- SHA-3
- ChaCha20
- Poly1305

さらに、RSA および ECC には Single Precision 演算を使用しています。

この構成でもライブラリ全体のコードサイズは約 **712KB** です。

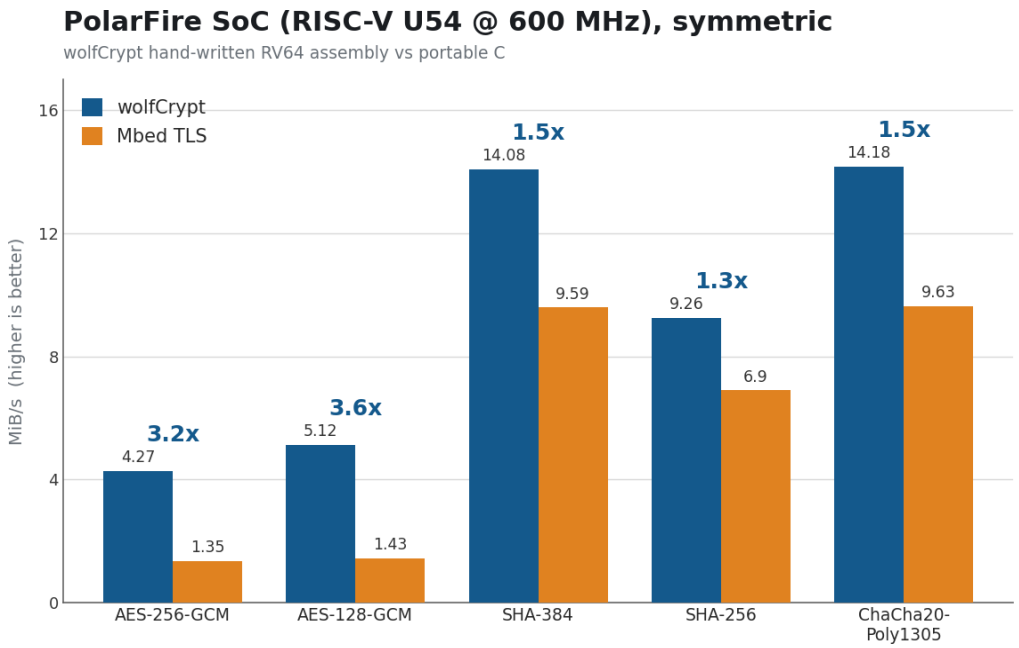


Figure 8. PolarFire SoC U54, symmetric throughput (MiB/s).

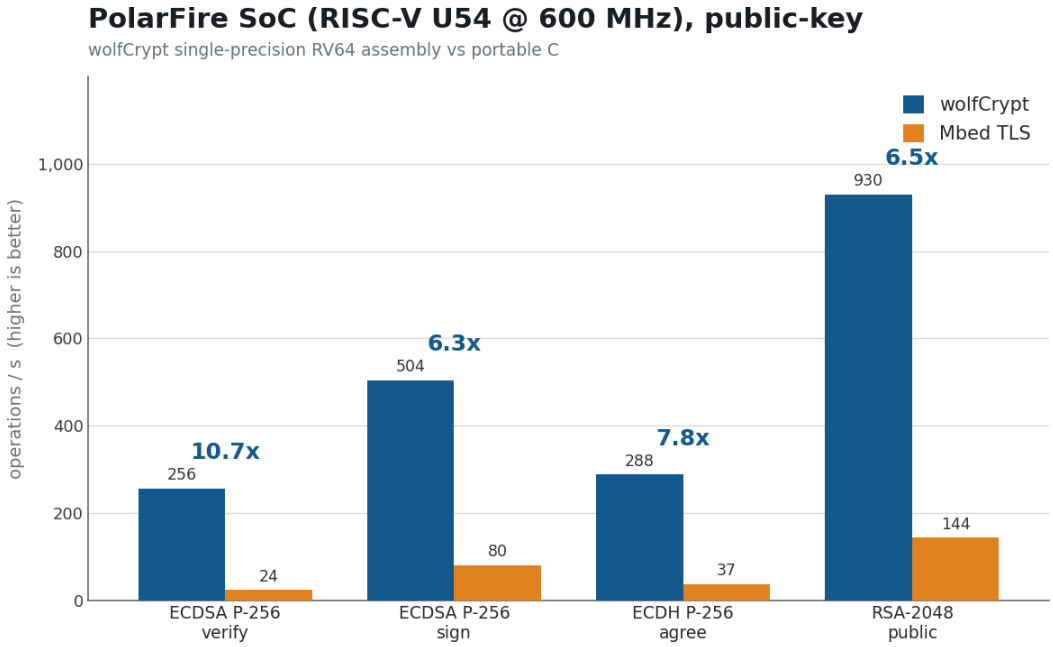


Figure 9. PolarFire SoC U54, public-key operations per second.

Operation	wolfCrypt	MbedTLS	wolfSSL Advantage
AES-256-GCM	4.3 MiB/s	1.3 MiB/s	**3.2x**
AES-128-GCM	5.1 MiB/s	1.4 MiB/s	**3.6x**
ChaCha20-Poly1305	14.2 MiB/s	9.6 MiB/s	**1.5x**
SHA-384	14.1 MiB/s	9.6 MiB/s *	**1.5x**
SHA-256	9.3 MiB/s	6.9 MiB/s	**1.3x**
ECDSA P-256 verify	256/s	24/s	**10.7x**
ECDH P-256 agree	288/s	37/s	**7.8x**
RSA-2048 public	930/s	144/s	**6.5x**

* MbedTLS は SHA-384 ではなく SHA-512 のみをベンチマーク対象としています。しかし両者は同じ内部実装（コア）を共有しており、実行速度も同一であるため、本比較は公平なものです。U54 には暗号命令が搭載されていないため、この環境では両ライブラリともソフトウェア実装のみで動作しています。

wolfCrypt は、すべての暗号処理において MbedTLS を上回る性能を示しました。

最も大きな差が見られるのは公開鍵暗号であり、

- RSA-2048 公開鍵演算：**6.5 倍**
- ECDH：**7.8 倍**
- ECDSA 署名検証：**10.7 倍**

の性能向上を実現しています。

この差は、wolfCrypt が採用する **Single Precision（SP）多倍長演算**が、MbedTLS の移植性を重視した多倍長整数（bignum）実装より高速であることによるものです。

対称暗号についても、AES-128-GCM で最大 **3.6 倍**の性能差があり、これは wolfCrypt の **RISC-V 向け手書きアセンブリ実装**と、高速な GHASH 実装によるものです。

さらに、RISC-V アセンブリを有効にするだけでも、wolfCrypt の移植性重視 C 実装と比べてソフトウェア暗号処理の性能は大きく向上します。

- ChaCha20：**33%高速**
- Poly1305：**25%高速**
- SHA-256：**24%高速**

また、テーブルを使用しない定数時間（constant-time）の AES 実装では、スループットが **0.12 MiB/s から 6.4 MiB/s へ**大幅に向上しています。

wolfSSLがサポートするMbedTLSにはない機能

速度だけでなく、速度重視に最適化された同じビルド構成で、wolfCrypt は MbedTLS には存在しない数多くの暗号アルゴリズムも提供しています。

特に重要なのは、**完全な耐量子暗号（Post-Quantum Cryptography：PQC）スイート**を備えていることです。

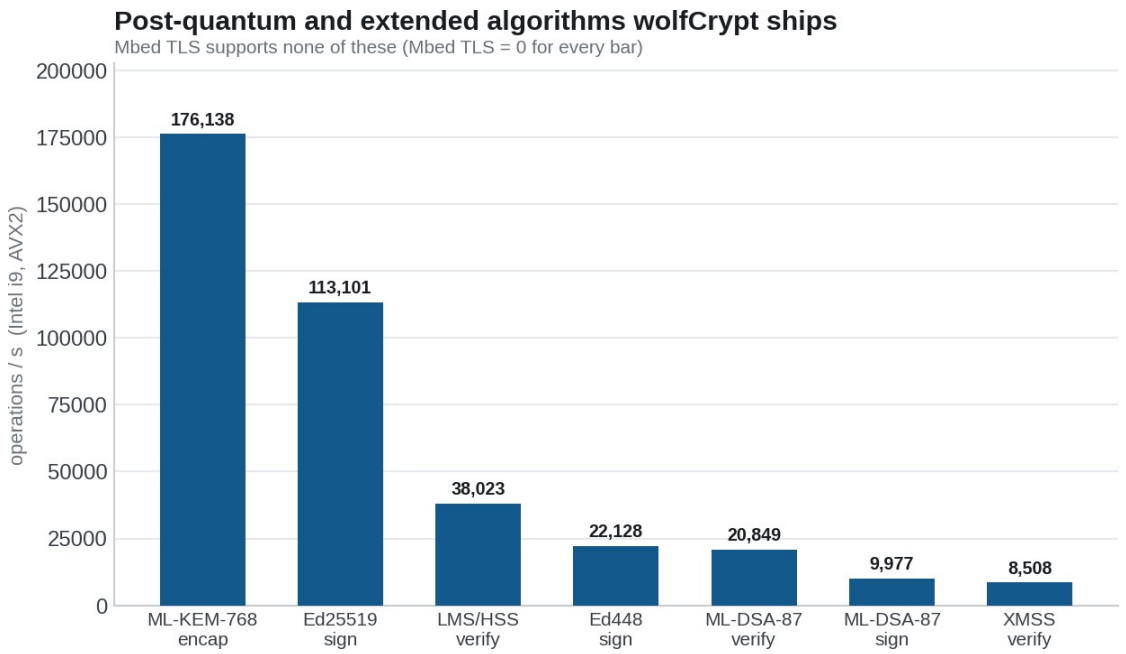


Figure 7. Post-quantum and extended algorithms wolfCrypt ships, ES256-class throughput on Intel i9. MbedTLS supports none of these.

Category	wolfSSL	MbedTLS
Post-quantum signatures	ML-DSA-44/65/87 (FIPS 204; ML-DSA-87 verify 20,849/s), LMS / HSS, XMSS / XMSS-MT, SLH-DSA (FIPS 205)	None
Post-quantum KEM	ML-KEM-512/768/1024 (FIPS 203; ML-KEM-768 encap 176,138/s)	None
EdDSA	Ed25519 (sign 113,101/s), Ed448	None
Identity-based (MIKEY-SAKKE)	ECCSI, SAKKE	None
Extra symmetric / hash / KDF	AES-SIV, AES-GCM-STREAM, AES-OFB, SHAKE128/256, RIPEMD-160, BLAKE2b / 2s, SipHash, scrypt, SRTP / SRTCP KDF	None

wolfPSAによるPSA Crypto API互換

wolfCrypt は **wolfPSA** を通じて、Arm PSA Crypto API を実装しています。

そのため、これまで紹介した性能やアルゴリズム群を、**PSA ベースのアプリケーション設計を変更することなく**利用できます。

例えば、TF-M（Trusted Firmware-M）などで Mbed TLS の PSA Crypto API を利用しているシステムでは、アプリケーションが現在呼び出している

```
psa_*
```

API をそのまま維持したまま、バックエンドを wolfPSA へ置き換えることができます。

つまり、

- PSA アーキテクチャはそのまま維持
- wolfCrypt の高性能暗号エンジンを利用
- FIPS 140-3 対応への道筋を確保
- 耐量子暗号スイートを利用可能

という構成を実現できます。

TF-M や PSA を採用した既存アプリケーションにも、そのまま組み込むことが可能です。

アプリケーションの書き換えは不要

実際には、アプリケーションはこれまで通り標準の PSA Crypto API を呼び出し続けます。

wolfPSA が各 API 呼び出しを wolfCrypt へ振り分けるため、

- 高速な AES-GCM
- Single Precision による高速公開鍵演算
- 耐量子暗号アルゴリズム

といった wolfCrypt の機能を、アプリケーションを書き換えることなく利用できます。

速度だけではないwolfSSLとMbedTLSの違い

性能と対応アルゴリズムの豊富さは、wolfSSL の特長の一部に過ぎません。

wolfSSL は、**オリジナル開発チーム自身によって継続的に開発・保守されている商用サポート付きの暗号ライブラリ**です。

幅広いプラットフォームへの対応、高度な機能、各種認証取得を備えています。

一方、MbedTLS は、**XySSL / PolarSSL を起源とするコミュニティ主体のオープンソースプロジェクト**であり、認証取得やサポート体制、機能の広さという点では wolfSSL とは異なります。

以下では、その違いを比較します。

Technology

Capability	wolfSSL	MbedTLS
Ownership	wolfSSL Inc. (single owner)	Multiple owners; forked from XySSL / PolarSSL
Development team	Original developers still on the project	Not maintained by the original developers
移植性	Portable out of the box: Windows (Win32/64), Linux, macOS, Solaris, *BSD (FreeBSD, NetBSD, OpenBSD), QNX, embedded Linux (Yocto, Buildroot, OpenWRT), iOS, Android, and most RTOSes (FreeRTOS, SafeRTOS, Zephyr, ThreadX / Azure RTOS, VxWorks, Micrium uC/OS-II/III, Nucleus, Green Hills INTEGRITY, SEGGER embOS, Keil RTX / CMSIS-RTOS, TI-RTOS, NXP MQX, Apache Mynewt, NuttX, RIOT, ChibiOS, DDC-I DEOS, TRON / ITRON / uITRON, uT-Kernel), plus bare metal / no-OS, Linux kernel module, Arduino, ARM Mbed OS, Contiki-NG, TenAsys INtime, EBSnet, Nintendo / game consoles (devkitPro), Haiku, HP-UX, NonStop, MontaVista, TinyOS, and AIX	Fewer targets out of the box: Win32/64, Linux, macOS, *BSD, OpenWRT, iOS, Android, SEGGER embOS, Xbox
標準	SSL 3.0 to TLS 1.3; DTLS 1.0 / 1.2 / 1.3	TLS 1.2 / 1.3; DTLS 1.2

アセンブラー、CPUアクセラレータ	ARM AArch32 / AArch64 and Cortex-M Thumb-2 assembly, NEON, and ARMv8-A crypto	AES-NI/CLMUL + Armv8 AES/SHA + bignum asm
Capability	**wolfSSL**	**MbedTLS**
	extensions (AES, SHA-1 / SHA-256, SHA-512, SHA-3, SM3 / SM4) plus ARM ML-KEM post-quantum assembly; Intel and AMD x86-64 AES-NI, VAES, PCLMULQDQ (GHASH), AVX1 / AVX2, BMI2 / ADX, RDRAND / RDSEED and Intel QuickAssist (QAT), with AVX2 assembly for ChaCha20 / Poly1305 / X25519; RISC-V RV64 with scalar (Zk) and vector (Zvk) crypto extensions; PowerPC assembly; and single-precision (SP) math in C and assembly for RSA / ECC / DH	
ハードウェアアクセラレータ、セキュアエレメント	STMicroelectronics STM32 (F1/F2/F4/F7, L1/L4/L5/U5, H5/H7, WB, WL, MP13/MP25) PKA / CRYP / HASH and STSAFE-A110; NXP CAAM (i.MX 6/7/8, RT1170), DCP, LTC, Kinetis mmCAU, Coldfire SEC, CASPER and EdgeLock SE050; Microchip PIC32 MX / MZ, ATECC508A / 608A / 608B and TA100; Espressif ESP32 / S2 / S3 / C3 / C6 (AES / SHA / RSA); Renesas RX65N / RX72N (TSIP), RA6M4 / RZ (SCE) and Synergy; Silicon Labs Series 2 / EFR32 (SE, CRYPTO, TRNG); Nordic nRF52840 and nRF51 with ARM CryptoCell-310; Analog Devices / Maxim MAX3266x, MAXQ1065 / MAXQ1080; Cypress / Infineon PSoC 6; Texas Instruments TM4C (Tiva); Xilinx / AMD Zynq UltraScale+ and Versal; Cavium / Marvell Nitrox (III / V) and OcteonTX; Tropic Square TROPIC01; TPM 2.0 (wolfTPM); IoT-SAFE; PKCS#11 HSMs; Linux /dev/crypto, AF_ALG and KCAPI; and the Arm PSA Crypto API for TF-M, NXP SECO (i.MX 8 Security Controller), Raspberry Pi Pico (RP2040), NVIDIA CUDA (GPU AES), AUTOSAR crypto driver (CSM / CryIf), and Renesas RSIP	STMicroelectronics STM32 (CRYP / HASH / PKA); NXP (CAAM, ELS / SSS); Espressif ESP32 / S2 / S3 / C3 / C6 (AES / SHA / RSA); Nordic nRF (Arm CryptoCell CC310 / CC312); Silicon Labs EFR32 Series 1 / 2 (SE / CRYPTO); Renesas RA (SCE); Cypress / Infineon PSoC 6; Microchip ATECC608A and NXP SE050 (PSA secure-element drivers); Arm PSA Crypto API and TF-M

コマンドラインユーティリティ (wolfCLU)	Yes	No
Capability	**wolfSSL**	**MbedTLS**
第三者認証	FIPS 140-3 (#4718, #5041); DO-178C DAL-A; ISO 26262 ASIL D	No
証明書の取り消し	CRL, OCSP, OCSP stapling	CRL
暗号化抽象レイヤー	Yes	No
TLS inspection (sniffer)	Yes	No
圧縮	zlib	No
OpenSSL 互換API	Yes, 1,600+ functions, actively maintained	Limited, out of date
耐量子暗号	ML-KEM (FIPS 203), ML-DSA (FIPS 204), SLH-DSA (FIPS 205), LMS/HSS, XMSS/XMSS-MT, Falcon FN-DSA	No
サポート対象オープンソースプロジェクト	OpenSSH, stunnel, wpa_supplicant, lighttpd, cURL, OpenVPN, NGINX, mongoose, and 900 others	Limited
品質保証	API tests, peer review, static analysis, many compilers and platforms, hardware-accelerated testing, fuzzing (internal fuzzer, AFL, libFuzzer, TLS Fuzzer), big / little endian, known user configs, external validation	Compilation tests, code coverage, regressions, test vectors, interop, build configs, memory checks, fuzzing, static analysis
AI	Internally developed Skoll and Fenrir advanced AI static-analysis scanning tools	None

Support, documentation, and licensing

Capability	wolfSSL	MbedTLS
ドキュメント	Complete: full manual, API reference, build instructions, extensions reference, tutorials, examples, benchmarking	Partial: build instructions, API reference, source
脆弱性対応	Fixes typically available within days	Fixes can take months, or not at all
ライセンス	Dual GPLv3 / commercial	Dual GPLv2 / Apache 2.0
Royalty-free	**Yes**	**Yes**
商用サポート	Yes, from the original developers: email, phone, and forums; tiers from presale to 24/7; response within 3 business days	Community forums only

まとめ

同一条件でビルドし、同一ハードウェア上で測定した結果、**wolfCrypt はすべての評価対象プラットフォームで MbedTLS を上回る性能**を示しました。

公開鍵暗号では、

- x86 環境で **15〜53 倍**
- Raspberry Pi 5 で **3〜28 倍**
- STM32H563 で**最大 14 倍**
- PolarFire RISC-V で **6〜11 倍**

高速でした。

特に、wolfCrypt が手書きアセンブリを提供しているアルゴリズムでは、その差が顕著に表れています。

さらに wolfCrypt は、

- ML-KEM
- ML-DSA
- SLH-DSA
- LMS
- XMSS

から成る**完全な耐量子暗号スイート**に加え、

EdDSA や多数の拡張アルゴリズムも提供しており、MbedTLS では利用できない機能を数多く備えています。

性能やアルゴリズムだけでなく、

- **FIPS 140-3 認証（#4718、#5041）**
- **DO-178C DAL-A 対応**
- **ISO 26262 ASIL D 対応**
- **MISRA C 準拠チェック**
- **TF-M 向け Arm PSA Crypto API サポート**

など、セキュリティ・安全性が求められる分野に必要な要件にも対応しています。

さらに、これらは **wolfSSL オリジナル開発チームによる 24 時間 365 日の商用サポート**によって支えられています。

Versions and platforms

Libraries: wolfSSL v5.9.1 (dual GPLv3 / commercial), MbedTLS 3.6.6. Tested June 2026.

Builds: wolfSSL tuned for maximum speed on each target. Intel `--enable-intelasm --enable-sp=yes,asm`; Raspberry Pi `--enable-armasm --enable-sp=yes,asm`; STM32 Thumb2 plus SP Cortex-M assembly. MbedTLS in its fastest stock configuration (`MBEDTLS_HAVE_ASM`). Identical sources built from source on every target.

Platforms: Intel i9-11950H, x86_64, GCC 12.2. Raspberry Pi 5, Cortex-A76 at 2.4 GHz, Debian 12, GCC 12.2. STM32H563 (NUCLEO-H563ZI), Cortex-M33 at 250 MHz, bare metal, arm-none-eabi-gcc 13.2, instruction cache enabled. PolarFire SoC Video Kit (MPFS250T), 4x SiFive U54-MC at 600 MHz, Yocto Linux 6.12, GCC 13.3.

Measurement: `wolfcrypt/benchmark` vs `programs/test/benchmark` on the desktop and server targets; on the MCU, the same operations timed with the on-chip DWT cycle counter. Block size 1024 B, about 1 s per algorithm. Symmetric in MiB/s or KiB/s, public-key in operations per second.