XTS モードの実装性能調査

株式会社レピダム
2020 年 2 月

本評価結果の概要（エグゼクティブサマリー）

本調査では、ブロック暗号利用モードのひとつである XTS の実装性能評価を行い、その結果を報告する。現在、XTS は CRYPTREC 暗号リストへの追加候補として検討されているブロック暗号利用モードであり、2019 年に安全性の観点で峯松氏により評価 [1]が行われた。その結果、XTS は現実的に安全であると結論づけられ、CRYPTREC においても CRYPTREC 暗号リストに追加するための要件を満たしていると判断された。本報告書は、現実的に安全であると結論づけられた XTS に対して、電子政府システム等で利用する用途において十分な実装性能を有するか判断するための判断根拠として調査を実施した。本調査は大別して「XTS の市販品などでの実装採用状況」および「XTS の実装性能」の 2 つの観点から調査を実施した。

調査結果として、「XTS の市販品などでの実装採用状況」については、製品や OSS において XTS が広く採用されていることがわかった。代表的な製品としては、Microsoft 社の Windows においてディスク暗号化で用いられる BitLocker や Apple 社の macOS においてディスク暗号化で用いられる FileVault 2 などのストレージ暗号化製品が挙げられる。また、OSS においても、世界中で広く利用されている暗号ライブラリである OpenSSL や組み込み機器で利用される WolfCrypt や mbed TLS などでサポートされていることがわかった。

また、もう 1 つの調査観点である「XTS の実装性能」について、検索エンジンによる XTS の実装性能に関する公開情報による評価結果や、Windows 環境、macOS 環境および Linux 環境における OpenSSL および WolfCrypt を用いた実環境での実装性能測定を実施した。検索エンジンによる XTS の実装性能に関する公開情報に関する調査結果としては、2010 年など比較的古い実装性能に関する情報が公開されている。最新の実装性能としてはベンチマーク製品において拡張現実や機械学習といった高負荷な処理の代表である最先端テクノロジーと同様に、ベンチマークにおける性能測定項目として AES-XTS が含まれたスコアリングが行われていた。また、Windows 環境、macOS 環境および Linux 環境における OpenSSL および WolfCrypt を用いた実環境での実装性能を計測した結果として、CRYPTREC 暗号リストで採用されているブロック暗号利用モードと実装性能を比較した結果、実装性能において大きな遜色はないものと判断した。

目次

- Arm または mbed は、Arm Ltd.の登録商標です
- Windows® の正式名称は Microsoft® Windows® Operating System です。
- Microsoft 、Windows 、Windows 10 、BitLocker 、Visual Studio は 、米 国 Microsoft Corporation.の米国およびその他の国における登録商標です。
- Linux は、Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
- IEEE は、The Institute of Electrical and Electronics Engineers, Inc.の登録商標または商標です。
- Apple、FileVault、Mac、macOS は、米国およびその他の国における Apple Inc. の登録商標です。
- OpenSSL は, 米国およびその他の国における米国 OpenSSL Software Foundation の登録商標です。
- RSA および BSAFE は、米国 EMC コーポレーションの米国およびその他の国における商標または登録商標です。
- TCG は、Trusted Computing Group の米国およびその他の国における商標または登録商標です。
- Intel、Intel Optane はアメリカ合衆国およびその他の国におけるインテルコーポレーションまたはその子会社の商標または登録商標です。
- GitHub は、GitHub Inc.の商標または登録商標です。
- Ubuntu は、Canonical Ltd.の商標または登録商標です。
- TrueCrypt と TrueCrypt のロゴは TrueCrypt Foundation の商標です。

1　はじめに

本報告書は、ブロック暗号利用モードのひとつである XTS の実装性能評価を行い、その結果を報告する。XTS は現在 CRYPTREC 暗号リストへの追加候補として検討されている暗号利用モードであり、2019 年に安全性の観点で峯松氏により評価が行われた [1]。評価では、XTS は現実的に安全であると結論づけられ、CRYPTREC においても CRYPTREC 暗号リストに追加するための要件を満たしていると判断された。本報告書は、安全性評価に続いて、XTS を実装性能の観点から評価を行い、CRYPTREC 暗号リストに追加される要件を満たしているかどうかの判断根拠を提供することを目的とする。

XTS は、2004 年に Rogaway によって提案された XEX モード [2]をベースとした方式である。XTS という名前は、NIST では XEX Tweakable Block Cipher with Ciphertext Sealing の略とされ、IEEE では XEX encryption mode with Tweak and ciphertext Stealing の略であるとされている。XTS は、2007 年に IEEE Storage in Security Work Group (SISWG)により、ハードディスクなどのストレージデバイスで利用することを目的として標準仕様が制定され、IEEE 1619-2007 として発行されている [3]。使用するブロック暗号を AES として仕様が定められており、この場合の利用モードは XTS-AES と呼ばれている。IEEE での標準化に続いて、NIST は IEEE 1619-2007 の標準方式に対し、暗号化の対象となる平文の長さ（データユニットと呼ばれる）の制限を要求事項として加えることで、NIST 推奨方式として承認した。標準化文書として NIST SP800-38E [4]が発行されている。その後、2018 年には、IEEE SISWG によって IEEE 1619-2007 が改訂され、IEEE 1619-2018 として発行された [5]。IEEE 1619-2018 では、IEEE 1619-2007 では推奨項目となっていたデータユニットサイズの制限が、NIST SP800-38E と同様に守るべき要求事項として示されている。

XTS は標準仕様だけでなく、製品や OSS での採用実績もある。代表的な製品としては、Microsoft 社の Windows の BitLocker や Apple 社の macOS の FileVault 2 などのストレージ暗号化製品が挙げられる。OSS においても、例えば、広く利用されている暗号ライブラリである OpenSSL や、WolfCrypt、mbed TLS などでサポートされている。

以上の標準化動向や実装採用状況に鑑み、CRYPTREC では電子政府システムなどでの利用が見込まれると判断される暗号技術として XTS の評価が進められている。2019 年には、外部評価者として峯松氏により XTS の安全性評価が実施され、XTS-AES は現実的に安全であるという結論となっている [1]。この結果を受けて、CRYPTREC 暗号技術検討会では、利用条件を満たす範囲において、暗号利用モード（秘匿モード）として CRYPTREC 暗号リストへ追加するための安全性要件を満たしていると判断されている [6]。本報告書は、安全性評価に続き、XTS の実装性能の観点で評価を行い、その結果を報告する。

本報告書の構成について述べる。まず 2 章では、XTS の歴史的背景を説明し、IEEE 1619-2018 を参考に仕様を説明する。また、実装された XTS の客観的な評価のための試験仕様である XTSVS について説明する。3 章では、XTS の普及状況として、製品での採用実績状況を調査した結果を示す。4 章では、学術論文や製品提供を行う企業が公開するホワイトペーパーを調査することで、XTS の客観的な実装性能評価結果を示す。5 章で

は、実装性能調査対象の暗号利用モードである XTS と CRYPTREC 暗号リストで採択されている暗号利用モードとの実装性能評価を実施することで、現在採択されている暗号利用モードと実装性能の比較結果を示す。6 章では、XTS 本来の利用方法に注目し、ディスク暗号化としての実装性能調査を実施する。最後に 7 章でまとめる。

2　XTS の技術的解説

本章では、XTS の技術的解説として、まず、XTS の歴史的背景について述べた後、2018 年に発行された IEEE 1619 [5]を参考に XTS の仕様について説明する。加えて、実装された XTS のアルゴリズム的な正しさや、安全性を確認するための仕組みの一つとして、NIST から提供されている XTSVS（The XTS-AES Validation system） [7]と呼ばれる検証システムについて説明する。

2.1　歴史的背景

本節では、XTS の歴史的背景として、IEEE や NIST での標準化動向や普及状況を示す。また国内の活動として、XTS に関連した CRYPTREC の動向を示す。

XTS はブロック暗号を利用した暗号利用モードである。方式としては、2004 年に Rogaway によって提案された XEX モード [2]をベースとしており、鍵の扱いや平文がブロックの長さの倍数でない場合の処理（Ciphertext Stealing と呼ばれる）を加えたものとなっている。
XTS は、2007 年に IEEE Storage in Security Work Group (SISWG)により、ハードディスクなどのストレージデバイスで利用することを目的として標準仕様が設計された。標準化文書 IEEE 1619-2007 として制定されている [3]。使用するブロック暗号を AES として仕様が定められており、この場合の利用モードは XTS-AES と呼ばれている。IEEE での標準化に続き、NIST は IEEE 1619-2007 の標準方式に対し、暗号化の対象となる平文の長さ（データユニットと呼ばれる）の制限を要求事項として加えることで、NIST 推奨方式として承認した。この方式に対し、標準化文書として NIST SP800-38E [4]が 2010 年 1 月に発行されている。また、同年 3 月には、実装された XTS が NIST SP800-38E に準拠しているかどうかを第三者機関で検証できるように、XTSVS [7]と呼ばれる試験仕様が NIST から提供されている。その後、2018 年には、IEEE SISWG により、IEEE 1619-2007 を改訂した標準仕様 IEEE 1619-2018 が発行された [5]。改訂された点として、データユニットサイズの制限の取り扱い方の変更が挙げられる。IEEE 1619-2007 では、データユニットサイズの制限について示されてはいたものの、推奨扱いであった。これに対し、IEEE 1619-2018 は、NIST SP800-38E と同様に、守るべき要求事項としてデータユニットサイズの制限を示している*。

また、XTS は実際の利用も広がっている。例えば、広く利用されている暗号ライブラリである OpenSSL では 2012 年に搭載され、その他、WolfCrypt や mbed TLS、BSAFE など、いくつかの暗号ライブラリでサポートされている。製品では、例えば、Windows 10 の BitLocker や macOS の FileVault 2 など、多くのストレージ暗号化製品で採用されている。

---

* IEEE 1619-2007 では、データユニットサイズの制限について、「The number of 128-bit blocks **should not** exceed 2^{20}」と記述されていたが、IEEE 1619-2018 では、「The number of 128-b blocks within the data unit **shall not** exceed 2^{20}.」と要求の程度が強まっていると見受けられる。

以上の標準化動向や実装採用実績が増加している現状に鑑み、CRYPTREC では電子政府システムなどでの利用が見込まれると判断される暗号技術として XTS の評価が進められている。2019 年には、外部評価者として峯松氏により XTS の安全性評価が実施され、その結果が報告書として発行された [1]。この報告書では、2011 年に発行された CRYPTREC レポート [8]での XTS の安全性評価結果を含め、IEEE および NIST の標準化文書および関連文献が精査されたうえで XTS の安全性が評価されており、XTS-AES は現実的に安全であるという結論となっている。この結果を受けて、CRYPTREC 暗号技術検討会では、利用条件を満たす範囲において、XTS は暗号利用モード（秘匿モード）として CRYPTREC 暗号リストへ追加するための安全性要件を満たしていると判断されている [6]。なお、利用条件は以下の 3 つである。

（条件 1）　利用用途は IEEE および NIST SP-800-38E の規格に沿ったストレージやディスクの暗号化に限る。
（条件 2）　XTS 内のブロック暗号には、CRYPTREC 暗号リスト掲載の 128 ビットブロック暗号を使う。
（条件 3）　鍵を変えずに処理するデータ量は、2^{20} ブロックまでとする。

2.2　XTS の仕様

本節では、2020 年 1 月現在の最新仕様である IEEE 1619-2018 [5]を参考に、XTS の仕様として暗号化および復号の手順を示す。IEEE はブロック暗号として AES を採用し、ブロックサイズを 128 ビットとしているため、本節で対象とする XTS は、XTS-AES とする。なお、ブロック暗号の種類やブロックサイズを限定しない一般的な XTS の詳細な仕様については、2019 年に発行された峯松氏による CRYPTREC 報告書を参照のこと [1]。

2.2.1　準備

XTS-AES の仕様を説明する準備として、説明で使用する記号とその意味を表 1 に示す。

表 1 XTS-AES の仕様の説明で用いる記号とその意味

項番	記号	意味
1	$\oplus$	排他的論理和
2	$\otimes$	既約多項式を $x^{128}+x^7+x^2+x+1$ とする GF(2^{128})の乗算
3	α	事前に定義された GF(2^{128})の生成元
4	$\leftarrow$	変数への代入
5	$\|\|$	データの連結 （例. $K1=001_2$、$K2=101010_2$とした場合、$K1\|\|K2=001101010_2$を得る）
6	$\|A\|$	データ A の長さ

2.2.2　暗号化の手順

XTS-AES は、ブロック暗号として AES を利用する暗号利用モードである。暗号化対象の平文 M に対し、AES の鍵を二つと、Tweak（調整値）と呼ばれる値を用いて暗号化を行う構成となっている。XTS-AES の暗号化関数 XTS-AES-Enc を以下のように定義する。

$$C \leftarrow \text{XTS-AES-Enc}(K, M, T)$$

ここで、C を XTS-AES 暗号化で得る暗号文、M を平文、T を Tweak、K をブロック暗号用の鍵とし、鍵 K は二つの鍵 K1 と K2 から K=K1||K2 で与えられるとする。なお、使用する鍵について、IEEE では 128 ビットの鍵を二つ、あるいは、256 ビットの鍵を二つ利用することが推奨されている。よって、全体の鍵 K の長さとしては、128 ビットの鍵と 256 ビットの鍵の場合それぞれに対して、256 ビットと 512 ビットとなる。Tweak としては、例えばハードディスクなどのストレージデバイスの場合、セクタ番号が利用される。平文に対しては、IEEE 1619-2018 [5]および NIST SP800-38E [4]において長さに対する制限が示されている。具体的に、平文 M は最低 1 ブロック（128 ビット）であり、2^{20} ブロックを超えないことが要求事項として示されている。よって、平文の長さ(ビット)の制限は以下のように表記できる。

$$128 \leqq |M| \leqq 2^{27} \ (= 2^{20} \cdot 128)$$

XTS-AES の暗号化の計算は、平文 M の長さがブロック長で割り切れる、すなわち、|M|がブロック長の倍数である場合とそうでない場合に、計算手順が異なる。|M|がブロック長の倍数である場合、平文 M をブロックごとに分割し、1 ブロック用の XTS-AES 暗号化の計算を最終ブロックまで繰り返し実行していく。一方、|M|がブロック長の倍数でない場合には、Ciphertext Stealing (CTS)と呼ばれる処理が導入される。

以下では、まず、1 ブロック用の XTS-AES 暗号化の計算内容を示した後、XTS-AES-Enc の計算内容を示す。

1 ブロック用の XTS-AES 暗号化関数 XTS-AES-blockEnc は、鍵 K、暗号化対象の平文ブロック M_b、Tweak の T および、何ブロック目の処理を行っているかを管理する変数 i を入力とし、出力を 1 ブロック分の暗号文 C_b とする関数として以下のように記述する。

$$C_b \leftarrow \text{XTS-AES-blockEnc}(K, M_b, T, i)$$

XTS-AES-blockEnc 関数のアルゴリズム Algorithm　1 に、XTS-AES-blockEnc のアルゴリズムをブロック図で表現したものを図 1 に示す。

1.	$T_E \leftarrow$ AES-enc(K2 , T)†
2.	$tmp_1 \leftarrow T_E \otimes \alpha^i$
3.	$tmp_2 \leftarrow M_b \oplus tmp_1$
4.	$tmp_3 \leftarrow$ AES-enc(K1 , tmp_2)
5.	$C_b \leftarrow tmp_3 \oplus tmp_1$

Algorithm 1 XTS-AES-blockEnc のアルゴリズム

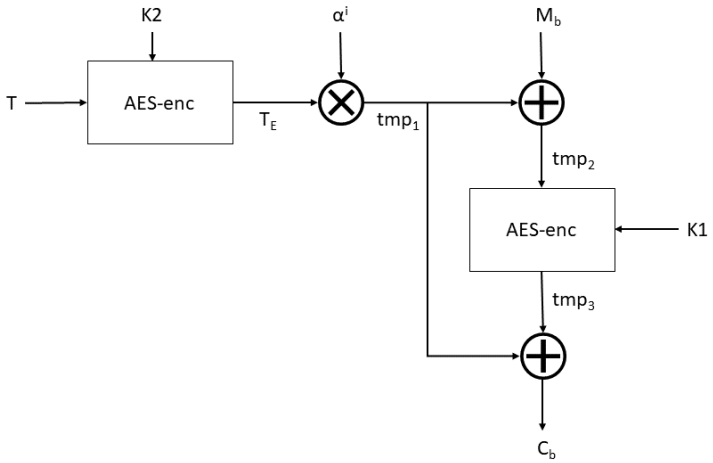


図 1 XTS-AES-blockEnc のブロック図

なお、α^iはガロア体 GF(2^{128})におけるべき乗演算を表し、i 個のαに対し、ガロア体 GF(2^{128})における乗算⊗を実行して得られる。AES-enc は AES の暗号化関数であり、例えば手順 1 の AES-enc(K2, T)は平文 T を鍵 K2 で AES 暗号化することを意味する。

次に、手順 2 で利用する GF(2^{128})の生成元αを利用した演算について説明する。IEEE 1619 で利用される演算⊗は、次式で与えられる既約多項式で定義されるガロア体 GF(2^{128})における乗算を意味する。

$$x^{128} + x^7 + x^2 + x + 1$$

生成元αを利用した乗算⊗の処理は、シフト演算と排他的論理和の処理で構成され、任意の元 A とαの乗算 B=A⊗αは、例えば Algorithm 2 のように計算される。なお、msb_1(A)を A の最上位ビットを与える関数、ビット単位の左シフト演算を「<<」、「0」が l 個連続することを表す記号として「0^l」と表記する。

† 具体的な計算方法は割愛するが、T_Eの値を 2 ブロック目以降でも使いまわすことで、2 ブロック目以降の Tweak に対する AES 暗号化の処理を不要にすることが可能である。

1.	if $msb_1(A) = 0$
2.	$B = A \ll 1$
3.	else
4.	$B = (A \ll 1) \oplus (0^{120}10000111)_2$
5.	end if

Algorithm 2 GF(2^{128})におけるαとの演算$\otimes$のアルゴリズム

次に、1 ブロック用の XTS-AES-blockEnc 関数を用いて、XTS-AES の暗号化関数 XTS-AES-Enc の計算内容について説明する。

IEEE 1619 では、平文 M を次式のように(m+1)ブロックからなるデータとして扱うため、本報告書においてもそれに従う。

$$M = M_0 \,||\, M_1 \,||\, \cdots ||\, M_{m-1} \,||\, M_m$$

ここで、m は$|M| < 128 \cdot m$を満たす最大の整数とする。このとき、m 個のブロック $M_0, M_1, \cdots, M_{m-1}$ は 128 ビットのブロックとなり、M_m は 127 ビット以下のブロックとなる。なお、|M|が 128 の倍数である場合、M_m は 0 となる。

XTS-AES-Enc のアルゴリズムを Algorithm 3 に示す。なお、入力を鍵 K=K1||K2、平文 M、Tweak T とし、出力を暗号文 C とする。

1.	for q ← 0 to (m-2)
2.	$C_q \leftarrow$ XTS-AES-blockEnc(K, M_q, T, q)
3.	end for
4.	b ← bit-size of M_m
5.	if b = 0 then
6.	$C_{m-1} \leftarrow$ XTS-AES-blockEnc(K, M_{m-1}, T, m-1)
7.	$C_m \leftarrow 0$
8.	Else
9.	$tmp_1 \leftarrow$ XTS-AES-blockEnc(K, M_{m-1}, T, m-1)
10.	$C_m \leftarrow$ first b bits of tmp_1
11.	$tmp_2 \leftarrow$ last (128 - b) bits of tmp_1
12.	$tmp_3 \leftarrow M_m \,||\, tmp_2$
13.	$C_{m-1} \leftarrow$ XTS-AES-blockEnc(K, tmp_3, T, m)
14.	endif
15.	$C \leftarrow C_0 ||\cdots ||C_{m-1}||C_m$

Algorithm 3 XTS-AES-Enc のアルゴリズム

Algorithm 3 のうち、手順 9～13 の処理が CTS と呼ばれる計算に相当する。CTS の計算をブロック図で表現したものを図 2 に示す。

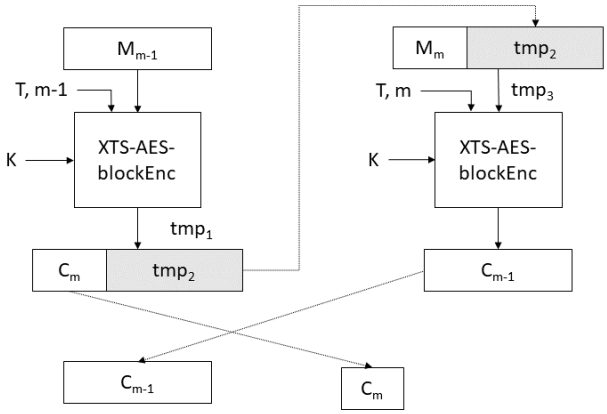


図 2 XTS-AES-Enc における Ciphertext Stealing のブロック図

2.2.3 復号の手順

ここでは、XTS-AES の復号手順を説明する。

XTS-AES の復号では、XTS-AES の暗号化と同様に、復号対象の暗号文をブロックに分割し、1 ブロック用の復号処理を繰り返し実行する構成となっている。まず、1 ブロック分の暗号文に対する XTS-AES の復号手順を示す。

1 ブロック用の XTS-AES 復号関数 XTS-AES-blockDec は、入力を鍵 K、復号対象の暗号文ブロック C_b、Tweak の T および、何ブロック目の処理を行っているかを管理する変数 i を入力とし、出力を 1 ブロック分の平文 M_b とする関数として以下のように定義する。

$$M_b \leftarrow \text{XTS-AES-blockDec}(K, C_b, T, i)$$

XTS-AES-blockDec 関数のアルゴリズムを Algorithm 4 に、XTS-AES-blockDec のアルゴリズムをブロック図で表現したものを**図 3** に示す。

1.	$T_E \leftarrow$ AES-enc(K2 , T)
2.	$tmp_1 \leftarrow T_E \otimes \alpha^i$
3.	$tmp_2 \leftarrow C_b \oplus tmp_1$
4.	$tmp_3 \leftarrow$ AES-dec(K1 , tmp_2)
5.	$M_b \leftarrow tmp_3 \oplus tmp_1$

Algorithm 4 XTS-AES-blockDec のアルゴリズム

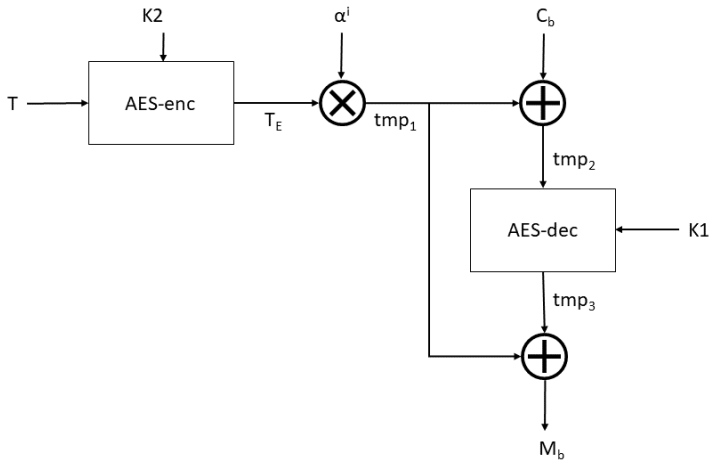


図 3 XTS-AES-blockDec のブロック図

次に、1 ブロック用の XTS-AES-blockDec 関数を用いて、XTS-AES の復号関数 XTS-AES-Dec の計算内容を説明する。

暗号化の場合と同様に、IEEE 1619 では暗号文 C を次式のように(m+1)ブロックからなるデータとして扱うため、本報告書においてもその表記に従う。

$$C = C_0 \| C_1 \| \cdots \| C_{m-1} \| C_m$$

ここで、m は|C| < 128・m を満たす最大の整数とする。このとき、m 個のブロック C_0, C_1, …, C_{m-1} は 128 ビットのブロックとなり、C_m は 127 ビット以下のブロックとなる。なお、|C|が 128 の倍数である場合、C_m は 0 となる。

XTS-AES-Dec のアルゴリズムを Algorithm 5 に示す。なお、鍵 K=K1||K2, 暗号文 C, Tweak T を入力とし、出力を平文 M とする。

1.	for q ← 0 to (m−2)
2.	M_q ← XTS-AES-blockDec(K, C_q, T, q)
3.	end for
4.	b ← bit-size of C_m
5.	if b = 0 then
6.	M_{m-1} ← XTS-AES-blockDec(K, C_{m-1}, T, m−1)
7.	M_m ← 0
8.	else
9.	tmp_1 ← XTS-AES-blockDec(K, C_{m-1}, T, m)
10.	M_m ← first b bits of tmp_1
11.	tmp_2 ← last (128 − b) bits of tmp_1
12.	tmp_3 ← C_m || tmp_2
13.	M_{m-1} ← XTS-AES-blockEnc(K, tmp_3, T, m−1)
14.	endif
15.	M ← M_0||… ||M_{m-1}||M_m

Algorithm 5 XTS-AES-Dec のアルゴリズム

暗号化の場合と同様に、暗号文の長さ(ビット)がブロックサイズで割り切れない場合に CTS と呼ばれる処理が実行される。CTS の処理は Algorithm 5 のうち手順 9～13 が対応する。図 4 に、XTS-AES の復号における CTS の計算をブロック図で示す。

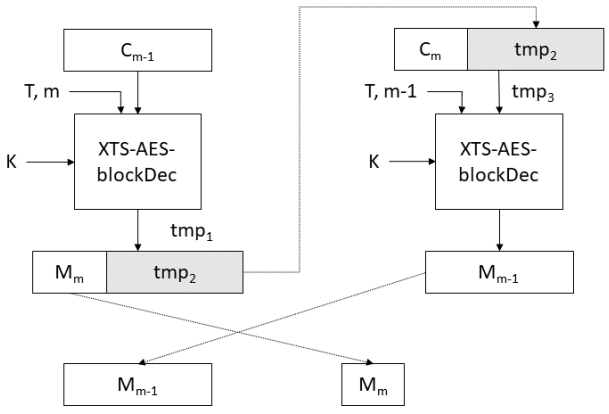


図 4 XTS-AES-Dec における Ciphertext Stealing のブロック図

2.3 実装的側面

実装された XTS-AES が NIST SP800-38E [4]に準拠しているかどうかを第三者の検証機関によって組織的に試験するための仕組みとして、米国 NIST から XTSVS(The XTS-AES Validation System)と呼ばれる試験仕様が提供されている [7]。本節では、XTSVS で示されている試験方法を説明する。また、NIST SP800-38E で要求事項とされているデータユニットサイズの制限に対し、OSS として提供されている暗号ライブラリでの対策状況を調査した結果を示し、対策の実装例を示す。

2.3.1 XTSVS の検証方法

XTSVS では、試験対象(Implementation Under Test, IUT)を実装したベンダーと第三者の検証機関の間で実施する試験手順、その手順の中でやり取りされるファイルのフォーマットが示されている。XTSVS に従って検証を行うことで、NIST SP800-38E で要求される以下の要件が満たされているかどうかを確認できる。

- IUT の暗号化/復号アルゴリズムの正しさ
- IUT がサポートするデータユニットサイズの制限

以下では、これらの検証方法を説明する。

2.3.1.1 暗号化/復号アルゴリズムの検証方法

暗号化/復号アルゴリズムの正しさは、第三者の検証機関が生成するテストベクタを使用した試験を実施することで検証される。テストベクタとは、入力値と出力値の正しい組のことである。XTS の場合、例えば、鍵、平文、Tweak を入力とし、暗号文を出力とした組である。

XTSVS では、ベンダーが IUT の試験に必要なパラメータの情報(鍵長、平文サイズ、Tweak のフォーマットなど)を第三者の検証機関に提出する。第三者の検証機関は提供されたパラメータを基に、リファレンスとなる XTS の実装を使ってテストベクタを生成する。テストベクタの数は、暗号化/復号機能それぞれにおいて、各平文/暗号文サイズに対して 100 個である。ベンダーはテストベクタの生成に利用した入力値を第三者の検証機関から入手し、IUT で得た出力値を検証機関に提出する。第三者の検証機関はテストベクタを用いて、提出された IUT の出力値が正しい値であるかどうかを確認することでアルゴリズムが適切に実装されていることを検証可能である。なお、NIST からテストベクタのサンプルが公開されているため [‡]、ベンダーは評価対象のモジュールにおける XTS-AES のアルゴリズムとしての正しさを検証可能である。

[‡] NIST CAVP Testing: Block Cipher Modes XTS-AES Test Vectors (https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation-Program/documents/aes/XTSTestVectors.zip

2.3.1.2 サポートするデータユニットサイズの試験方法

NIST SP800-38E では、XTS の利用上の注意点として、「入力されるデータユニットが 2^{20} ブロック以下であること」[§]という要求事項が示されている。ここで、データユニットとは、XTS の暗号化関数に平文として入力されるデータのことである。XTSVS では、ベンダーは IUT がこの要求事項を満たす証拠を提出する必要があり、テストベクタによる試験は実施されない。なお、証拠となる提出物の詳細や試験方法については、2020 年 1 月現在の最新版である 2013 年 9 月 5 日更新版の XTSVS では示されていない。

2.3.2 データユニットサイズの制限に対する対策例

XTSVS では、データユニットサイズに関連する要求事項が示されているが、その要求事項を満たすための具体的な実装方法は示されていない。そこで、本調査では、OSS として公開されている暗号ライブラリの中で、XTS が実装されているものを選定し、データユニット長の制限に対してどのような対策が実施されているかを確認した。調査した OSS に対する対策状況を表 2 に示す。

表 2 データユニットサイズに関する調査対象暗号ライブラリ

項番	暗号ライブラリ名	バージョン	URL	対策
1	BoringSSL	fips-20190808	https://boringssl.googlesource.com/boringssl/	未
2	GnuTLS	3_6_11	https://www.gnutls.org/download.html	未
3	LibreSSL	v3.0.1	https://ftp.openbsd.org/pub/OpenBSD/LibreSSL/	未
4	mbed TLS	2.16.4	https://tls.mbed.org/download-archive	**済**
5	OpenSSL	1.1.1d	https://www.openssl.org/source/	未
6	OpenSSL(FIPS)	fips_2.0.16	https://www.openssl.org/source/	**済**
7	WolfCrypt	v4.2.0-stable	https://www.wolfssl.jp/download/	未

調査の結果、OpenSSL(FIPS)と mbed TLS では、データユニットサイズに関する対策が導入されていることを確認できた。実際の対策例として、OpenSSL(FIPS)と mbed TLS での対策の実装方法を示す。図 5 は、FIPS 認証版の OpenSSL (openssl-fips-2.0.16)に実装されている XTS のソースコード(e_aes.c)の一部を抜粋したものである。1236 行目に注目すると、XTS の暗号化関数に入力される平文の長さを表す引数 len(単位はバイト)に対し、既定の長さよりも大きいかどうかのチェックを行っている。具体的には、XTS で利用する AES のブロックサイズは 16 バイトであるため、引数 len が(2^{20} × 16)よりも大きい場合に引数エラーとする実装となっていることが確認できる。

次に mbed TLS の場合を示す。図 6 は、mbed TLS(mbedtls-2.16.4)に実装されている XTS のソースコード(aes.c)の一部を抜粋したものである。1198 行目に注目すると、mbed

[§] 原文では「The length of the data unit for any instance of an implementation of XTS-AES shall not exceed 2^{20} AES blocks. 」と記述されている。

TLS も OpenSSL(FIPS)と同様に、XTS の暗号化関数に渡された平文の長さを表す引数 length に対して、大きさのチェックをしていることが確認できる。

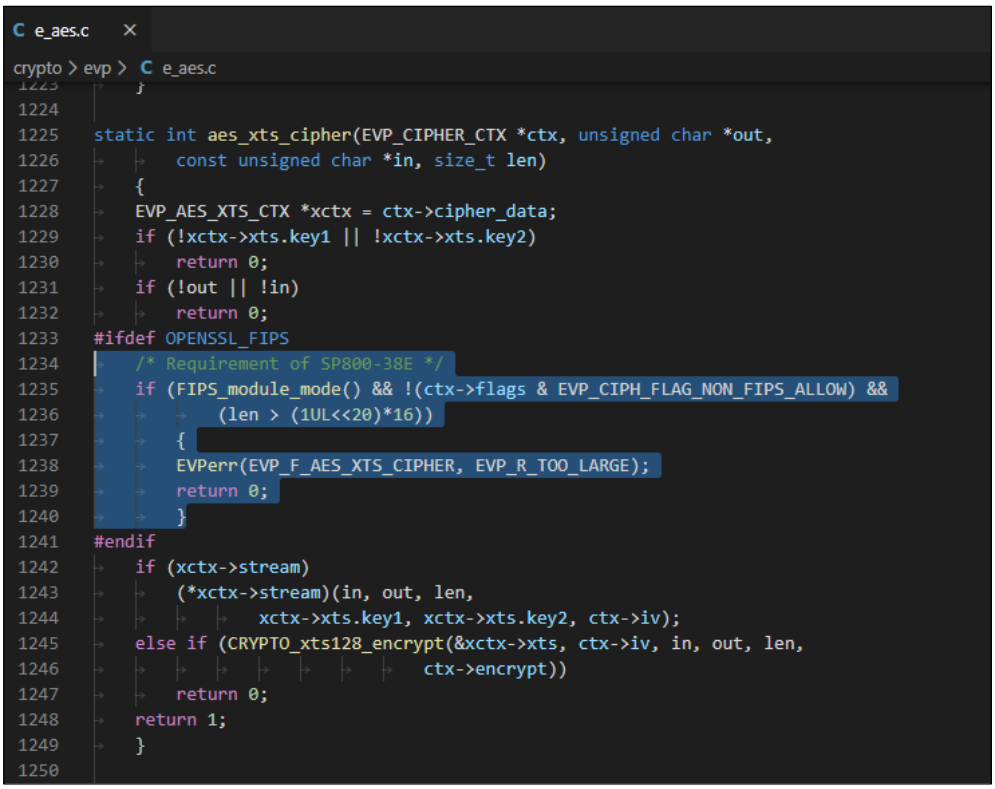


図 5　データユニットサイズの制限に対する対策例（FIPS 版の OpenSSL）

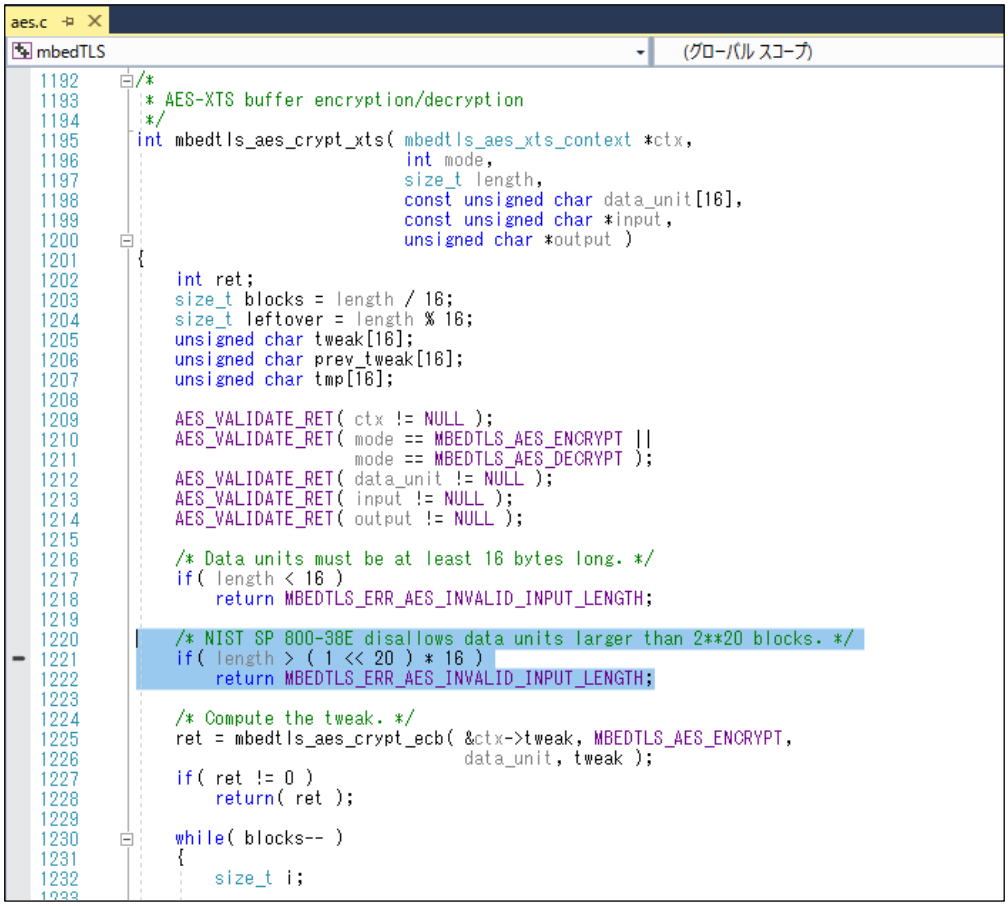


図 6　データユニットサイズの制限に対する対策例（mbed TLS）

3　XTS の市販製品などでの実装採用状況

XTS は、現在までにストレージ暗号化ソフトの製品での採用実績があり、例えば Microsoft Windows 搭載の BitLocker や macOS 搭載の FileVault 2 などの製品での採用実績がある。

本章では、XTS の実装採用状況として、XTS を採用している製品を販売しているベンダー数や製品の種類について調査した結果を示す。本調査では、セキュリティ製品の第三者認証の主流である CMVP(Cryptographic Module Validation Program)認証を取得している製品の中で、認証アルゴリズムとして XTS を含む製品を対象とする。

CMVP 認証製品を対象とする理由を以下に述べる。本報告書は XTS を CRYPTREC 暗号リストに採択するか否かを判断するための材料の提供を目的としている。今後、XTS が CRYPTREC 暗号リストに採択された場合に、日本でのセキュリティ第三者認証制度である JCMVP（Japan Cryptographic Module Validation Program）の試験対象に加わる。JCMVP は CMVP に基づく試験を実施していることから、本調査の対象として XTS の試験を先行して実施している CMVP 認証製品を採用した。

3.1 CMVP 認証製品の抽出方法

本節では、認証アルゴリズムとして XTS を含む CMVP 認証製品の抽出方法を示す。

CMVP は、米国およびカナダが運営する暗号モジュール試験および認証制度であり、NIST FIPS 140-2 (Federal Information Processing Standard 140-2)を採用している。CMVP 認証を取得した製品の一覧を検索可能な Web ページを NIST が公開しており、認証番号だけでなく、ベンダー名やモジュール名、取得年などをキーとして検索可能なインタフェースとなっている [**]。

本調査で抽出したい製品は XTS を認証アルゴリズムに含む CMVP 認証製品である。検索ページでは検索する対象として認証アルゴリズムの選択が可能であるが、AES に関しては、「AES」と「CCM」の 2 種類しか選択できない。XTS が対象となる「AES」で抽出した場合、2020 年 1 月 15 日現在、約 1200 件の製品が CMVP 認証を取得している。これらの中には XTS を認証アルゴリズムとして含まない製品が多くあるため、対象を絞り込む必要がある。

本調査では、XTS を認証アルゴリズムに含む CMVP 認証製品を取得するために、CMVP 認証の過程で実施する必要のある CAVP（Cryptographic Algorithm Validation Program）に注目した。

CAVP は、製品に搭載された暗号モジュールのアルゴリズムが正しく実装されていることを試験するものであり、CMVP と同様に、FIPS 140-2 に基づく。2 章で解説した通り、XTS の CAVP 試験は XTSVS に従い実施される。CAVP に対しても、認証を取得した製品の一覧を検索可能なページが NIST から公開されており、ベンダー名や製品名などをキーとして検索可能なインタフェースとなっている [††]。CAVP の検索ページでは、検索の際に指定の認証暗号アルゴリズムによって絞り込むことが可能であり、検索ページの「Supports Algorithm(s)」に「AES-XTS」を選択することで認証アルゴリズムに XTS を含む CAVP 認証製品を抽出可能である。

CAVP 認証は CMVP 試験の過程で取得するため、XTS を含む CAVP 認証製品が CMVP 認証を取得しているかどうかを確認する必要がある。一方で、CMVP の検索ページでは、特定の CAVP 認証製品に対して CMVP 認証製品となったかどうかを検索可能な項目が提供されていない。そこで、本調査では、CMVP と CAVP の検索項目として共通のベンダー名を使った抽出を実施した。具体的には、CAVP 認証を取得したベンダーに対し、AES を認証アルゴリズムに含む製品の CMVP 取得状況を調査した。さらに、検索結果として出力される CMVP 製品に対し、その製品が FIPS 140-2 のセキュリティ要件をどのように満たしているかを説明するセキュリティポリシーの中で、XTS が認証アルゴリズムとして記載されているかどうかを確認した。

XTS を認証アルゴリズムに含む CMVP 取得製品の抽出方法を図 7 に示す。

[**] NIST Cryptographic Module Validation Program 検索 Web ページ
https://csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search

[††] NIST Cryptographic Algorithm Validation Program 検索ページ
https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/validation

1: CAVP 検索ページに対し、XTS-AES の CAVP 製品を取得するため、「Supports Algorithm(s)に「AES-XTS」を入力し、検索する
2: CMVP 検索ページに対し、1 で得たベンダー名を「Vendor:」に入力し、「Algorithm:」に「AES」を選択し、検索する
3: 2 で得た結果に対し、各 CMVP 取得製品のセキュリティポリシーの内容を確認し、認証アルゴリズムとして XTS が記載されているかどうかを確認する

図 7 XTS を認証アルゴリズムに含む CMVP 取得製品の抽出方法

3.2 CMVP 認証製品の抽出結果

本節では、3.1 節で示した XTS の CMVP 認証製品の抽出方法に基づき抽出した結果を示す。

表 3 に、2020 年 1 月現在までに XTS を認証アルゴリズムとして含む CMVP 認証を取得したベンダー数と取得数を示す。

表 3 XTS を認証アルゴリズムに含む CMVP 認証の取得数

項番	モジュールタイプ	ベンダー数	CMVP 取得数
1	Software	63	182
2	Hardware	34	111
3	Firmware	3	3
	合計	90[##]	296

XTS を含む CMVP 認証は、2020 年 1 月現在までに 90 社のベンダーが 296 件の CMVP 認証を取得していることがわかった。XTS はストレージデバイス向けの暗号利用モードであるが、表 3 より、ストレージ製品で取得されると思われる Hardware タイプよりも Software タイプの方が多く取得されている結果となった。

3.3 XTS の市販製品などでの実装採用状況の考察

本節では、XTS の CMVP 認証を取得しているベンダーや製品の種類を整理することで、XTS の実装採用状況を考察する。

3.3.1 CMVP 取得ベンダーからみた XTS の採用状況の考察

本項では、XTS の CMVP 認証を多く取得しているベンダーがどのような製品で CMVP 認証を取得しているかを確認することで、XTS の採用状況を考察する。表 2 に示したベンダー数のうち、CMVP 認証を 10 件以上取得しているベンダーを表 4 に示す。

[##] ベンダー数の合計は、Software タイプと Hardware タイプの両方で CMVP 認証を取得するベンダーもあるため、その場合は 1 ベンダーとして計算している。

表 4　CMVP 認証を 10 件以上取得しているベンダーとその件数

項番	ベンダー名	CMVP 取得数
1	Microsoft Corporation	30
2	Samsung Electronics Co., Ltd.	20
3	Apple Inc.	18
4	Western Digital Company (HGST の CMVP 取得数を含む)	17
5	RSA	16

XTS の CMVP 認証を最も取得しているベンダーは Microsoft であり、CMVP 取得数は 30 件である。これらの CMVP 取得製品を調べた結果、ブートローダーなどの Windows OS 向けのモジュールに加え、BitLocker 関連のモジュールに対し CMVP 認証が取得されている。CMVP 認証の取得方法として、BitLocker 製品単体で CMVP を取得しておらず、BitLocker を構成するモジュールや BitLocker を使用するモジュール単位で CMVP が取得されていることが分かった。

2 番目に多く CMVP 認証を取得しているベンダーは Samsung Electronics であり、20 件であった。調査の結果、Samsung Electronics の XTS の CMVP 認証取得製品の多くは、自己暗号化ドライブ（Self Encrypting Drive, SED）と呼ばれるストレージであり、データの書き込み時にデータを自動的に暗号化する機能を内蔵している。順位通りではないが、SED に関連すると、4 番目の Western Digital が取得している XTS の CMVP 製品は全て SED 製品である。SED については、コンピュータの安全性を高める国際業界標準規格を策定している TCG（Trusted Computing Group）で規格化されている。その規格の中では、データの暗号化に XTS-AES や AES-CBC を利用することとされているため、CMVP 認証取得製品以外でも TCG 準拠の SED 製品で XTS が普及していると推測される。

3 番目に多く CMVP 認証を取得しているベンダーは Apple であり、18 件であった。調査の結果、これら製品は macOS や iOS 向けの暗号モジュールであることがわかった。Apple からは、macOS 向けのストレージ暗号化機能として FileVault 2 が提供されているため、これら CMVP 製品が FileVault で使用されているかどうかを調査した。セキュリティポリシーを調査した限りでは、暗号モジュールの適用先として FileVault と明記されているものは見つからなかったものの、XTS の利用上の注意として、ハードウェアストレージアプリケーションでの利用に制限されていることがわかった。

最後に、5 番目に多く CMVP 認証を取得しているベンダーは RSA 社であり、16 件であった。これらの製品は RSA 社から提供されている暗号ライブラリ BSAFE を構成するモジュールであった。BSAFE は汎用的な暗号ライブラリであるが、最新の CMVP 取得製品では利用先がハードウェアストレージアプリケーションに限定されており、ストレージ暗号化での利用を想定したものであると推定できる。

また、CMVP 取得件数 10 件に迫るベンダーとして、国内では、日立製作所が XTS の CMVP 取得件数 8 件であり、ストレージ仮想化製品で取得している。

3.3.2 CMVP 取得製品の種類からみた採用状況の考察

本項では、XTS の CMVP 認証を取得している製品がどのような製品種別であるかを確認することで、XTS の採用状況を考察する。XTS を認証アルゴリズムに含む CMVP 認証製品において、製品種別に対する CMVP 件数と取得ベンダー数を表 5 に示す。

表 5 XTS を認証アルゴリズムに含む CMVP 認証製品の製品種別とその件数

項番	製品種別	CMVP 取得件数	CMVP 取得ベンダー数
1	暗号ライブラリ	190	69
2	ストレージ暗号化ソフトウェア	17	1
3	ストレージ（HDD、SSD、SED、USB メモリなど）	79	19
4	その他	10	8

表 5 より、CMVP 認証取得製品 296 件のうち、約 64%の 190 件が項番 1 の暗号ライブラリであることがわかる。この暗号ライブラリの CMVP 取得ベンダー数は 69 社であり、全 90 社に対して約 76%であった。これらの CMVP のセキュリティポリシーを調査した結果、XTS に対して利用用途をストレージ暗号化に制限しているものや、データユニットサイズに対する注意を明記しているものがある一方で、そのような記載がないものもあった。注意点が記載されていないものについては、単に XTS をブロック暗号の利用モードのひとつとして扱っており、汎用的な暗号ライブラリの機能のバリエーションをそろえる意味で搭載されていると推定される。そのような暗号ライブラリにおいては、製品にデプロイした際に XTS を誤用するおそれがあると考えられる。項番 2 のストレージ暗号化ソフトウェア関連の CMVP 認証取得件数は 19 件であり、Microsoft の BitLocker 関連の製品である。今回抽出した CMVP 取得製品の中では、モジュール名に製品名が明記されているものは BitLocker のみであった。項番 3 のストレージ製品は、3.3.1 項で述べた Samsung や Western Digital 以外にも 17 社あり、SED や HDD、USB の暗号化向けに CMVP 認証を取得していることがわかった。国内では、東芝が SED 関連で XTS の CMVP 認証を取得している。最後に、項番 1～3 のカテゴリに該当しなかった CMVP 取得製品として、無線 LAN アクセスポイントなど、ストレージ暗号化用途として該当しないと思われる製品で採用されていることがわかった。

3.4 XTS の市販製品などでの実装採用状況のまとめ

XTS の市販製品などでの実装採用状況についてまとめる。XTS が採用されている代表的な製品としては Microsoft 社の BitLocker や Apple 社の FileVault 2 などが挙げられるが、調査の結果、ストレージ暗号化製品としては SED や HDD、USB などのストレージ製品で採用されており、CMVP 認証が取得されていることが分かった。また、CMVP 認証製品の中には、XTS を搭載した暗号ライブラリも含まれている。これらの暗号ライブラリには、ストレージ暗号化の用途で XTS を搭載しているものもあることから、製品として CMVP 認証が取得されていないものであっても、XTS が採用されている可能性があると考えられる。

4　XTS の公開されている評価結果に関する調査

IEEE 1619 [5]や NIST SP800-38E [4]などの仕様において、XTS はストレージ暗号化として利用することが記述されており、ストレージのデータ保護の観点からも重要な技術であると考えられる。そのような状況を踏まえて、本章では今後の利用を考慮した際に、本調査対象である XTS に対して、客観的な実装性能評価結果として学術論文や製品提供を行う企業が公開するホワイトペーパーに関する調査を行う。また、今までの経験より、新しい暗号アルゴリズムを実装するケースにおいて実装バグや脆弱性を作り込んでしまうケースというのは多いと考えられる。XTS はデータ保護のために利用されており、脆弱性や実装バグがあると影響が比較的大きなことが予想される。この状況を踏まえて今まで公開されている製品などでの脆弱性の有無を確認することで XTS を実装した製品が安定してきているかどうかを確認する。

4.1　客観性のある実装性能評価に関する調査

客観性のある実装性能評価に関する調査として、学術論文や AES-XTS を搭載した製品に関するベンチマークの情報の有無、どのような組織が情報を公開しているのかを調査する。その調査方法としては、検索エンジンを用いて、表 6 に示す検索ワードの組合せを用いて、ベースとなるキーワードとの組合せによって調査を実施した。

表 6　客観性のある実装性能評価に関する調査のためのキーワード

項番	キーワード	選定理由
1	XTS	調査対象そのものであるため （ベースとなるキーワード）
2	Performance	実装性能に関係するキーワードであるため
3	Implement	実装に関するキーワードであるため
4	Hardware	ハードウェアに関する情報であれば多く存在していると期待したため

上記の調査によって、発見された情報を（表 7）に整理する。なお、Blog などの情報については、情報に関する信憑性を客観性に判断することが困難であると考え、本調査では抽出する項目から除外している。

表 7　客観性のある実装性能評価に関する調査結果

項番	タイトル等	概要	公開 URL
1	High Performance Storage Encryption on Intel® Architecture Processors	Intel 社による AES-NI を用いた AES-XTS（鍵長：128 ビット）について、Intel® Core™i5 650 processor および Hyper-Threading 技術を併用することで、巨大なバッファサイズに対して集約レートで最大 18GB/s を実現することを示している。 なお、情報が 2010 年 8 月と最新環境ではないことに注意が必要である。	https://pdfs.semanticscholar.org/d98d/35c0446e1e1f14d79545316afa647972d42a.pdf
2	Intel® AES-NI Performance Enhancements: HyTrust DataControl Case Study	ケーススタディとして、HyTrust 社が提供する DataControl による暗号化および鍵管理ソリューションを提供している。このホワイトペーパーでは、彼らの提供しているサービスにおける AES-XTS に関する性能について情報を共有している。 なお、情報が 2014 年 9 月と最新環境ではないことに注意が必要である。	https://software.intel.com/en-us/articles/intel-aes-ni-performance-enhancements-hytrust-datacontrol-case-study
3	Improving dm-crypt performance for XTS-AES mode through extended requests: first results	省電力向け Atmel Board において AES-ECB の暗号エンジンが搭載されている。Linux でフルディスク暗号化に使用される dm-crypt モジュールを用いて、大きなブロックサイズを検討するときに緩和できる制限があることが示されている。研究成果として、ほぼ 2 倍の性能向上が可能であることが示されている。	https://hal.inria.fr/hal-01399967/file/grehack16_dmcrypt_performance_for_aes_xts.pdf
4	Geekbench Browser	Primate Labs Inc.が提供するマルチプラットフォーム対応の総合ベンチマークツールである。最新版の「Geekbench 5」では、以前のバージョンより正確に CPU パフォーマンスを測定するため、暗号化、機械学習、拡張現実のような応用を配慮した測定項目になっている。	https://browser.geekbench.com/

XTS はディスク暗号化で利用されていることから、利用者が実装性能に関して興味を持つ技術であると考えられ、OS ベンダーやストレージベンダーなどから多くの実装性能に関する情報が公開されていることを期待したが、表 6 で示したキーワードによる検索結果においては、表 7 の項番 1 ～ 2 のように 2010 年など比較的古い環境での実装性能に関する情報が公開されていた。その当時のコンピュータリソースなどを考慮すると XTS による暗号化に関するオーバーヘッドを意識する必要があったのではないかと推測する。この推測を裏付けるものとして、項番 4 にある Geekbench Browser は、2019 年 9 月に最新版のバージョン 5 が公開されており、CPU のベンチマーク項目として AES-XTS に関する実装性能が収集されている。このことから、AES-XTS は現在においても CPU の性能を判断するために重要な性能測定項目であることが予想され、コンピュータリソースを多く消費する処理として注目されていると考えられる。また、学術論文においては複数 OS や異なる CPU 環境での実装性能を測定すると言った新規性が低いことが予想されることから、そのような XTS に関する実装性能に関する論文は発見できず、IoT のようなリソースや実装されている機能に制約条件の多い環境下での高速化に関する論文が存在していた。

4.2 脆弱性や実装バグに関する調査

客観的な実装物に対する脆弱性等の調査として、米国 NIST が公開している National Vulnerability Database（https://nvd.nist.gov/vuln/search）において、表 8 に示すキーワードを用いて検索を実施した。また、Search Vulnerability Database での検索条件を、表 9 に示す。

表 8 脆弱性や実装バグに関する調査のためのキーワード

項番	キーワード	選定理由
1	AES	調査対象そのものであるため（ベースとなるキーワード）
2	XTS	調査対象そのものであるため（ベースとなるキーワード）
3	storage	対象を示すキーワードであるため
4	crypt	暗号に関するキーワードであるため

表 9 Search Vulnerability Database における検索条件

項番	設定項目	設定値
1	Search Type	Basic
2	Results Type	Overview
3	Search Type	All Time
4	Contains HyperLinks	チェックなし

上記の条件で検索を行うことで 150 件以上の検索結果が得られるが、その多くの検索は「teXTS」のようにキーワードの一部を包含するようなものが多いため、検索結果に対して Summary を確認し、条件に合致しているものかどうかを確認した。本調査で得られた結果としては、表 10 に示す 1 件の CVE がストレージ暗号化に関する脆弱性として存在した。しかしながら、この脆弱性は、Intel Optane メモリーモジュールを搭載したシステムにおけるストレージメディアの情報漏えいの脆弱性によって攻撃者は物理アクセスを介してデータを回復できる可能性がある脆弱性であったため、ディスク暗号化としては XTS を利用していることが予想されるが、XTS に起因するような脆弱性ではなかった。

表 10 Search Vulnerability Database における検索結果

項番	CVE	サマリー	URL
1	CVE-2018-3619	Information disclosure vulnerability in storage media in systems with Intel Optane memory module with Whole Disk Encryption may allow an attacker to recover data via physical access.	https://nvd.nist.gov/vuln/detail/CVE-2018-3619

また、XTS の機能が実装されている暗号ライブラリやソフトウェアにおいて、XTS の機能が利用されている中で実装コードの改善やバグ抽出されることで、ソフトウェアの実装として洗練されてきているか、XTS を実装している OSS である暗号ライブラリに対して、GitHub の Issue やメーリングリストを調査した。その結果として、XTS に関連する実装部分に対しての実装コードの改善が行われていることを知ることができた。このことから、OSS コミュニティとしてコミットなどの活発なコミュニティや OSS である暗号ライブラリとして FIPS 認証を取得するようなコミュニティであれば、コード品質も日々改善され、XTS に関連する実装部分も安定していると想定される。

4.3　XTS の公開されている評価結果に関するまとめ

XTS に関する公開情報の評価としてまとめると、XTS に対する実装性能の第三者が作成した情報はあまり存在していなかったが、AES-NI のような暗号機能を補強する機能があることによって、利用者が意識することなく暗号機能が利用されているのではないかと考える。また、他の暗号利用モードと比較すると新しい部類に入る XTS ではあるが、XTSVS のようなテストベクタの充実などによって、実装者が気軽に正しく実装できているかどうかを確認できる土壌があるため、XTS という暗号利用モード自体の実装ミスはなく、その機能が正しく利用されているかどうかという視点が重要であると考える。

5　他の暗号利用モードとの実装性能比較調査

本章では、実装性能調査対象の暗号利用モードである XTS と CRYPTREC 暗号リスト§§（2020 年 1 月現在の最新版：CRYPTREC LS-0001-2012R4）で採択されている暗号利用モードとの実装性能評価を実施することで、現在採択されている暗号利用モードと比較して実装性能について調査する。実装性能調査を行う際に注目した測定対象の選定観点として、評価対象となる「暗号アルゴリズム」、評価対象となる「暗号利用モード」、実環境における実装性能評価対象となる「暗号ライブラリ」の 3 つになる。以下に、それらの測定の観点における選定結果を示す。

＜評価対象となる「暗号アルゴリズム」＞

評価対象となる暗号アルゴリズムとして、NIST SP800-38E [4]や IEEE 1619-2018 [5]などで XTS との組合せとしてドキュメント化されている AES を評価時の暗号アルゴリズムとして選定した。その選定理由としては、CMVP での認証対象であることから多くの製品や暗号ライブラリなどで実装が多くされていると期待されるためである。また、利用する共通鍵暗号アルゴリズムの鍵長としては、AES-XTS の仕様で規定されている 128bit および 256bit を評価対象とした。

＜評価対象となる「暗号利用モード」＞

本実装性能調査として対象とする暗号利用モードは、現在、CRYPTREC 暗号リストで採択されている暗号利用モード（表 11）および、CRYPTREC 暗号リストでは採択されていない ECB モードを共通鍵暗号アルゴリズムとしての素朴な実装性能として判断する基準値となると考えて調査対象とした。

表 11　CRYPTREC 暗号リストにおける暗号利用モード

暗号利用モード	秘匿モード	CBC
		CFB
		CTR
		OFB
	認証付き秘匿モード	CCM
		GCM

以下に、本実装評価調査として対象となる暗号利用モードとその選定理由を以下に整理する。（表 12）

§§ https://www.cryptrec.go.jp/list/cryptrec-ls-0001-2012r4.pdf

表 12 実装評価対象一覧

項番	暗号利用モード名	選定理由
1	XTS	本調査での評価対象
2	ECB	実装性能時の基準値
3	CBC	CRYPTREC 暗号リストで採択
4	CFB	同上
5	CTR	同上
6	OFB	同上
7	CCM	同上
8	GCM	同上

＜実環境における実装性能評価対象となる「暗号ライブラリ」＞

実装性能評価対象とする暗号ライブラリとして、2 つの暗号ライブラリを選定することとした。その理由としては、選定した暗号ライブラリ特有の性能特性である可能性をできるだけ排除することを考慮して、独立した 2 実装を選択するようにした。その結果として、評価対象として選択した暗号ライブラリは、世界中で広く利用されていると予想される OpenSSL*** と組み込み環境などで利用されることが想定されている WolfSSL 社 ††† が提供している WolfCrypt ‡‡‡ とした。また、この2つの暗号ライブラリには、性能測定を行うための機能が実装されていることから、評価者の実装による実装性能への影響を排除し、他の第三者も確認できることも考慮して選定している。

本調査において利用した暗号ライブラリについて、取得先およびバージョンに関する情報は以下のとおり。（表 13）

表 13 評価で利用した暗号ラリブラリ

項番	暗号ライブラリ名	バージョン	取得先 URL
1	OpenSSL §§§	1.1.1d	https://www.openssl.org/source/
2	WolfCrypt	4.2.0	https://www.wolfssl.jp/download/

*** https://www.openssl.org/

††† https://www.wolfssl.com/

‡‡‡ https://www.wolfssl.com/products/wolfcrypt-2/

§§§ OpenSSL は FIPS 認証を受けており、FIPS 版モジュールも当該 Web サイトで公開している。なお、2020 年 1 月現在で公開されている FIPS 版モジュールは openssl-fips-2.0.16.tar.gz である。なお、ビルドについては通常の OpenSSL と同様に実施することができる。

5.1 実装性能評価に向けた準備

実装性能評価を実施する上での準備として、本調査で利用した「実装性能評価を行った環境」および「暗号ライブラリのビルド方法」を示す。

5.1.1 実装性能評価を行った環境

XTS で期待される利用がディスク暗号化であることを考慮し、主な OS である Microsoft 社 Windows、Apple 社 macOS、Linux の 3 環境を対象とし、実環境での性能評価を実施した。それぞれの環境に関するスペックは以下のとおりである。

＜Windows 環境＞

OS: Windows 10 Pro 64-bit Version 1909 (Build 18363.535)
CPU: Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz (12 CPUs)
Memory: 64GB

＜macOS 環境＞

OS: macOS Catalina 10.15.2
CPU: 2.4GHz dual-core Intel Core i5 processor
Memory: 16GB

＜Linux 環境＞

OS: Ubuntu 18.04.3 LTS
CPU: Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
Memory: 32GB

5.1.2 暗号ライブラリのビルド方法

5.1.2.1 OpenSSL

各実装評価を行った環境での OpenSSL に関するビルド方法を以下に示す。なお、現在の OS やアプリケーションの状況を考慮して、x64 向けのみを評価対象としている。

＜Windows 環境＞

```
perl Configure VC-WIN64A
nmake
nmake test
```

＜macOS 環境＞

```
./Configure darwin64-x86_64-cc
make
make test
```

＜Linux 環境＞

```
./Configure linux-x86_64
make
make test
```

5.1.2.2 WolfCrypt

各実装評価を行った環境での WolfCrypt に関するビルド方法を以下に示す。なお、現在の OS やアプリケーションの状況を考慮して、x64 向けのみを評価対象としている。

＜Windows 環境＞

Windows における WolfCrypt のビルドでは、圧縮ファイルに同梱されている wolfssl.vcxproj ファイルを用いて Visual Studio 上で行う必要がある。なお、x64 向けでの評価を行うために、以下に示す変更 ****を実施する必要がある。

・ベンチマークアプリが同梱ファイルでは、x86 向けのビルド設定のみであるため、ベンチマークアプリに対して、プロジェクトに x64 プラットフォームを追加する

・構成プロパティにおける「プログラム全体の最適化」オプションが無効になっているので、以下に示すフラグを有効にする:

- ・WOLFSSL_AES_XTS
- ・WOLFSSL_AES_CFB
- ・WOLFSSL_AES_COUNTER
- ・HAVE_CAMELLIA
- ・WOLFSSL_AESNI

＜macOS 環境＞

```
./configure --enable-xts --enable-aescfb --enable-aesctr --enable-camellia --
enable-aesni
make
make test
```

＜Linux 環境＞

Linux 環境でのビルドについては macOS 環境と同様に実行することができるため、macOS 環境を参照のこと。

**** ベンチマークアプリのプロジェクトファイル上は文字コードとして Shift-JIS が指定されているが、ソースコードが UTF-8 なので、ベンチマークアプリのソースコード側 Shift-JIS に変更

5.2 実装性能評価に向けた測定

実装性能評価対象である OpenSSL および WolfCrypt で実装されているベンチマーク機能として、OpenSSL では speed コマンドがあり、WolfCrypt では benchmark がある。ここでは異なる暗号ライブラリでのベンチマーク機能における測定の観点および測定対象となっている API について情報を示す。また、それぞれのベンチマーク機能に関する実行方法について情報を示す。

5.2.1.1 OpenSSL における測定の観点

OpenSSL で実装されているベンチマーク機能である speed コマンドは、今回評価対象である XTS だけではなく OpenSSL で実装されている多くの暗号アルゴリズム [††††]の性能測定を容易に実施することができる。この speed コマンドでの測定の観点は、該当する機能に対して 3 秒間のループを実施する。その際に処理を行ったデータ量に対して実行時間を用いてスループットを計算している。なお、その測定結果は対象データ(平文)として、16bytes、64bytes、256bytes、1024bytes、8192bytes、16384bytes を入力とし、暗号化/復号処理に関するスループットを測定が可能である。

また、speed コマンドで測定対象となる API に関する情報としては、各暗号処理を EVP(The Digital EnVeloPe library)と呼ばれる共通的な API を用いて実行している。測定対象となる関数は EVP_EncryptUpdate / EVP_DecryptUpdate であり、暗号化や復号機能を評価対象としている。

5.2.1.2 OpenSSL におけるベンチマーク実行方法

OpenSSL におけるベンチマークコマンドおよび本調査において利用したオプション(表 14)について説明を行う。

ベンチマークコマンド(OpenSSL)

openssl speed -evp alg -decrypt

表 14 本調査における利用オプション

項番	オプション	概要
1	-evp alg	EVP に対して alg によって指定された暗号アルゴリズムを測定対象として指定する
2	-decrypt	上記の-evp を指定した場合のみ有効であり、復号処理に関する性能測定を実施する ※ 本オプションを指定しない場合は、暗号化処理を実施する

[††††] 利用可能な暗号アルゴリズム一覧を取得したい場合には「openssl list -cipher-algorithms」を実行することで確認できる。

5.2.1.3 WolfCrypt における測定の観点

WolfCrypt で実装されているベンチマーク機能は benchmark である。今回評価対象である XTS だけではなく WolfCrypt で実装されている多くの暗号アルゴリズムの性能測定を容易に実施することができる。この benchmark での測定の観点は、入力データとして 1048576 bytes の平文に対して、測定対象機能の処理を 5 回実施する。その処理が 1 秒を経過していなければ、この処理を繰り返し実施する。また、1 秒経過していれば計測を終了する。その際に処理を行ったデータ量に対して実行時間を用いてスループットを計算している。

また、benchmark で測定対象となる API に関する情報としては、暗号化や復号機能を評価対象としている。なお、WolfCrypt における AES の実装は、利用できるハードウェアサポートの種類に応じて複数の種類が存在する。本評価においては、ディスク暗号化が主に利用されるマシン環境を想定し AES-NI を有効にするビルドを行った。

5.2.1.4 WolfCrypt におけるベンチマーク実行方法

WolfCrypt におけるベンチマークコマンドおよび本調査において利用したオプション（表 15）について説明を行う。

ベンチマークコマンド（WolfCrypt）

benchmark

表 15 本調査における利用オプション

項番	オプション	概要
1	指定なし	WolfCrypt の benchmark は利用オプションの指定は不要である。ベンチマークを実施することで、共通鍵暗号や公開鍵暗号、ハッシュ関数など 55 種類の測定項目に対して性能測定を実施できる

5.3 実装性能評価における評価対象項目と実装評価実施項目

「 実装評価対象一覧」で示した暗号利用モードと実装性能評価を行う暗号ライブラリにおいて実装されている暗号利用モードの関係を表 16 に整理する。本調査の暗号ライブラリに関する実装性能評価における測定対象となる。なお、表 16 で使用する記号の凡例は、「〇: 実装あり、×: 実装なし、△:問題あり」である。

表 16 より、実装性能評価で利用する暗号ライブラリで実装状況を考慮して、実装性能評価項目としては、項番 2、項番 5、項番 8 を中心に実装性能評価を行う。その他の項目については、各暗号利用モードにおける実装性能に関する性能傾向を知ることができる。

表 16 暗号ライブラリでの実装を考慮した測定対象の暗号利用モード

項番	表 12 での暗号利用モード	鍵長(bit)	OpenSSL	WolfCrypt
1	XTS	128	○	×
2		256	○	○
3	ECB	128	○	○
4		192	○	○
5		256	○	○
6	CBC	128	○	○
7		192	○	○
8		256	○	○
9	CFB	128	○	○
10		192	○	○
11		256	○	○
12	CTR	128	○	○
13		192	○	○
14		256	○	○
15	OFB	128	○	×
16		192	○	×
17		256	○	×
18	CCM	128	△	×
19		192	△	×
20		256	△	×
21	GCM	128	○	○
22		192	○	○
23		256	○	○

5.4　実装性能結果

実装性能結果として、表 16 でグレーによって塗りつぶされた項番に対して実装性能比較を行う。評価で利用した暗号ライブラリ毎に実装性能に関する測定結果を示している。

5.4.1.1　OpenSSL における実装性能結果

実行環境である 3 環境に対して、OpenSSL による AES-XTS（256bit）、AEC-ECB（256bit）および AES-CBC（256bit） での測定結果を図 8～図 12 および表 17～表 21 として示す。

＜Windows 環境＞

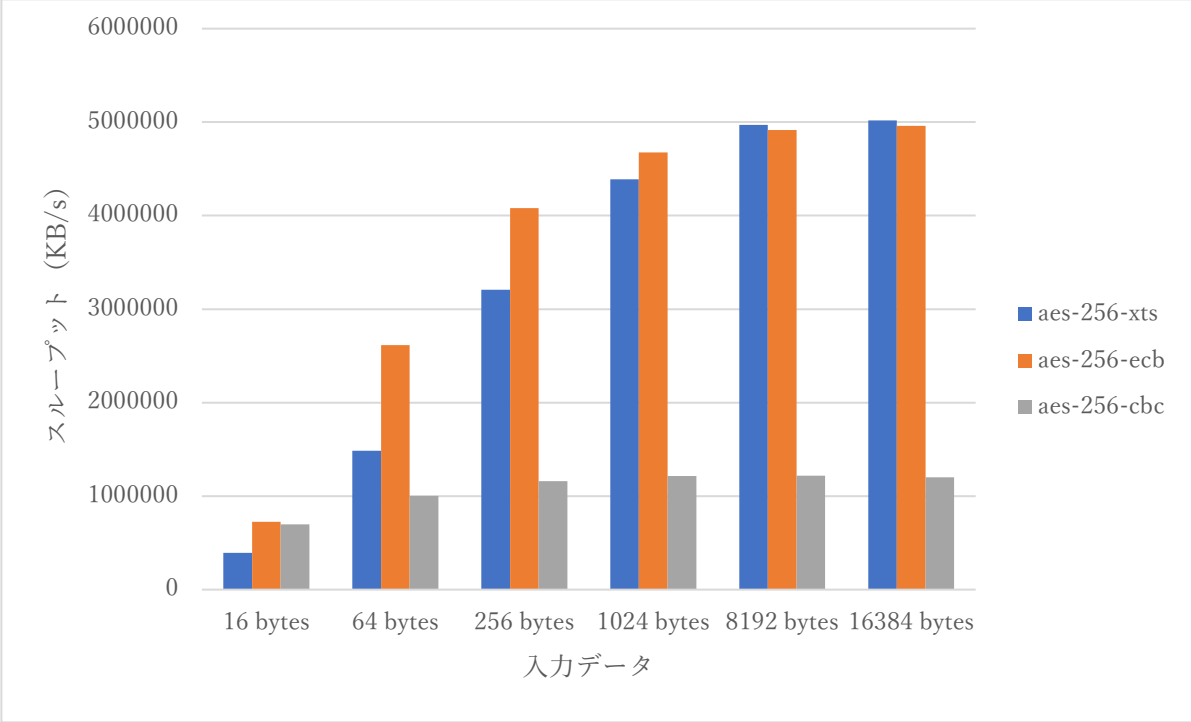


図 8　Windows 環境における実装性能結果（OpenSSL: 暗号化）

表 17　Windows 環境における実装性能結果（OpenSSL: 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	394505.55	1485969.69	3206756.95	4389973.33	4971151.36	5018834.26
aes-256-ecb	725685.11	2614959.79	4080193.11	4677160.96	4914577.92	4961577.64
aes-256-cbc	699745.21	1001478.6	1159585.19	1214693.03	1217844.57	1200974.51

（単位は KB/s）

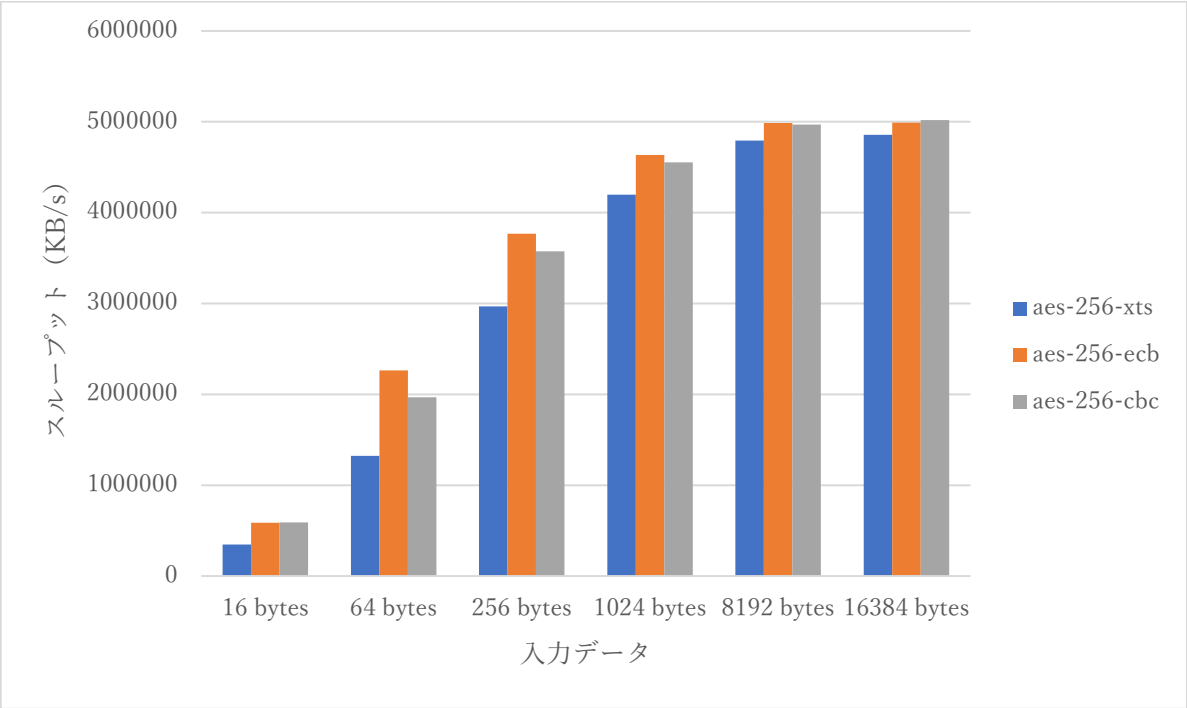


図 9　Windows 環境における実装性能結果（OpenSSL: 復号）

表 18　Windows 環境における実装性能結果（OpenSSL: 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	348375.03	1322908.18	2969041.83	4198649.86	4794376.19	4855540.39
aes-256-ecb	585959.97	2264927.85	3769427.54	4635617.28	4986170.03	4992002.73
aes-256-cbc	589592.83	1969084.55	3575207.34	4552853.16	4969368.23	5018932.57

（単位は KB/s）

＜macOS 環境＞

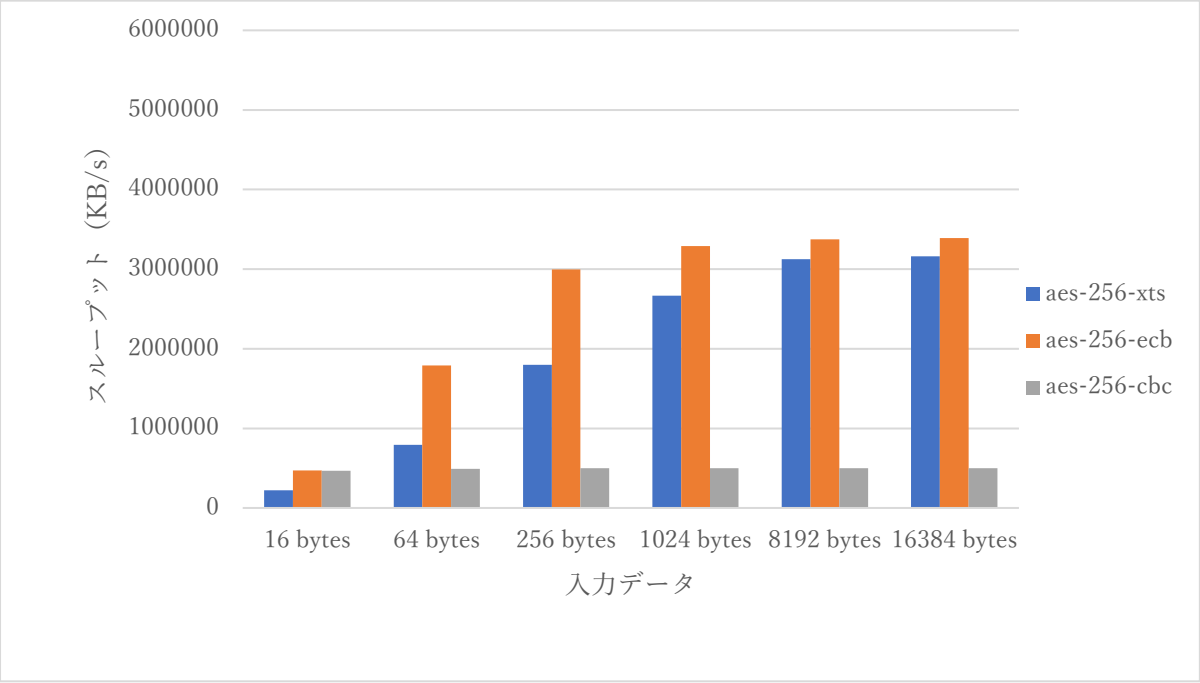


図 10　macOS 環境における実装性能結果（OpenSSL: 暗号化）

表 19　macOS 環境における実装性能結果（OpenSSL: 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	223888.69	791039.38	1799060.33	2668685.65	3125947.05	3159619.02
aes-256-ecb	470124.41	1788392.13	2995652.01	3291116.61	3372442.28	3389500.07
aes-256-cbc	466230.33	491065.47	497611.35	498652.16	500989.95	498715.90

（単位は KB/s）

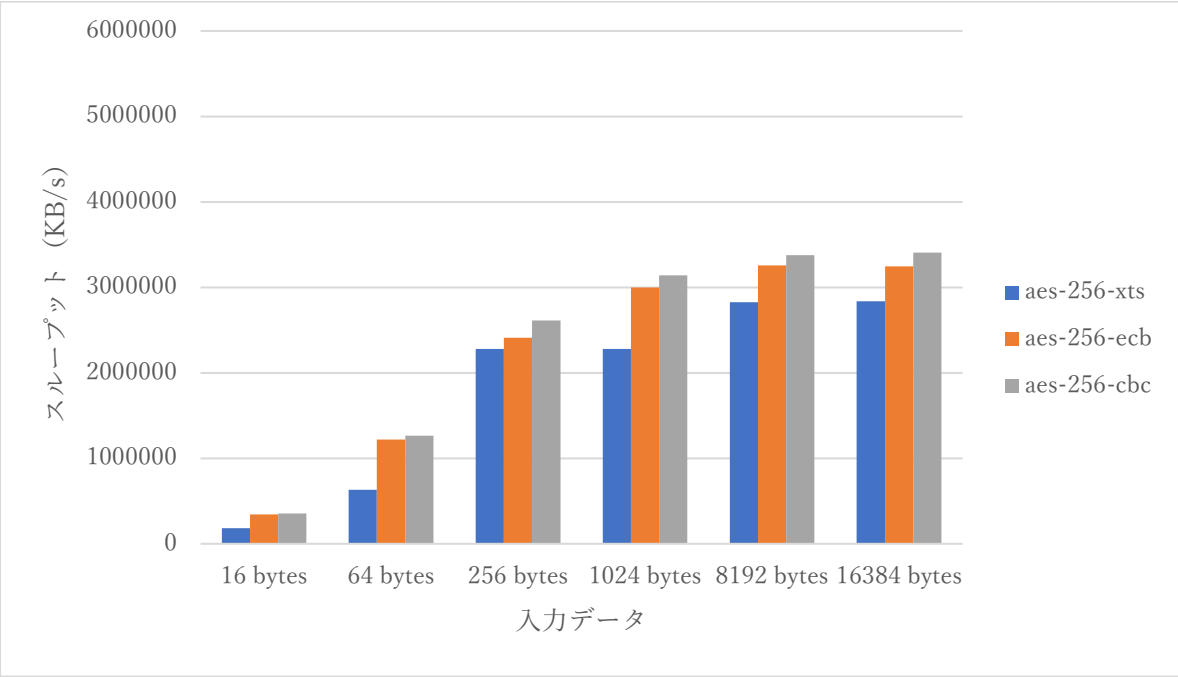


図 11　macOS 環境における実装性能結果（OpenSSL: 復号）

表 20　macOS 環境における実装性能結果（OpenSSL: 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	181981.5	633080.13	2280764.79	2280764.79	2825978.85	2836763.99
aes-256-ecb	345168.62	1218058.71	2409308.86	2999994.93	3256096.58	3247314.28
aes-256-cbc	353670.02	1263651.66	2614263.61	3140111.02	3376723.22	3406025.38

（単位は KB/s）

＜Linux 環境＞

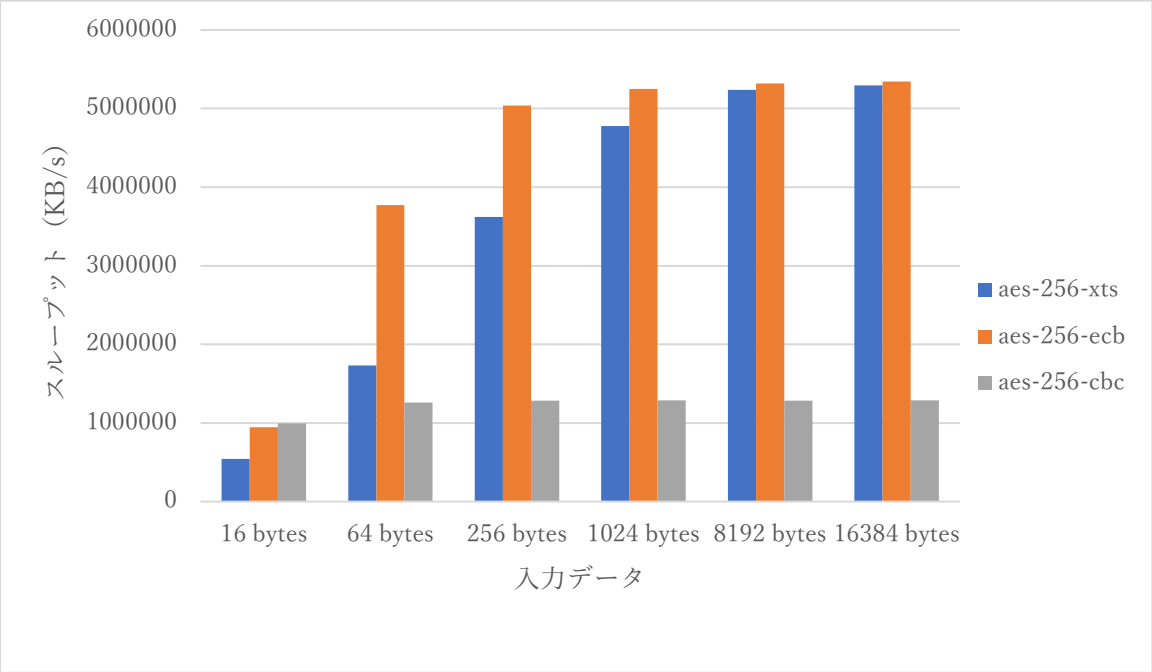


図 12 Linux 環境における実装性能結果（OpenSSL: 暗号化）

表 21　Linux 環境における実装性能結果（OpenSSL: 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	543179.57	1732118.38	3621865.56	4778003.8	5237033.64	5296936.28
aes-256-ecb	944778.55	3770500.5	5040330.84	5248844.8	5318967.3	5345012.39
aes-256-cbc	993300.81	1260529.54	1283763.2	1288221.7	1283825.66	1287591.25

（単位は KB/s）

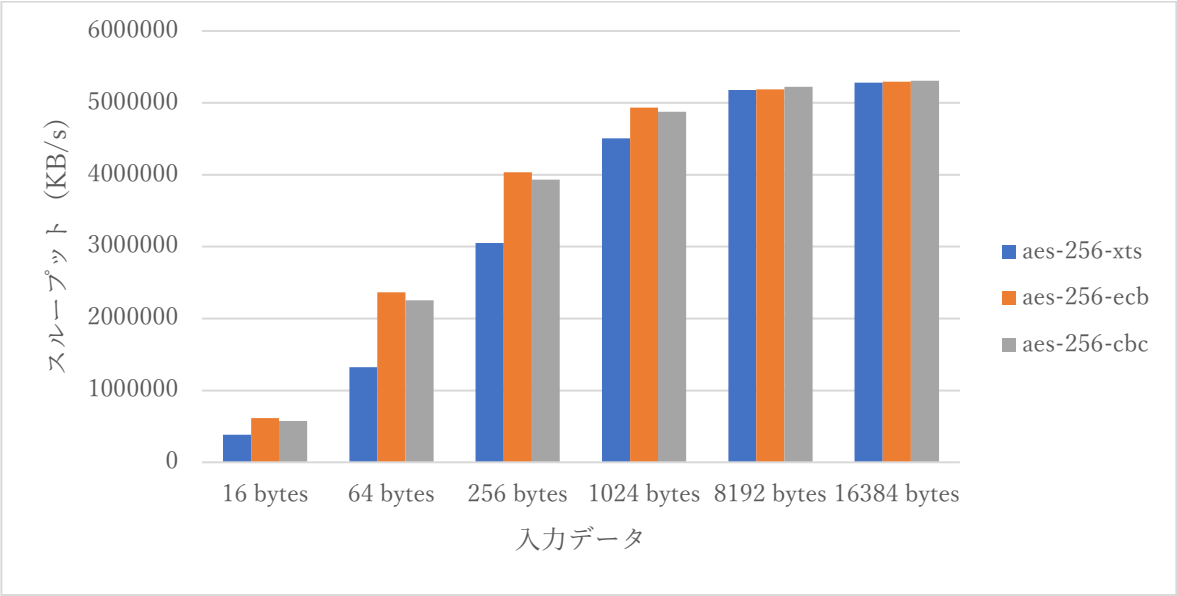


図 13　Linux 環境における実装性能結果（OpenSSL: 復号）

表 22　Linux 環境における実装性能結果（OpenSSL: 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	384665.19	1324694.98	3050718.81	4504600.58	5176423.77	5278963.03
aes-256-ecb	616197.9	2363642.99	4035974.23	4933271.55	5187674.11	5292894.89
aes-256-cbc	575186.49	2252488.51	3929712.13	4874141.35	5223746.22	5305805.48

（単位は KB/s）

また、OpenSSL における 3 環境での実装性能について、暗号化については図 14 および表 23 に整理する。また、復号については図 15 および表 24 に整理する。

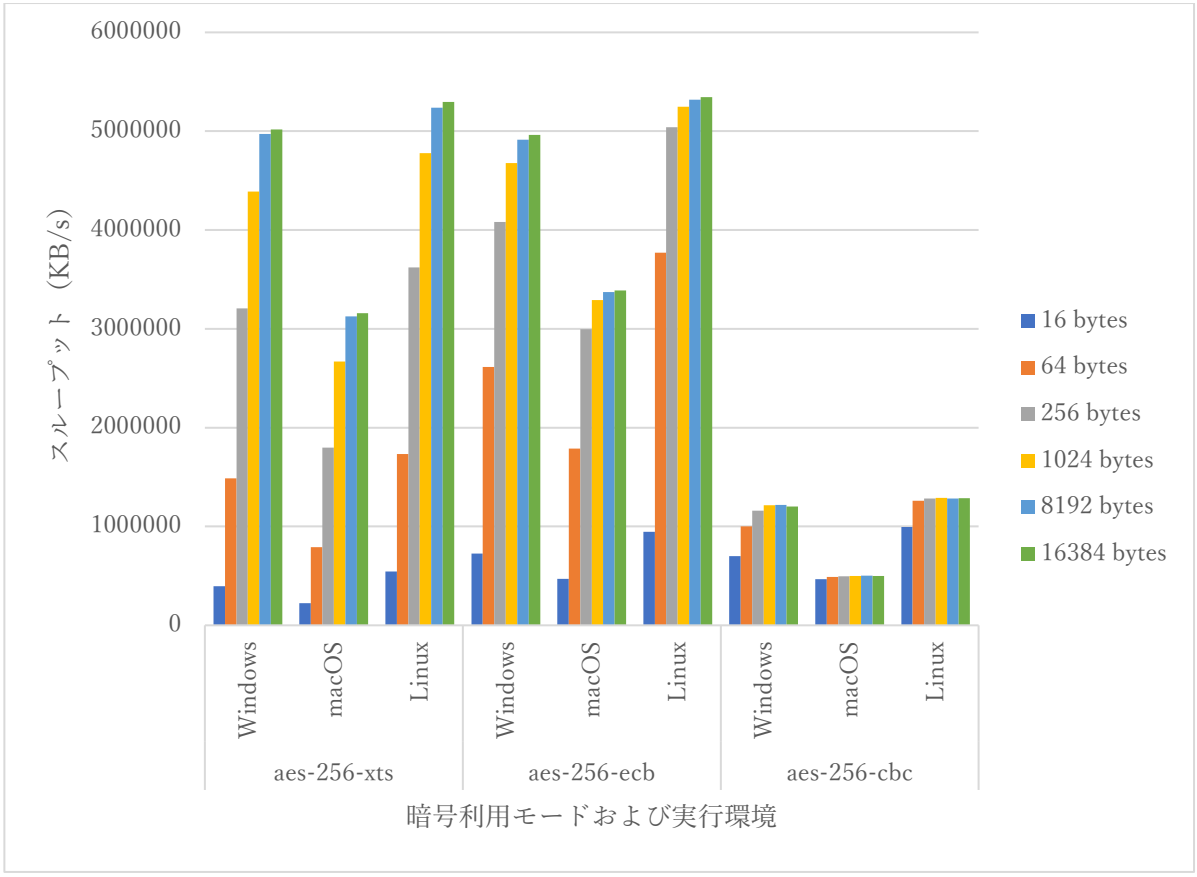


図 14　OpenSSL における 3 環境での実装性能結果一覧（暗号化）

表 23　OpenSSL における 3 環境での実装性能結果一覧（暗号化）

	aes-256-xts			aes-256-ecb			aes-256-cbc		
	Windows	macOS	Linux	Windows	macOS	Linux	Windows	macOS	Linux
16 bytes	394505.55	223888.69	543179.57	725685.11	470124.41	944778.55	699745.21	466230.33	993300.81
64 bytes	1485969.69	791039.38	1732118.38	2614959.79	1788392.13	3770500.5	1001478.6	491065.47	1260529.54
256 bytes	3206756.95	1799060.33	3621865.56	4080193.11	2995652.01	5040330.84	1159585.19	497611.35	1283763.2
1024 bytes	4389973.33	2668685.65	4778003.8	4677160.96	3291116.61	5248844.8	1214693.03	498652.16	1288221.7
8192 bytes	4971151.36	3125947.05	5237033.64	4914577.92	3372442.28	5318967.3	1217844.57	500989.95	1283825.66
16384 bytes	5018834.26	3159619.02	5296936.28	4961577.64	3389500.07	5345012.39	1200974.51	498715.90	1287591.25

（単位は KB/s）

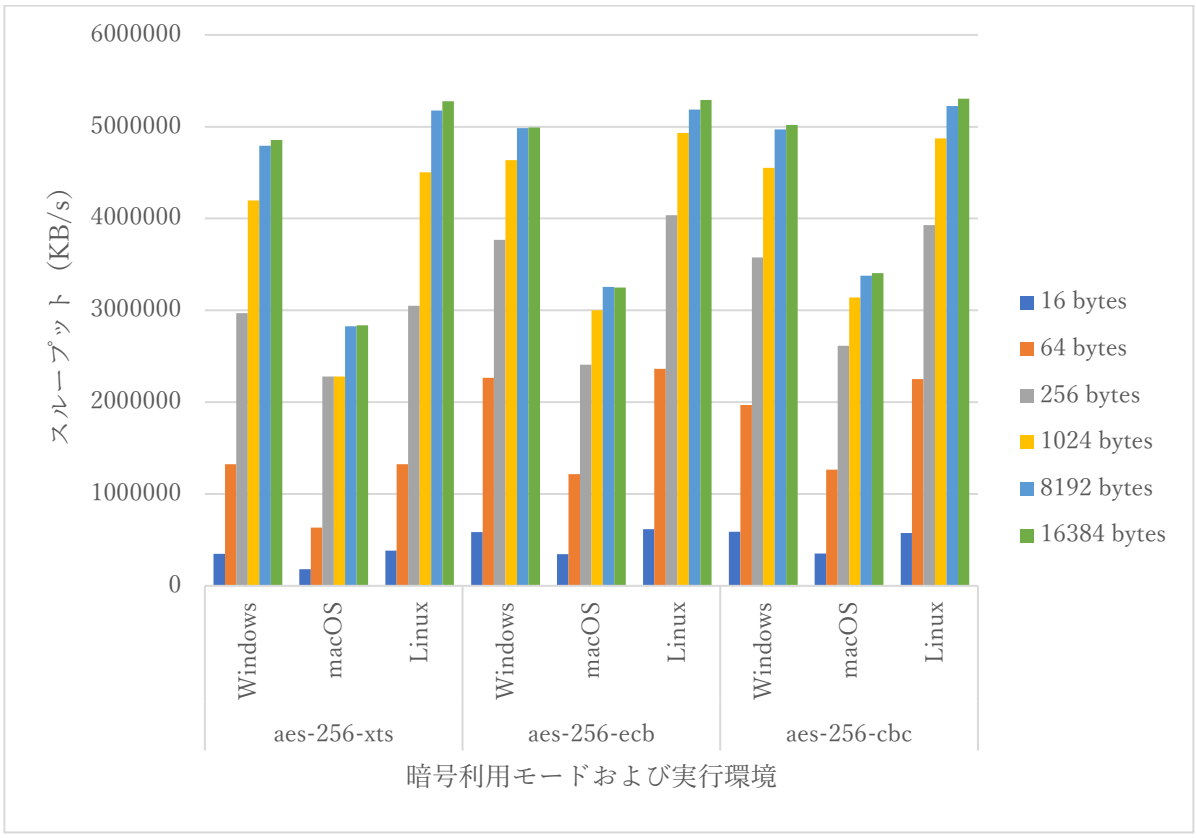


図 15　OpenSSL における 3 環境での実装性能結果一覧（復号）

表 24　OpenSSL における 3 環境での実装性能結果一覧（復号）

	aes-256-xts			aes-256-ecb			aes-256-cbc		
	Windows	macOS	Linux	Windows	macOS	Linux	Windows	macOS	Linux
16 bytes	348375.03	181981.5	384665.19	585959.97	345168.62	616197.9	589592.83	353670.02	575186.49
64 bytes	1322908.18	633080.13	1324694.98	2264927.85	1218058.71	2363642.99	1969084.55	1263651.66	2252488.51
256 bytes	2969041.83	2280764.79	3050718.81	3769427.54	2409308.86	4035974.23	3575207.34	2614263.61	3929712.13
1024 bytes	4198649.86	2280764.79	4504600.58	4635617.28	2999994.93	4933271.55	4552853.16	3140111.02	4874141.35
8192 bytes	4794376.19	2825978.85	5176423.77	4986170.03	3256096.58	5187674.11	4969368.23	3376723.22	5223746.22
16384 bytes	4855540.39	2836763.99	5278963.03	4992002.73	3247314.28	5292894.89	5018932.57	3406025.38	5305805.48

（単位は KB/s）

5.4.1.2 WolfCrypt における実装性能結果

実行環境である 3 環境に対して、WolfCrypt による AES-XTS（256bit）、AES-ECB（256bit）および AES-CBC（256bit） での測定結果を図 16 ～図 21 および表 25 ～表 30 として整理する。

＜Windows 環境＞

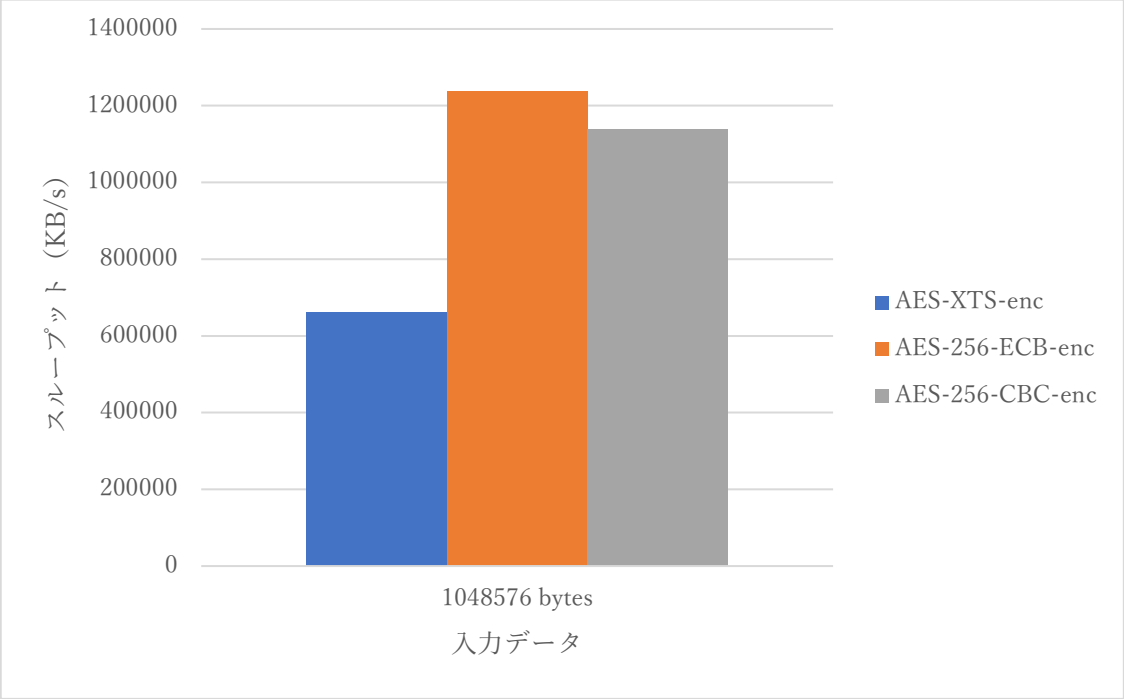


図 16 Windows 環境における実装性能結果（WolfCrypt: 暗号化）

表 25 Windows 環境における実装性能結果（WolfCrypt: 暗号化）

	1048576 bytes
AES-XTS-enc	661023
AES-256-ECB-enc	1238166
AES-256-CBC-enc	1136883

（単位は KB/s）

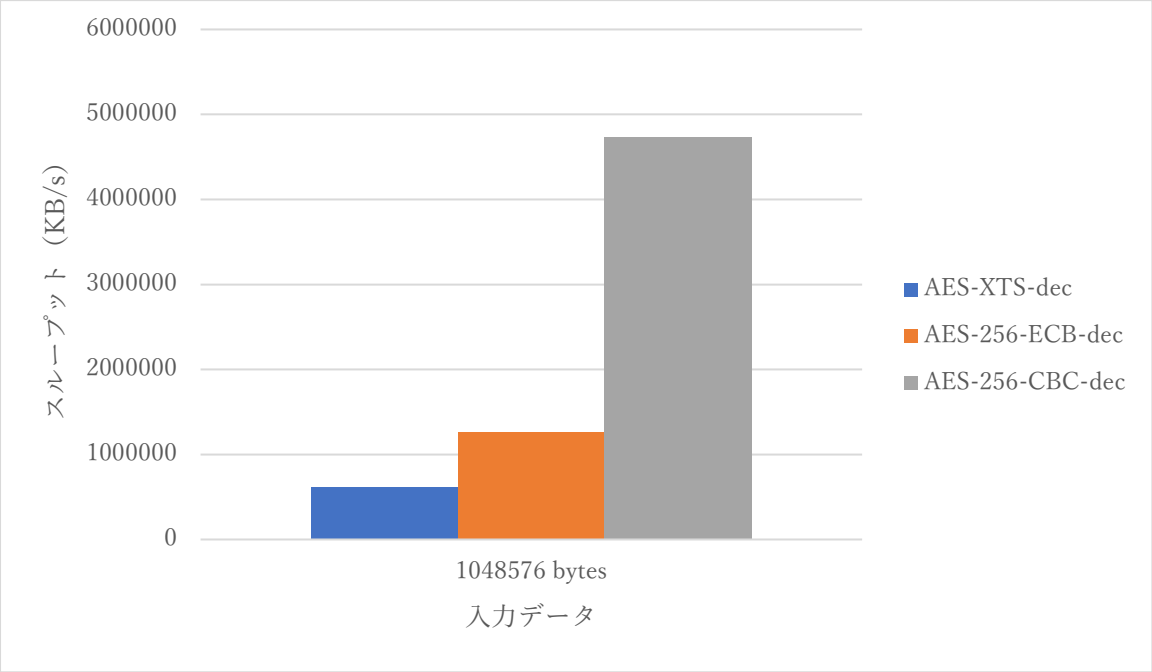


図 17　Windows 環境における実装性能結果（WolfCrypt: 復号）

表 26　Windows 環境における実装性能結果（WolfCrypt: 復号）

	1048576 bytes
AES-XTS-dec	614010
AES-256-ECB-dec	1264893
AES-256-CBC-dec	4728723

（単位は KB/s）

＜macOS 環境＞

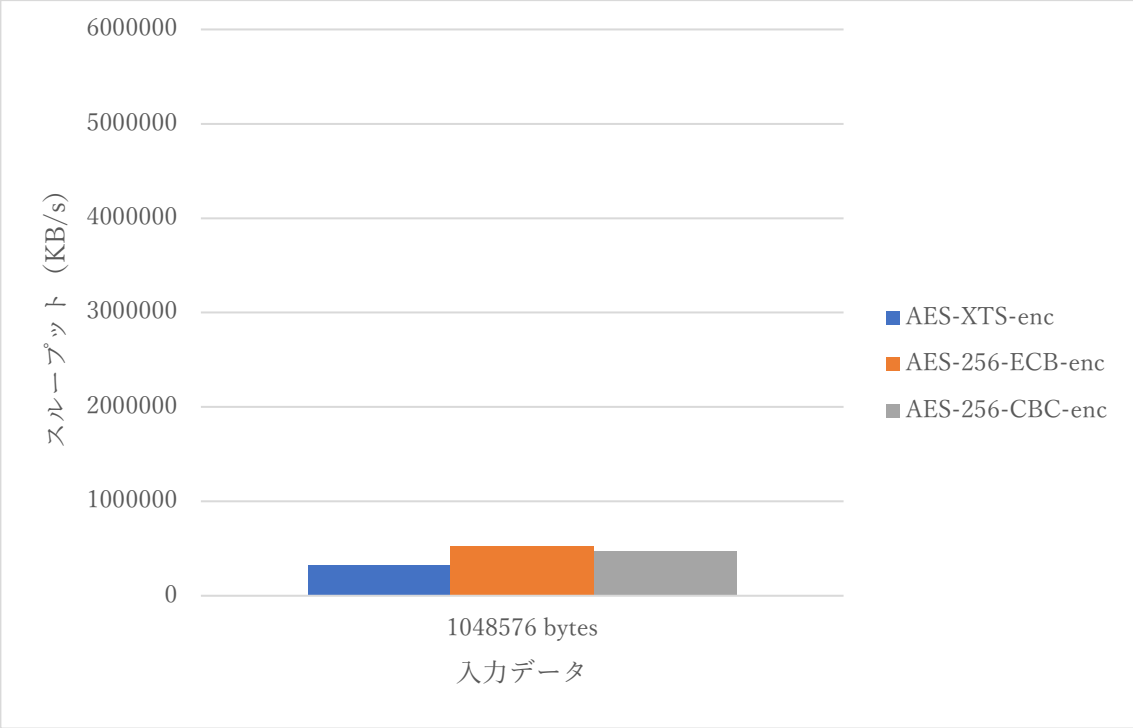


図 18　macOS 環境における実装性能結果（WolfCrypt: 暗号化）

表 27　macOS 環境における実装性能結果（WolfCrypt: 暗号化）

	1048576 bytes
AES-XTS-enc	326383
AES-256-ECB-enc	528321
AES-256-CBC-enc	469786

（単位は KB/s）

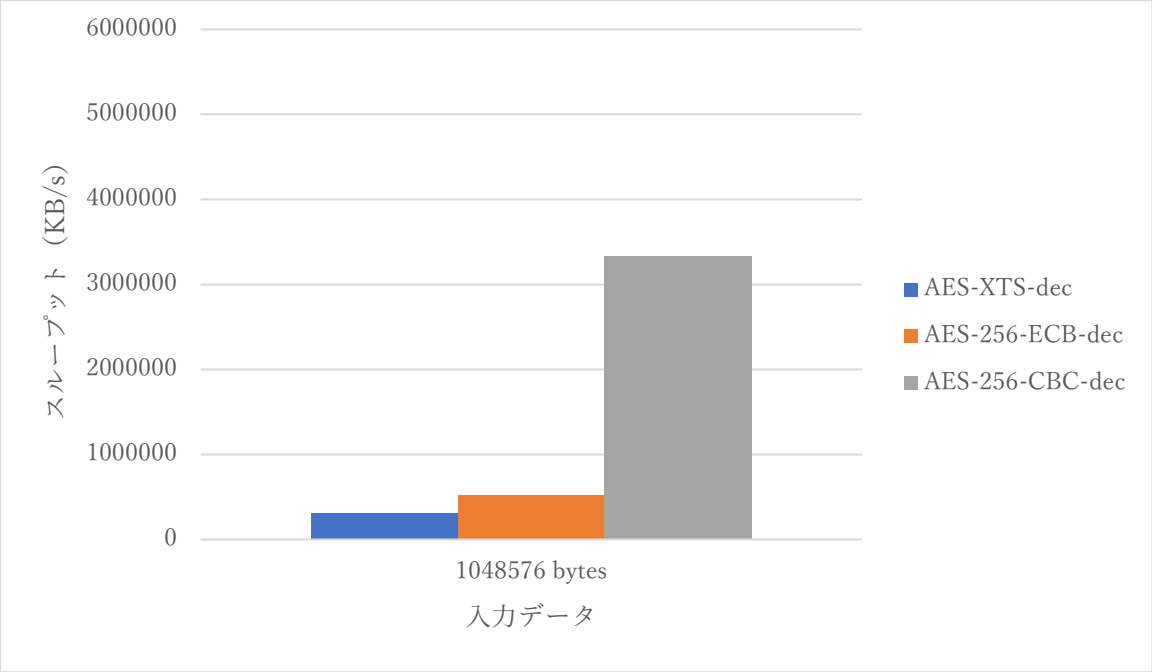


図 19　macOS 環境における実装性能結果（WolfCrypt: 復号）

表 28　macOS 環境における実装性能結果（WolfCrypt: 復号）

	1048576 bytes
AES-XTS-dec	312059
AES-256-ECB-dec	526088
AES-256-CBC-dec	3335104

（単位は KB/s）

＜Linux 環境＞

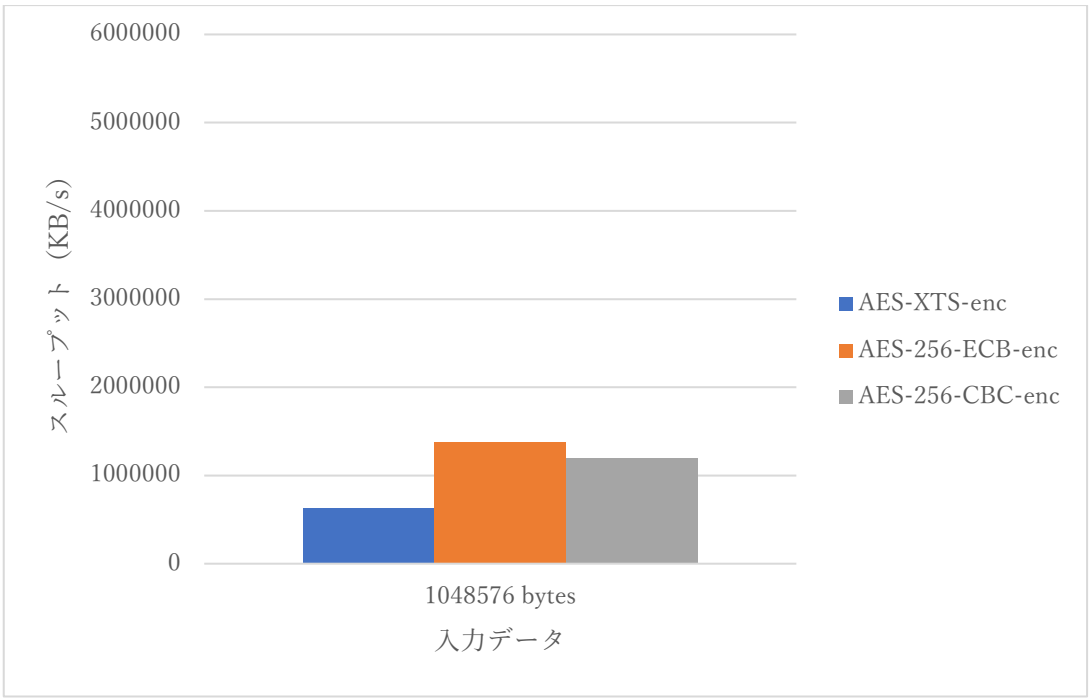


図 20　Linux 環境における実装性能結果（WolfCrypt: 暗号化）

表 29 Linux 環境における実装性能結果（WolfCrypt: 暗号化）

	1048576 bytes
AES-XTS-enc	630823
AES-256-ECB-enc	1381395
AES-256-CBC-enc	1197638

（単位は KB/s）

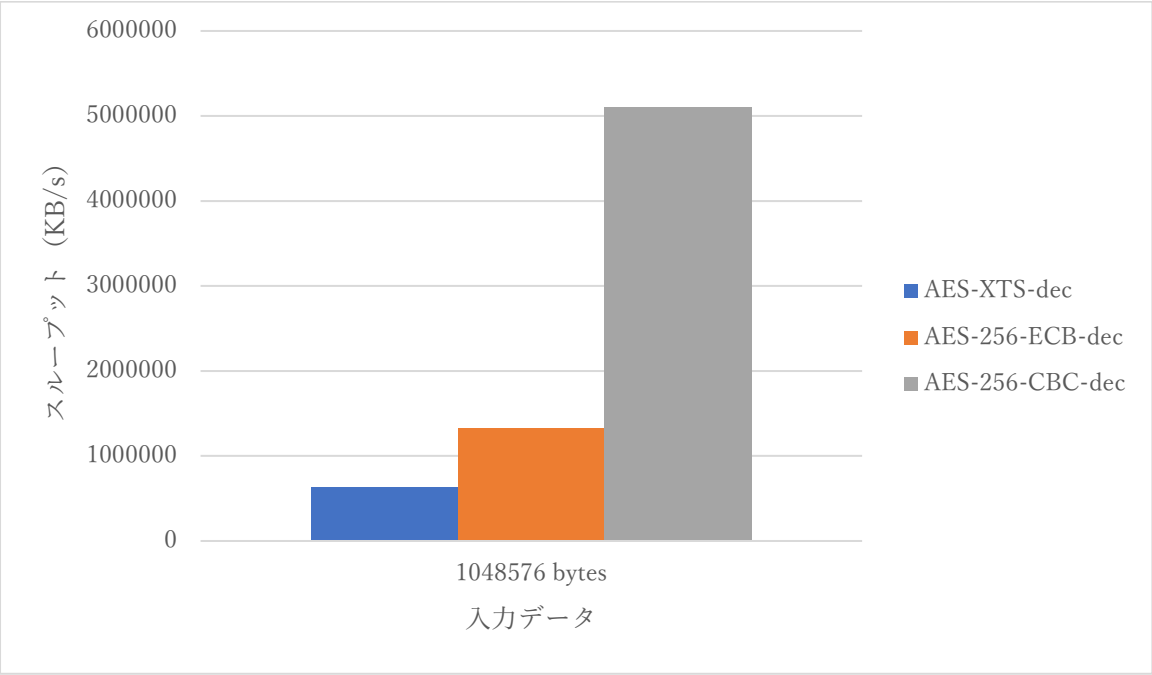


図 21　Linux 環境における実装性能結果（WolfCrypt: 復号）

表 30　Linux 環境における実装性能結果（WolfCrypt: 復号）

	1048576 bytes
AES-XTS-dec	630177
AES-256-ECB-dec	1327715
AES-256-CBC-dec	5100741

（単位は KB/s）

また、WolfCrypt における 3 環境での実装性能について、暗号化については**図 22** および**表 31** に整理する。また、復号については**図 23** および**表 32** に整理する。

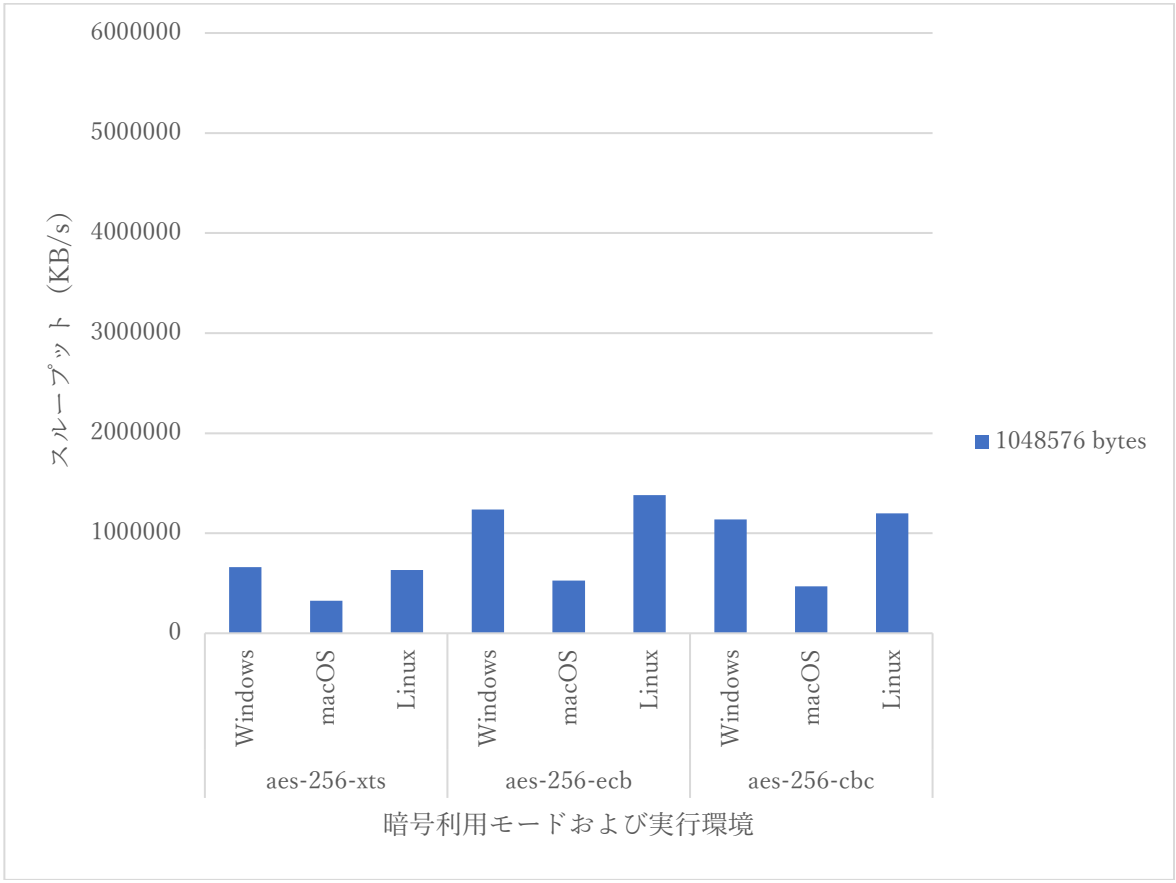


図 22　WolfCrypt における 3 環境での実装性能結果一覧（暗号化）

表 31　WolfCrypt における 3 環境での実装性能結果一覧（暗号化）

	aes-256-xts			aes-256-ecb			aes-256-cbc		
	Windows	macOS	Linux	Windows	macOS	Linux	Windows	macOS	Linux
1048576 bytes	661023	326383	630823	1238166	528321	1381395	1136883	469786	1197638

（単位は KB/s）

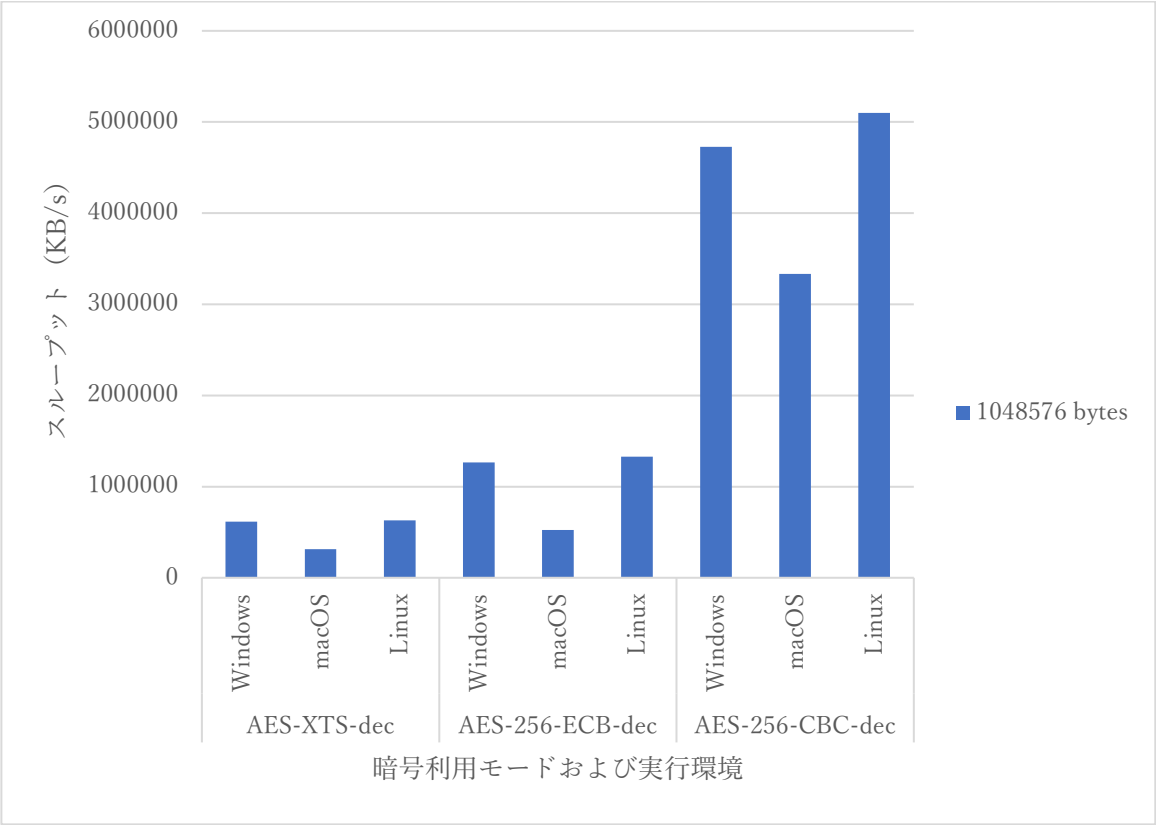


図 23　WolfCrypt における 3 環境での実装性能結果一覧（復号）

表 32　WolfCrypt における 3 環境での実装性能結果一覧（復号）

	AES-XTS-dec			AES-256-ECB-dec			AES-256-CBC-dec		
	Windows	macOS	Linux	Windows	macOS	Linux	Windows	macOS	Linux
1048576 bytes	614010	312059	630177	1264893	526088	1327715	4728723	3335104	5100741

（単位は KB/s）

5.5　実装性能評価における考察

本章では、実装性能評価として OpenSSL および WolfCrypt を用いた測定を実施した。これらの測定結果より、OpenSSL では Windows 環境、macOS 環境および Linux 環境において、測定性能の基準と考えられる ECB や広く利用されている暗号利用モードである CBC と比較しても XTS は遜色のないパフォーマンスを出していると考えられる。また、WolfCrypt においては、XTS と比較対象である ECB および CBC を比較すると、OpenSSL と異なる性能傾向となっていたが、これは WolfCrypt において ECB および CBC については AES-NI を利用している拡張命令セットが利用されているが、XTS においては AES-NI の拡張命令セットが限定的な利用がなされていることでパフォーマンスに差があるのではないかと考えられる。

これらの 2 つの OSS で公開されている暗号ライブラリを用いた実装性能測定結果を踏まえると、暗号利用モードに起因する要因で性能への影響があるというより、ソフトウェアの実装時に AES-NI のようなハードウェアサポートを利用することで、ストレージ暗号化などの利用用途に影響のない実装性能が得られることが期待される。

6　ディスク暗号化製品を使った実装性能比較調査

前章の「5 他の暗号利用モードとの実装性能比較調査」において、暗号ライブラリでの XTS に関する実装性能評価を行った。本章では、XTS 本来の利用方法に注目し、ディスク暗号化としての実装性能調査を実施する。

6.1　実装性能評価で選定した製品

ディスク暗号化機能を有する製品として、Microsoft BitLocker や Apple FileVault など実際の OS に搭載されている製品があるが、各 OS で実装性能比較を行うことができないため、本調査では様々な環境で動作可能であること、また、測定結果の裏付けを取るためにソースコードによる確認ができるよう OSS 製品から選定することとした。OSS の中でディスク暗号化を実現する製品として、VeraCrypt[####]、AES Crypt[§§§§]、Cryptomator[*****]など、多くの製品が公開されているが、本調査では Windows/Mac/Linux の 3 環境すべてに対応していること、ディスク暗号化に関する実装性能を測定する公式機能が実装されている点に注目して「VeraCrypt」を選定した。

今回、選定した VeraCrypt は、暗号化ユーティリティソフトウェアとして有名な TrueCrypt をフォークして開発された OSS である。VeraCrypt の機能としてはファイルやパーティションの暗号化やストレージ全体の暗号化など実現することができる。初版リリースは、2013 年 6 月 22 日であり、継続的にメンテナンスが行われバージョンアップが行われている。また、VeraCrypt が対応している暗号アルゴリズムおよび暗号利用モードを表 33 に示す。

表 33　VeraCrypt で利用可能な暗号アルゴリズムおよび暗号利用モード

項番	暗号アルゴリズム名	暗号利用モード
1	AES	XTS
2	Camellia	
3	Serpent	
4	Twofish	
5	Kuznyechik（GOST R 34.12-2015）	

本調査対象として、表 33 で対応している暗号アルゴリズムに関する性能比較を行うことはせずに、CRYPTREC 暗号リストの電子政府推奨暗号リストの 128 ビットブロック暗号として採択されている AES および Camellia に注目して性能測定を実施する。

本調査で採用した VeraCrypt については、表 34 に示した取得先 URL から Windows、macOS、Linux それぞれのソフトウェアを取得できる。本調査で利用した VeraCrypt のバージョンを以下に示す。

[####] https://www.veracrypt.fr/en/Home.html

[§§§§] https://www.aescrypt.com/

[*****] https://cryptomator.org/

表 34　評価で利用したソフトウェア

項番	ストレージ暗号化製品名	バージョン	取得先 URL
1	VeraCrypt	1.24-Update3	https://www.veracrypt.fr/en/Downloads.html

6.2　実装性能評価に向けた準備

実装性能評価を実施する上での準備として、本調査で利用した「実装性能評価を行った環境」および「暗号ライブラリのビルド方法」を示す。

6.2.1　実装性能評価を行った環境

XTS で期待される利用がディスク暗号化であることを考慮し、主な OS である Windows、macOS、Linux の 3 環境を対象とし、実環境での性能評価を実施した。それぞれの環境に関するスペックは以下のとおりである。なお、このマシン環境は、5.1 実装性能評価を行った環境と同一環境である。

＜Windows 環境＞
OS: Windows 10 Pro 64-bit Version 1909 (Build 18363.535)
CPU: Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz (12 CPUs)
Memory: 64GB

＜macOS 環境＞
OS: macOS Catalina 10.15.2
CPU: 2.4GHz dual-core Intel Core i5 processor
Memory: 16GB

＜Linux 環境＞
OS: Ubuntu 18.04.3 LTS
CPU: Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
Memory: 32GB

6.2.2 暗号ライブラリのビルド方法

VeraCrypt のソースコードは 公式サイト [ttttt]から入手可能である。本測定では、ソースコードからビルドして測定を試みると Windows 環境においてコード署名を要求されるためビルドを実施できなかった。実装性能測定の目的を想定して同一条件での性能測定を行うために公式サイトが提供しているインストーラーからインストールし、GUI 上でベンチマークを実施した。なお、実装性能に関する測定条件の調査や性能について実装を確認する際には、公式サイトで公開されているソースコードを解析した。

6.3 実装性能評価に向けた測定

実装性能評価対象である VeraCrypt で実装されているベンチマーク機能として、benchmark がある。ここでは、ベンチマーク機能における測定の観点および測定対象となっている API について情報を示す。また、ベンチマーク機能に関する実行方法について情報を示す。

6.3.1 VeraCrypt における測定の観点

VeraCrypt で実装されているベンチマーク機能である Algorithms Benchmark は、今回評価対象である XTS のみの実装である。また、その測定結果は、バッファサイズとして環境などで選択できるサイズは異なるが、1024KB や 50MB、1GB など複数の選択肢から選択でき、暗号化/復号処理に関するスループットを測定が可能である。なお、ベンチマークおよび実際に利用される暗号処理で用いられる src/Crypto/Aes.h において、AES_256 フラグのみが有効になっていることから鍵長は 256 ビットであった。

6.3.2 VeraCrypt におけるベンチマーク実行方法

VeraCrypt におけるベンチマークに関する概要について説明を行う。なお、macOS 環境を例示する。
表 34 で示した取得先 URL からダウンロードしたソフトウェアをインストールして起動後、メニューバーにある Tool から今回利用する benchmark を起動できる。（図 24）

[ttttt] https://www.veracrypt.fr/code/VeraCrypt/

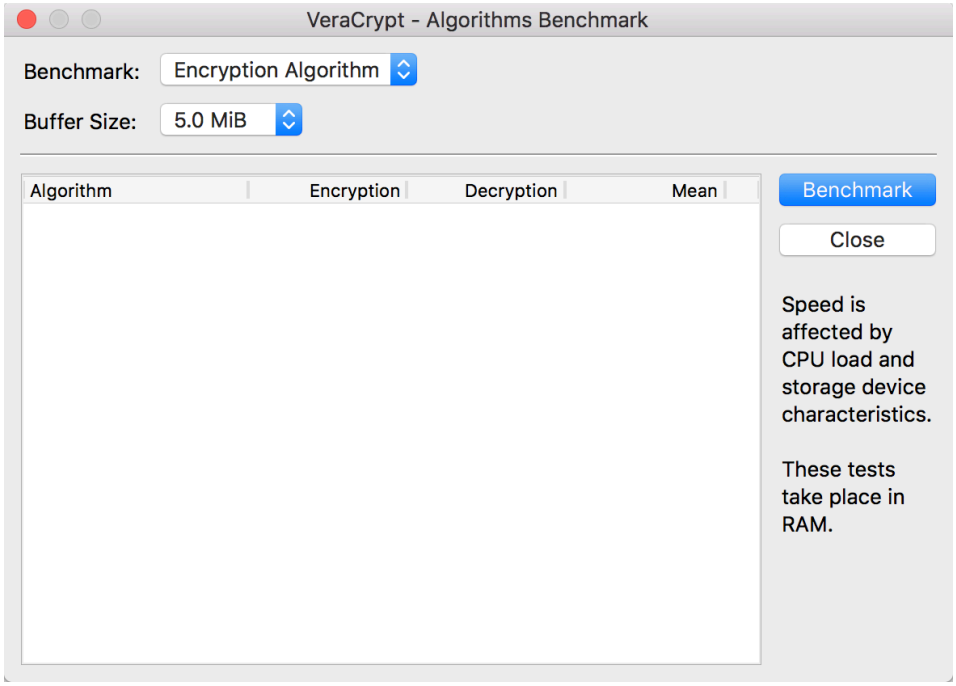


図 24　VeraCrypt Algorithms Benchmark

このベンチマークによって提供されている機能は、ディスク暗号化で利用可能な共通鍵暗号に関する性能評価（Encryption Algorithm）と PKCS#5 での PBKDF2 に関する性能評価（PKCS-5 PRF）、ハッシュ関数に関する性能評価（Hash Algorithm）を行うことができる。（図 25）

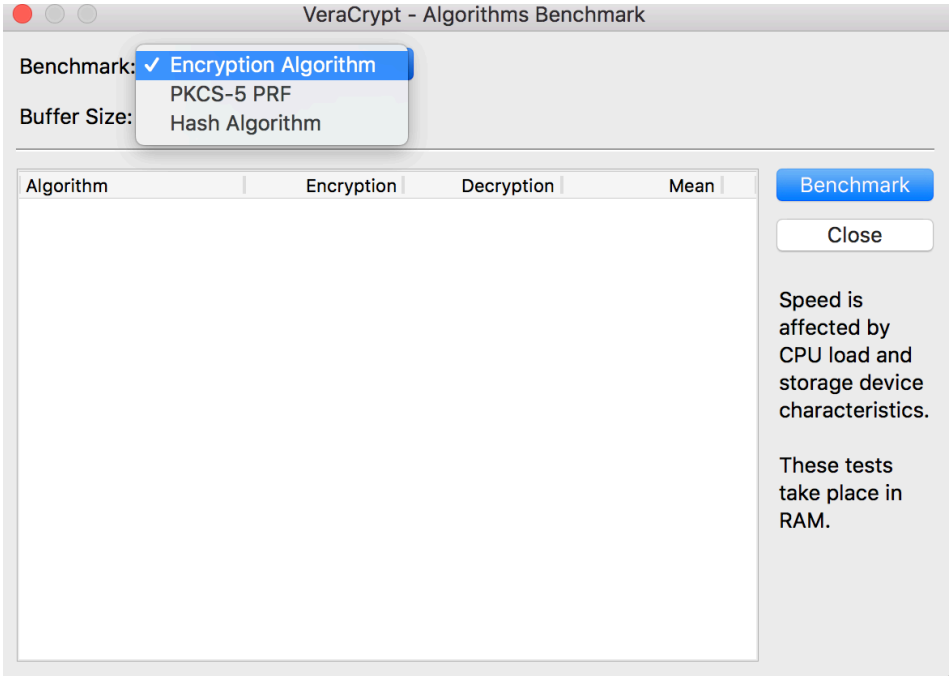


図 25　Benchmark 設定項目

6.4 VeraCrypt を用いた実装性能測定

6.4.1 ベンチマークによる性能測定

実行環境である 3 環境に対して、VeraCrypt のベンチマークで測定可能な AES-XTS（256bit）、Camellia-XTS（256bit）に関する測定結果（図 26、図 27、図 28）について示す。

＜Windows 環境＞

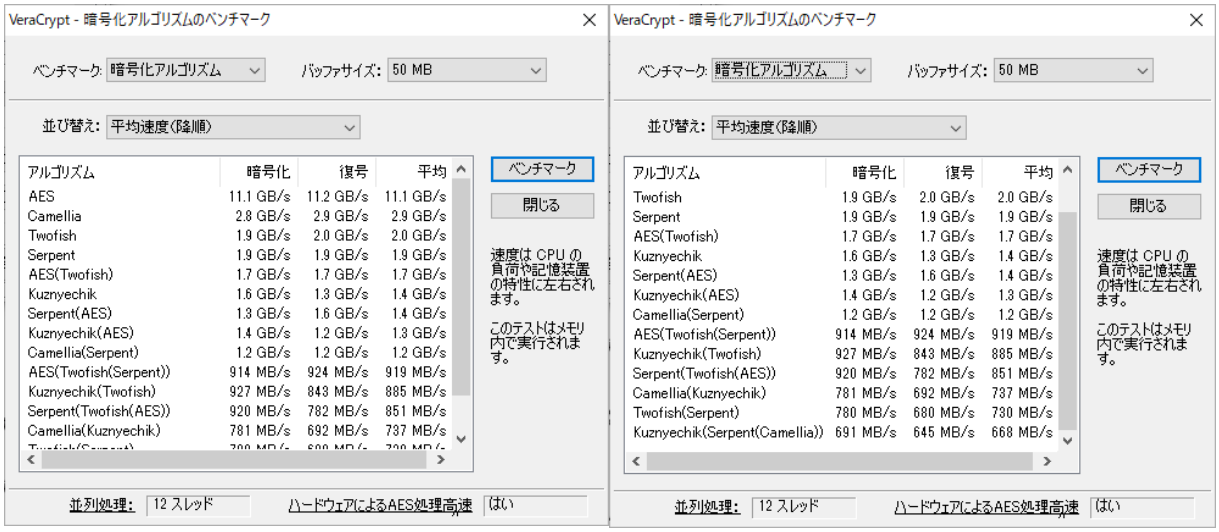


図 26 VeraCrypt ベンチマーク（Windows 環境）

＜macOS 環境＞

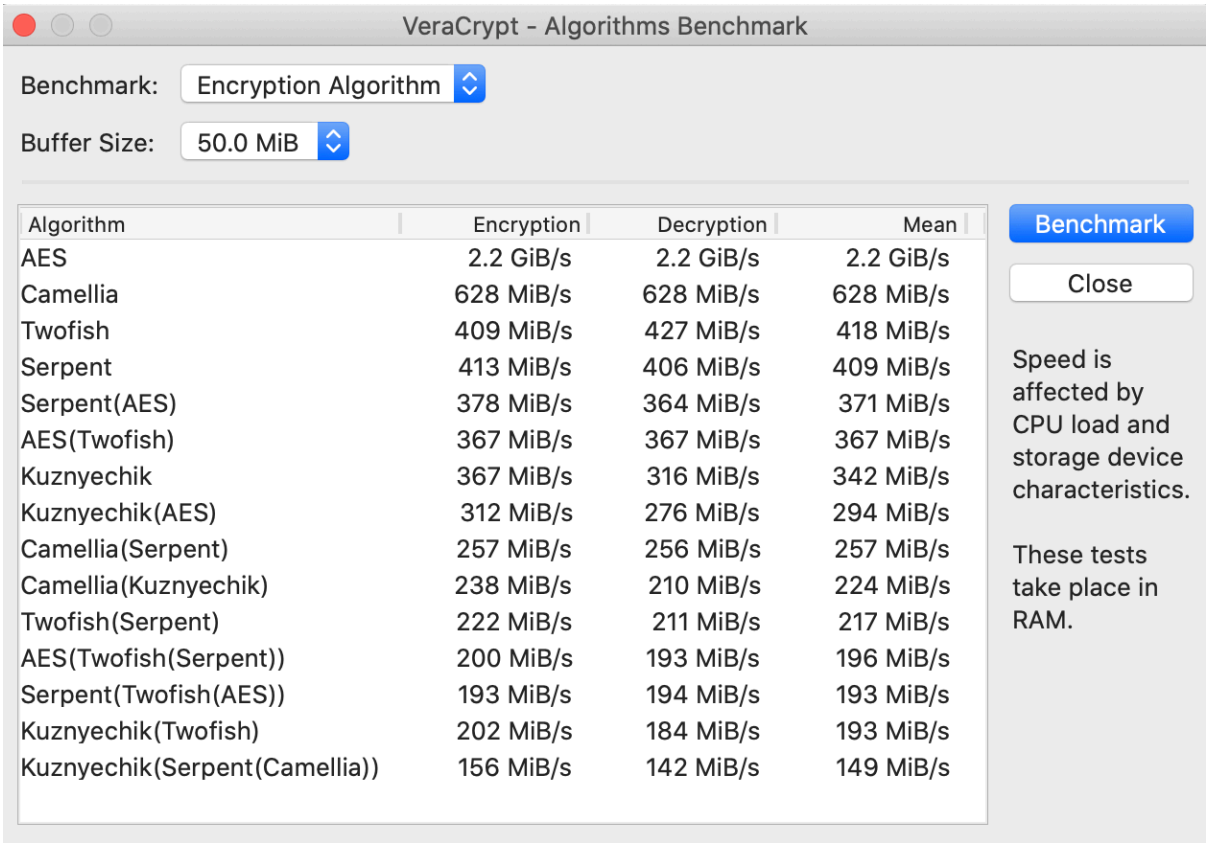
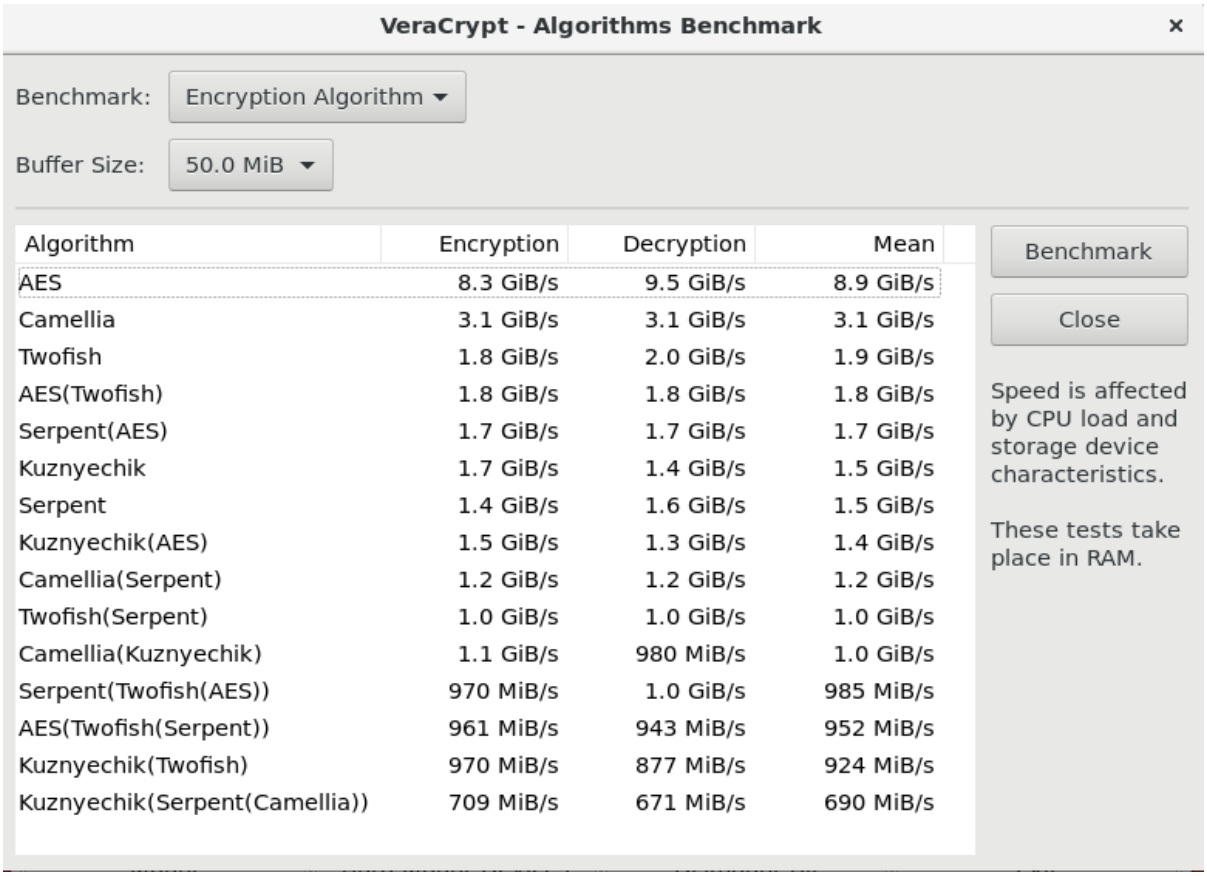


図 27 VeraCrypt ベンチマーク（macOS 環境）

＜Linux 環境＞



Algorithm	Encryption	Decryption	Mean
AES	8.3 GiB/s	9.5 GiB/s	8.9 GiB/s
Camellia	3.1 GiB/s	3.1 GiB/s	3.1 GiB/s
Twofish	1.8 GiB/s	2.0 GiB/s	1.9 GiB/s
AES(Twofish)	1.8 GiB/s	1.8 GiB/s	1.8 GiB/s
Serpent(AES)	1.7 GiB/s	1.7 GiB/s	1.7 GiB/s
Kuznyechik	1.7 GiB/s	1.4 GiB/s	1.5 GiB/s
Serpent	1.4 GiB/s	1.6 GiB/s	1.5 GiB/s
Kuznyechik(AES)	1.5 GiB/s	1.3 GiB/s	1.4 GiB/s
Camellia(Serpent)	1.2 GiB/s	1.2 GiB/s	1.2 GiB/s
Twofish(Serpent)	1.0 GiB/s	1.0 GiB/s	1.0 GiB/s
Camellia(Kuznyechik)	1.1 GiB/s	980 MiB/s	1.0 GiB/s
Serpent(Twofish(AES))	970 MiB/s	1.0 GiB/s	985 MiB/s
AES(Twofish(Serpent))	961 MiB/s	943 MiB/s	952 MiB/s
Kuznyechik(Twofish)	970 MiB/s	877 MiB/s	924 MiB/s
Kuznyechik(Serpent(Camellia))	709 MiB/s	671 MiB/s	690 MiB/s

図 28 VeraCrypt ベンチマーク(Linux 環境)

VeraCrypt を用いた実装性能測定の結果を、AES-XTS および Camellia-XTS における 3 環境での暗号化に関する測定結果（図 29、表 35）を整理する。

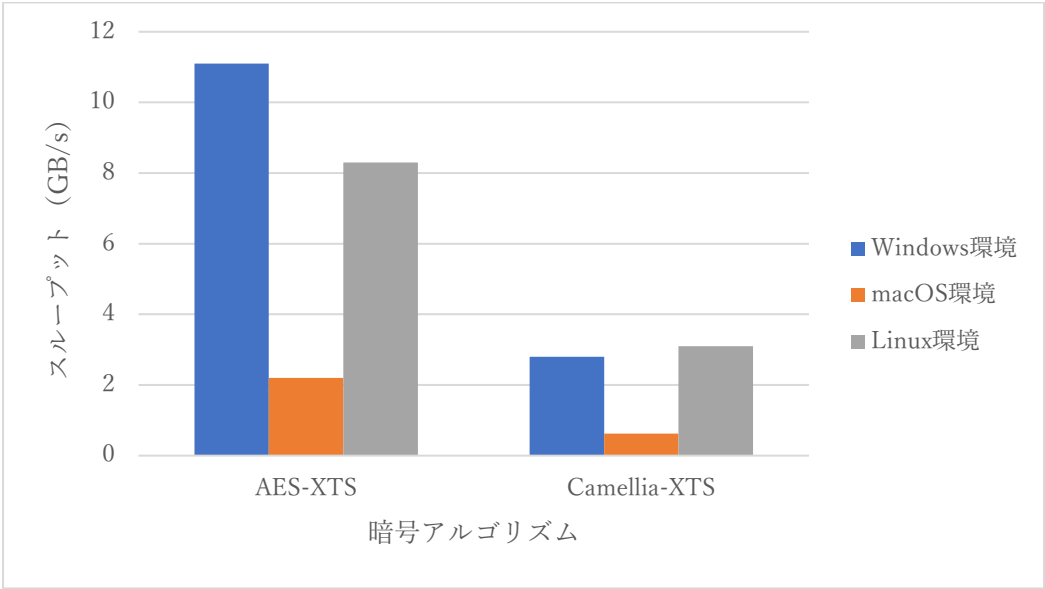


図 29 VeraCrypt 実装性能比較

表 35 VeraCrypt 実装性能比較（数値）

	Windows環境	macOS環境	Linux環境
AES-XTS	11.1	2.2	8.3
Camellia-XTS	2.8	0.628	3.1

（単位は GB/s）

6.4.2 ストレージに対するベンチマークソフトによる性能測定

このベンチマークでは暗号アルゴリズム毎の性能傾向を知ることができるが、実際にストレージへの書き込み時にどのようなパフォーマンスなのかが判断できない課題がある。そこで、評価対象の暗号アルゴリズムである AES および Camellia を用いて仮想的な暗号ドライブの作成を行い、ストレージの読込/書込に関する性能を確認するため、AmorphousDiskMark[ǂǂǂǂǂ]を用いて実機での性能測定も実施した。AmorphousDiskMark が実施する評価項目については、表 36 に示すとおりである。なお、Queue Depth[§§§§§]はデフォルト値である 32 を指定している。

表 36 AmorphousDiskMark における評価項目

項番	評価項目	概要
1	Seq QD32	シーケンシャルな 128KB (Queue Depth = 32) 読込/書込みテスト
2	4K QD32	4KB のランダム I/O (Queue Depth = 32) 読込/書込みテスト
3	Seq	シーケンシャルな 1MB (Queue Depth = 32) 読込/書込みテスト
4	4K	4KB のランダム 読込/書込みテスト

[ǂǂǂǂǂ] http://www.katsurashareware.com/pgs/adm.html

[§§§§§] ストレージが一度に受け取れる I/O リクエスト数のこと

＜macOS 環境＞

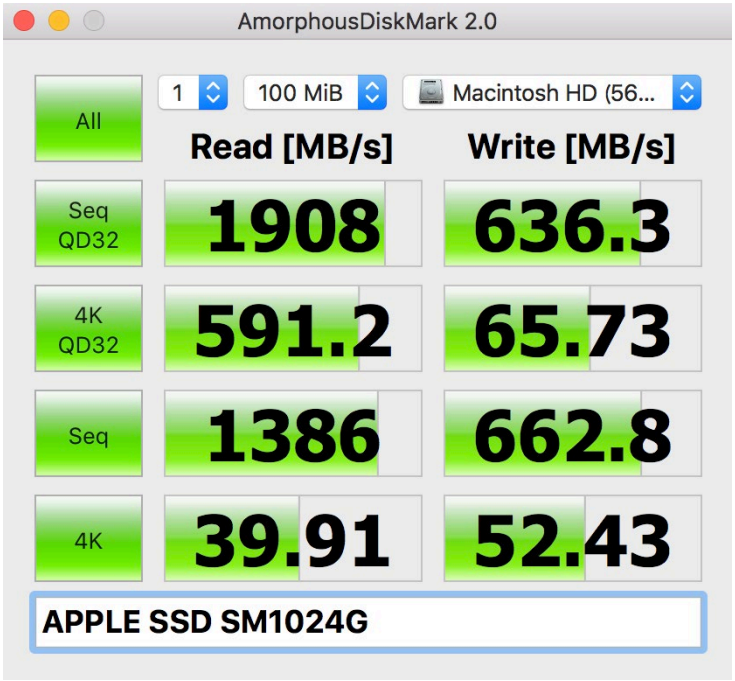


図 30　macOS 環境におけるベンチマーク結果（暗号化なし）

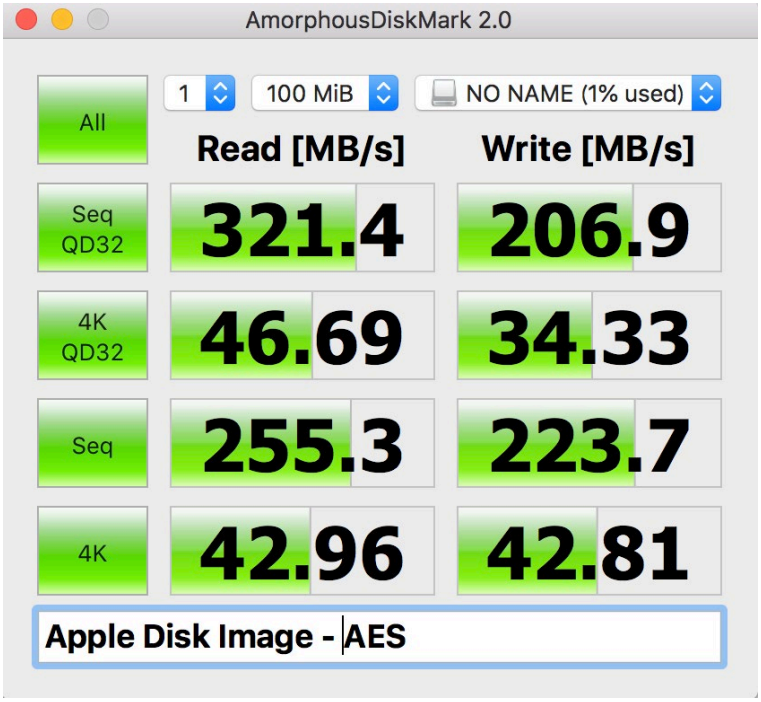


図 31　macOS 環境におけるベンチマーク結果（AES-XTS）

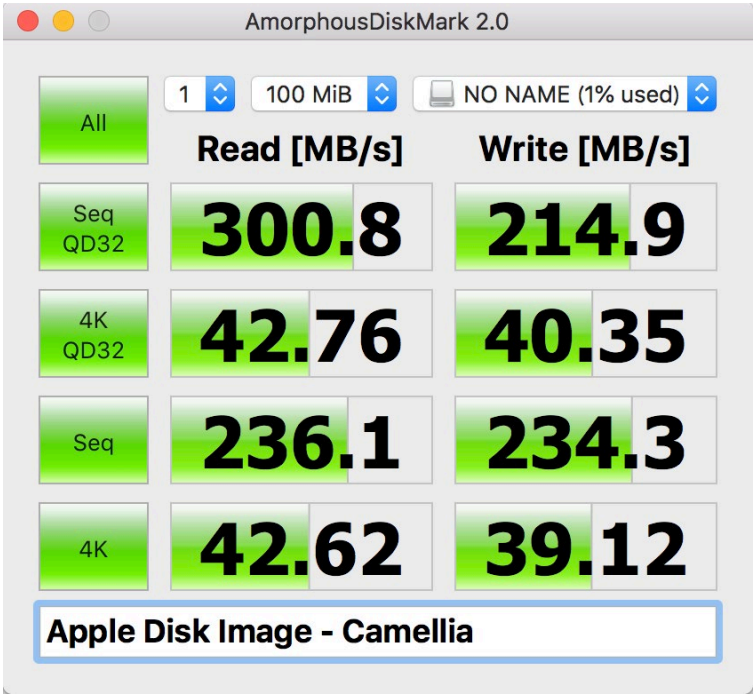


図 32　macOS 環境におけるベンチマーク結果（Camellia-XTS）

6.5　ディスク暗号化製品を使った実装性能比較調査に関する考察

ディスク暗号化に関する技術としては、Windows に搭載されている BitLocker や Apple FileVault、SSD などのハードウェアで実現される SED などにより、ディスク暗号化技術が実用上問題ないことを経験しているが、本節では実際にストレージ暗号化を行う VeraCrypt を用いて、ベンチマークによるスループットおよび仮想的な暗号化ストレージを用いて読込/書込みに関するベンチマークを行い、測定した範囲において実装性能を示すことができた。一般的な暗号ライブラリで実行されるメモリ上でのスループットにより、実際のストレージに対して I/O を行うような測定条件ではスループットが低くなることが確認された。今後、想定される利用シーンを考慮すると、AES-NI や専用チップなどによるハードウェアアクセラレーションの重要性が増加することが予想される。

7　まとめ

本報告書では、XTS の CRYPTREC 暗号リストへの採択の判断材料を提供することを目的として、製品での普及状況および実装性能の観点での評価を行った。
結果として、普及状況の観点では多くの製品での採用実績があり、実装性能としては、ECB や CRYPTREC 暗号リストに採択されているブロック暗号利用モードである CBC と比較した結果、XTS は実際に利用する観点から実用上影響のないパフォーマンスであると判断した。本報告書での調査で得た結果を以下にまとめる。

普及状況の観点では、Windows の BitLocker や macOS の FileVault 2 などのストレージ暗号化ソフトに加え、SED、HDD、USB などのストレージ製品での採用実績を確認できた。XTS を承認アルゴリズムに含む CMVP 認証を取得しているベンダーは 90 社あり、CMVP 認証取得製品だけでも 296 件の製品で XTS が採用されていることがわかった。

実装性能の観点では、客観的な評価として公開されている評価結果に関する調査と実際の計測による評価を実施した。
公開されている評価結果による調査では、2010 年など比較的古い環境での実装性能評価結果は見つかった。一方で、新しい環境での性能評価結果については、今回調査した条件下では存在していなかった。これは、AES-NI のような暗号化機能を補強する機能があることによって、利用者が意識することなく暗号機能が利用されているのではないかと考える。
実際の計測による評価では、OSS として提供されている暗号ライブラリを利用した評価とディスク暗号化ソフトウェアを利用した評価を実施した。
暗号ライブラリを利用した評価では、XTS と CRYPTREC 暗号リストで採択されている暗号利用モードとの実装性能を比較評価するために、Windows 環境、macOS 環境および Linux 環境において、OpenSSL と WolfCrypt を使用した性能評価を行った。結果的に、OpenSSL では ECB や CBC と比較して実用上影響のないパフォーマンスであると判断した。一方、WolfCrypt では OpenSSL とは異なる性能傾向となっていたが AES-NI の拡張命令の限定的な利用がなされていたことによる差であると考えられる。
ディスク暗号化製品を使った実装性能調査では、実際にストレージ暗号化を行う VeraCrypt を用いてベンチマークによるスループットおよび仮想的な暗号化ストレージを用いたベンチマークを行った。その結果、暗号ライブラリで実行されるメモリ上でのスループットよりも、スループットが低くなることが観測された。BitLocker や Apple の FileVault などのストレージ暗号化製品により、ディスク暗号化技術が実用上問題ないことを経験しているが、今後、AES-NI や専用チップなどによるハードウェアアクセラレーションの重要性が増加することが予想される。

参考文献

[1] 峯松一彦, 暗号利用モード XTS の安全性に関する調査及び評価, https://www.cryptrec.go.jp/exreport/cryptrec-ex-2801-2018.pdf, CRYPTREC Report, CRYPTREC, 2019.

[2] P. Rogaway, Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC., ASIACRYPT 2004, volume 3329 of Lecture Notes in Computer Science, pp. 16–31, Springer, 2004.

[3] IEEE Standard for Cryptographic Protectioin of Data on Block-Oriented Storage Devices (IEEE Std 1619-2007), IEEE Security in Storage Working Group, 2007.

[4] M. Dworkin, Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices (NIST SP800-38E), National Institute of Standards and Technology, 2010.

[5] Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices, IEEE Security in Storage Working Group, 2018.

[6] CRYPTREC, 暗号技術検討会 2018 年度 報告書, https://www.cryptrec.go.jp/report/cryptrec-rp-1000-2018.pdf, 2019.

[7] The XTS-AES Validation System (XTSVS), National Institute of Standards and Technology, 2013.

[8] P. Rogaway, Evaluation of Some Blockcipher Modes of Operation, CRYPTREC Report, 2011.

OpenSSL に関する実装性能調査結果

「表 16　暗号ライブラリでの実装を考慮した測定対象の暗号利用モード」において、CRYPTREC 暗号リストで採択されている暗号利用モードに対して、OpenSSL を用いた実装性能測定を行った結果を示す。なお、鍵長については XTS において利用可能な鍵長である 128 ビットおよび 256 ビットの範囲とした。なお、CCM については本調査での事前による実装性能測定を行ったところ、スループットが頭打ちにならない状況だったため、本調査での測定対象から除外している。

＜Windows 環境：暗号化＞

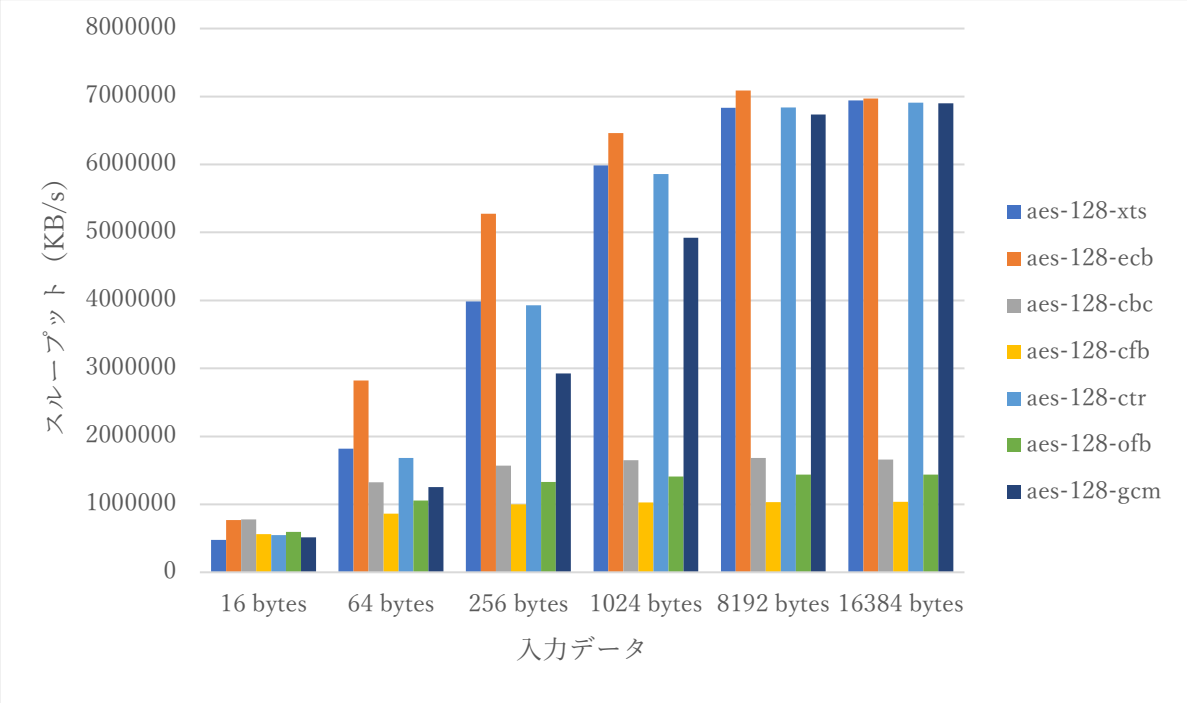


図 33　Windows 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

表 37　Windows 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	478609.98	1820717.5	3986596.18	5985235.81	6833793.71	6942627.16
aes-128-ecb	770128.73	2823577.49	5275877.12	6463844.35	7086702.59	6969715.37
aes-128-cbc	777187.64	1323790.29	1568680.62	1650111.15	1683996.67	1658619.04
aes-128-cfb	560854.78	862827.61	1000770.9	1029442.54	1030362.45	1038456.15
aes-128-ctr	546357.24	1681912.62	3926952.77	5860314.45	6837665.79	6911404.71
aes-128-ofb	593886.81	1056862.42	1329955.07	1410018.99	1436641.96	1438913.88
aes-128-gcm	512624.05	1252742.53	2924503.13	4921580.76	6736581.97	6900356.44

（単位は KB/s）

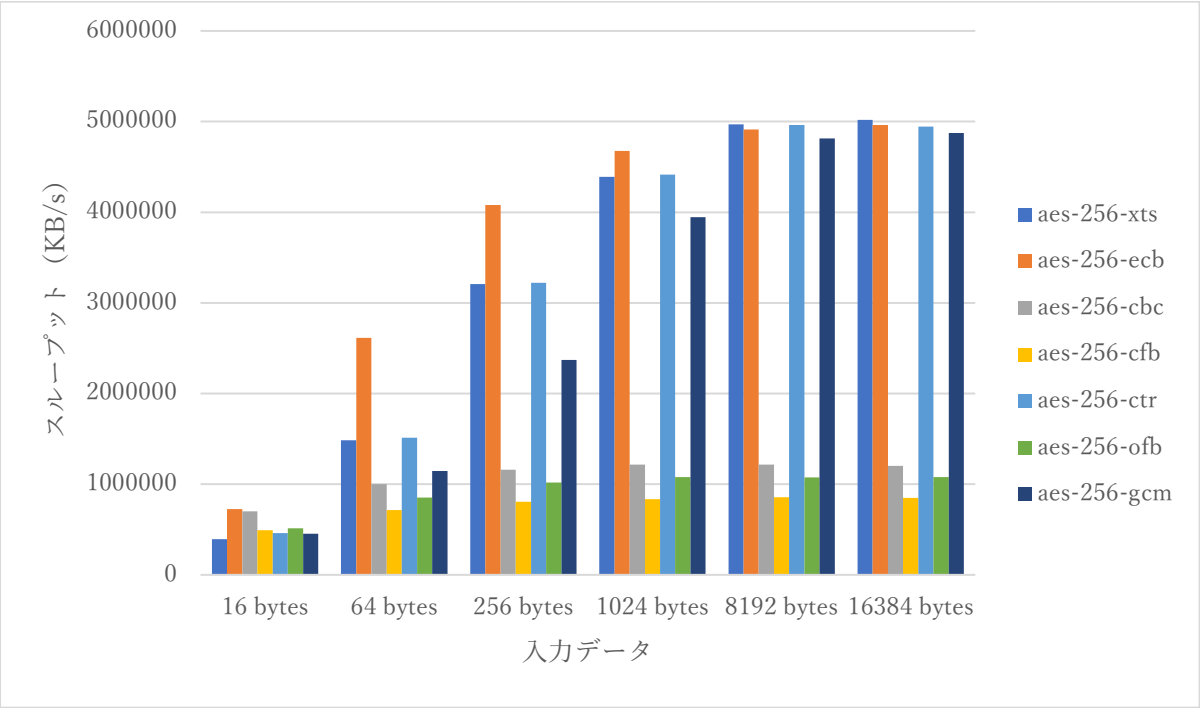


図 34　Windows 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

表 38　Windows 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	394505.55	1485969.69	3206756.95	4389973.33	4971151.36	5018834.26
aes-256-ecb	725685.11	2614959.79	4080193.11	4677160.96	4914577.92	4961577.64
aes-256-cbc	699745.21	1001478.6	1159585.19	1214693.03	1217844.57	1200974.51
aes-256-cfb	491771.83	715872.64	804908.54	833802.92	857511.25	849139.03
aes-256-ctr	458731.65	1512950.17	3220290.3	4415954.26	4962249.39	4945964.57
aes-256-ofb	514370.54	852667.48	1018049.3	1078692.52	1075800.75	1076838.4
aes-256-gcm	454155.61	1145228.82	2371759.36	3946969.09	4813305.17	4872754.4

（単位は KB/s）

＜Windows 環境：復号＞

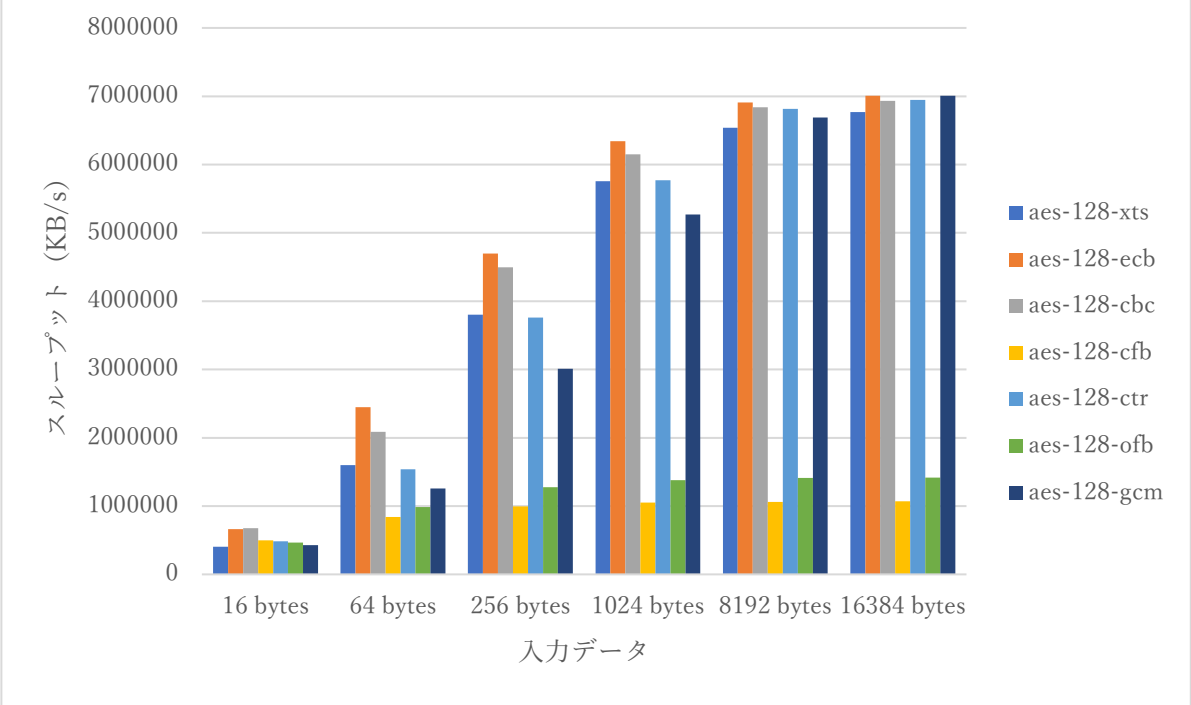


図 35　Windows 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

表 39　Windows 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	406550.97	1597159.1	3800503.13	5753965.23	6540137.81	6769076.91
aes-128-ecb	661185.37	2449162.37	4697878.1	6343713.79	6906519.55	7005536.26
aes-128-cbc	677288.13	2088954.32	4496207.62	6148311.72	6836240.38	6930699.61
aes-128-cfb	498211.86	841427.07	995144.87	1052394.15	1062267.56	1069170.69
aes-128-ctr	486533.34	1537274.67	3758323.88	5768802.99	6816617.81	6946903.38
aes-128-ofb	465405.97	984717.44	1275794.35	1381383.17	1412098.73	1416511.49
aes-128-gcm	426416.38	1259254.74	3008196.27	5268030.88	6689554.43	7005356.03

（単位は KB/s）

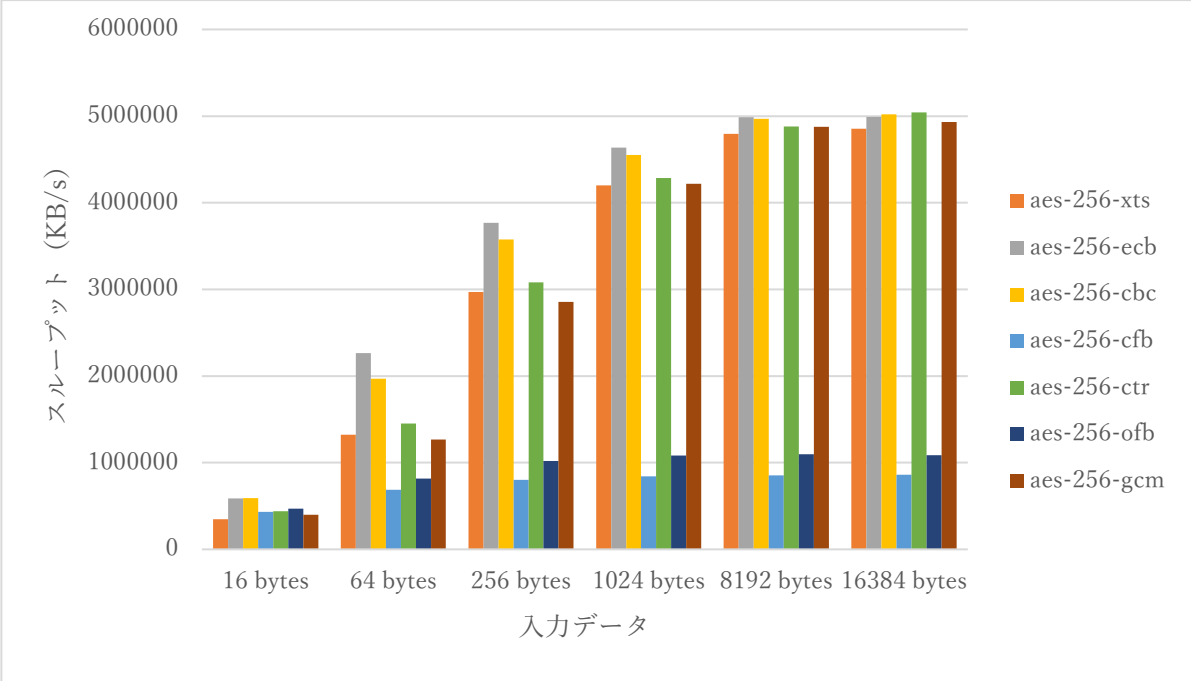


図 36　Windows 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

表 40　Windows 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	348375.03	1322908.18	2969041.83	4198649.86	4794376.19	4855540.39
aes-256-ecb	585959.97	2264927.85	3769427.54	4635617.28	4986170.03	4992002.73
aes-256-cbc	589592.83	1969084.55	3575207.34	4552853.16	4969368.23	5018932.57
aes-256-cfb	432998.7	689175.87	800611.93	843241.81	853079.38	862633.98
aes-256-ctr	438964.59	1452627.31	3081824.6	4286603.26	4881496.16	5043268.27
aes-256-ofb	471026.18	815142.31	1018159.79	1083636.73	1097394.86	1086253.74
aes-256-gcm	400891.97	1266506.9	2855870.38	4218216.79	4878136.66	4930868.57

（単位は KB/s）

＜macOS 環境：暗号化＞

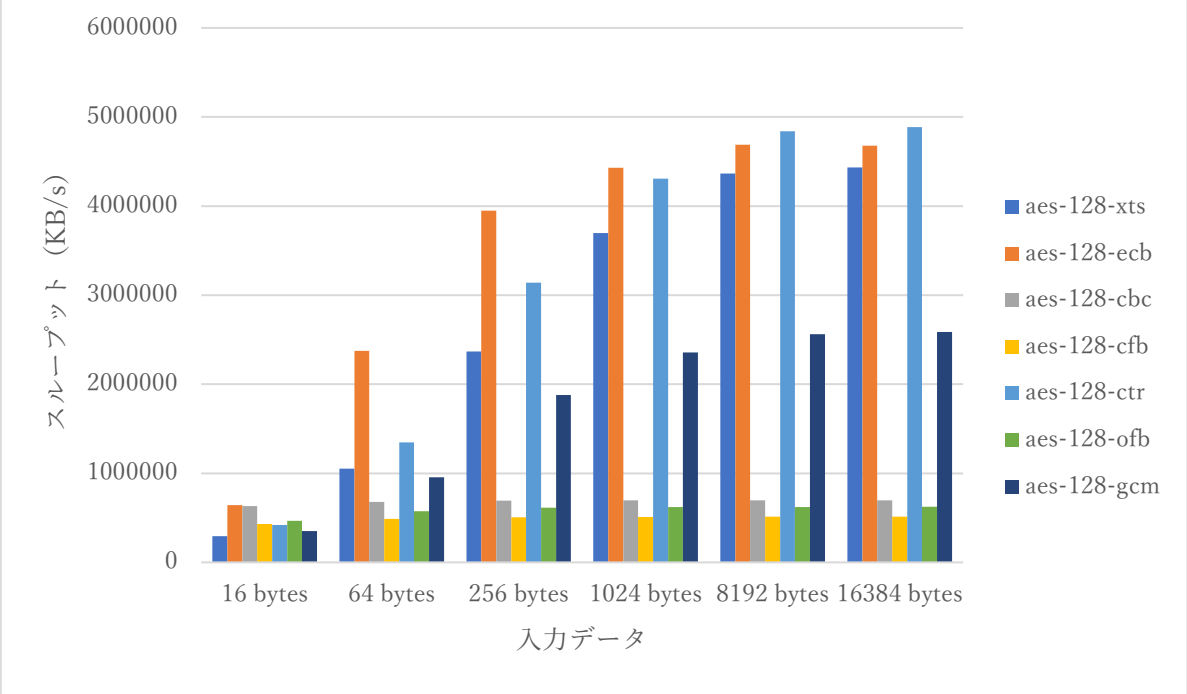


図 37　macOS 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

表 41　macOS 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	295392.33	1051413.08	2366826.84	3697282.05	4366398.81	4433149.95
aes-128-ecb	641190.44	2375444.44	3949783.43	4430856.87	4688794.97	4678402.05
aes-128-cbc	630710.03	678794.93	691990.02	694800.04	697338.54	697860.1
aes-128-cfb	431095.69	488732.29	506515.54	511437.48	511545.49	511670.69
aes-128-ctr	420790.13	1348449.54	3140157.78	4307738.37	4839994.71	4885867.18
aes-128-ofb	464845.4	576056.41	612423.85	620429.69	622197.37	624263.17
aes-128-gcm	351343.31	955675.34	1878452.48	2356976.3	2561720.05	2585045.67

（単位は KB/s）

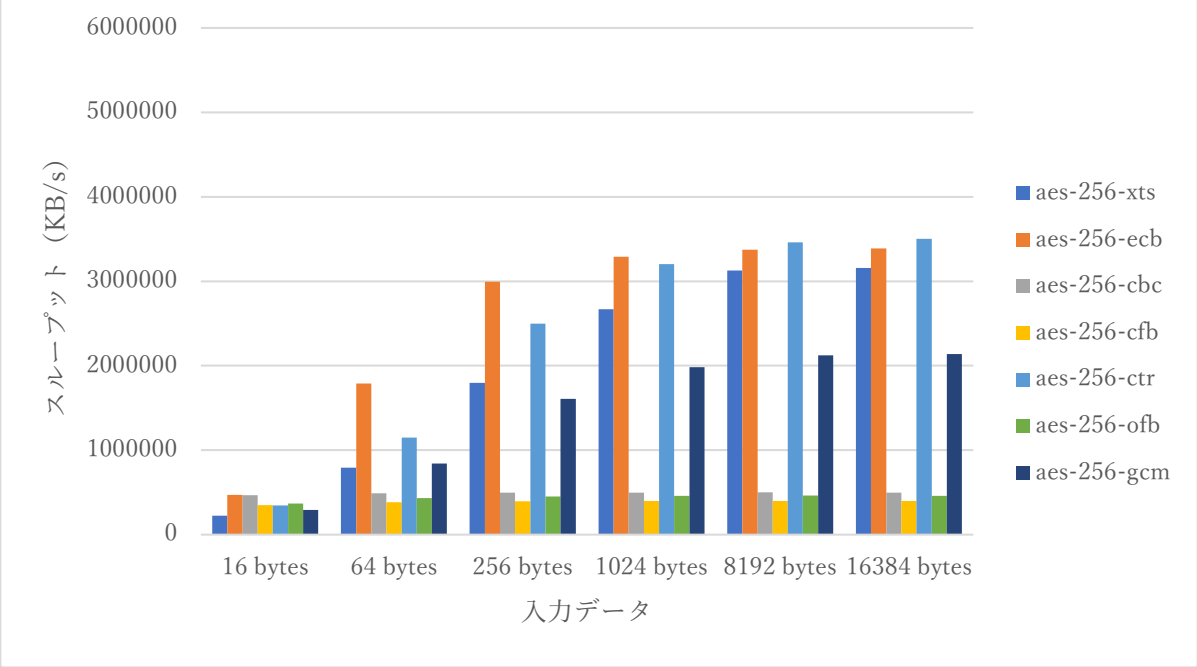


図 38　macOS 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

表 42　macOS 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	223888.69	791039.38	1799060.33	2668685.65	3125947.05	3159619.02
aes-256-ecb	470124.41	1788392.13	2995652.01	3291116.61	3372442.28	3389500.07
aes-256-cbc	466230.33	491065.47	497611.35	498652.16	500989.95	498715.9
aes-256-cfb	348141.06	383924.69	394012.33	396787.75	397614.1	398202.2
aes-256-ctr	345684.77	1148385.54	2499133.44	3204895.4	3459179.5	3502745.77
aes-256-ofb	368329.83	433497.73	453150.62	460091.73	461512.7	459325.44
aes-256-gcm	292808.43	841369.96	1609073.54	1982040.75	2123197.1	2137210.88

（単位は KB/s）

＜macOS 環境：復号＞

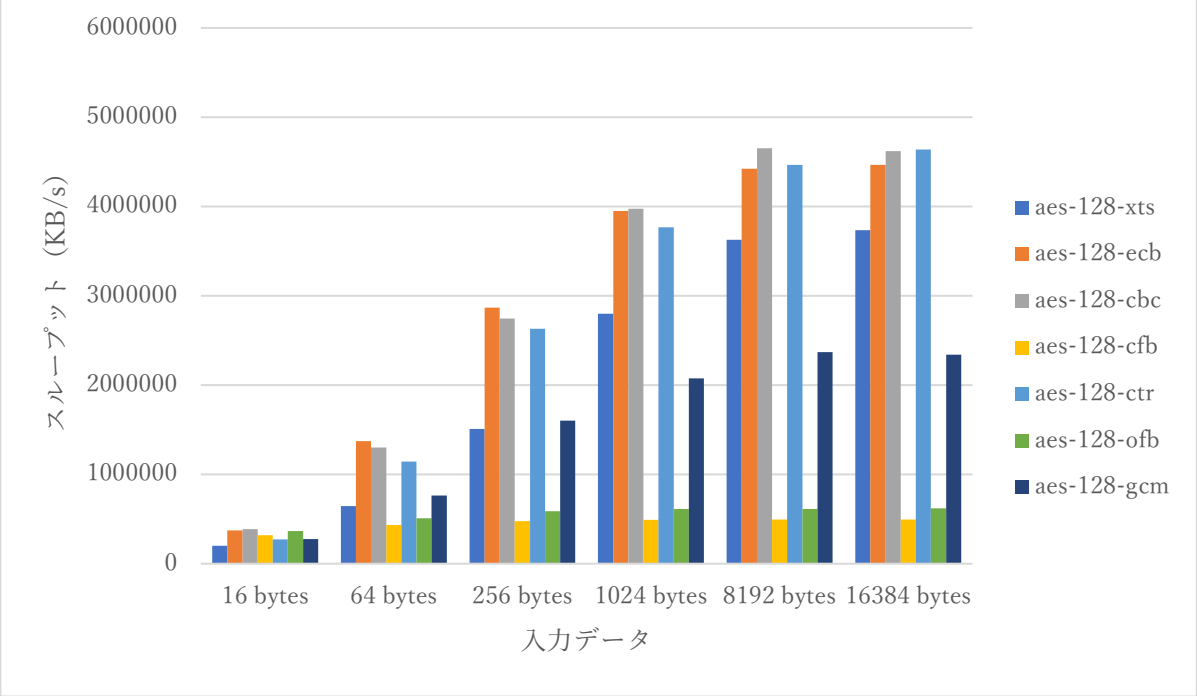


図 39　macOS 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

表 43　macOS 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	200245.97	644834.17	1508976.67	2801045.12	3626081.59	3736288.73
aes-128-ecb	372301.69	1374527.02	2867709.87	3951113.59	4421830.64	4466393.47
aes-128-cbc	387057.07	1302147.15	2746423.82	3974182.57	4651389.92	4620315.58
aes-128-cfb	320172.6	433507.4	478339.08	492003.48	496153	495574.63
aes-128-ctr	271475.72	1143494.66	2629435.88	3767739.85	4467682.35	4637601.9
aes-128-ofb	365591.46	509757.1	588611.95	612237.8	614295.19	621342.65
aes-128-gcm	276501.44	762940.59	1600615.43	2074104.41	2370083.98	2341478.4

（単位は KB/s）

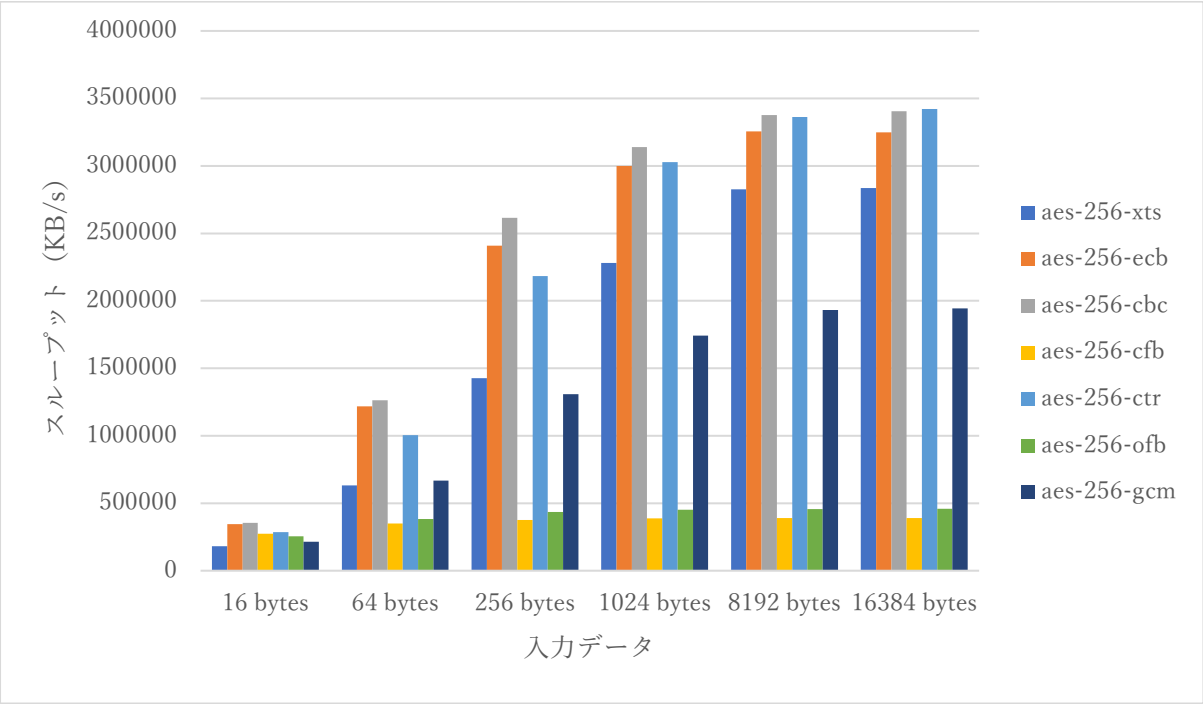


図 40　macOS 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

表 44　macOS 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	181981.5	633080.13	1427062.81	2280764.79	2825978.85	2836763.99
aes-256-ecb	345168.62	1218058.71	2409308.86	2999994.93	3256096.58	3247314.28
aes-256-cbc	353670.02	1263651.66	2614263.61	3140111.02	3376723.22	3406025.38
aes-256-cfb	273790.64	350124.63	375911.52	388039.49	390117.29	390049.53
aes-256-ctr	286953.27	1004552.44	2182518.57	3028736.35	3362234.01	3420763.33
aes-256-ofb	255134.2	383889.94	435321.6	452945.58	457643.35	458125.23
aes-256-gcm	215680.52	667849.82	1307931.22	1741574.14	1932315.31	1944021.67

（単位は KB/s）

＜Linux 環境：暗号化＞

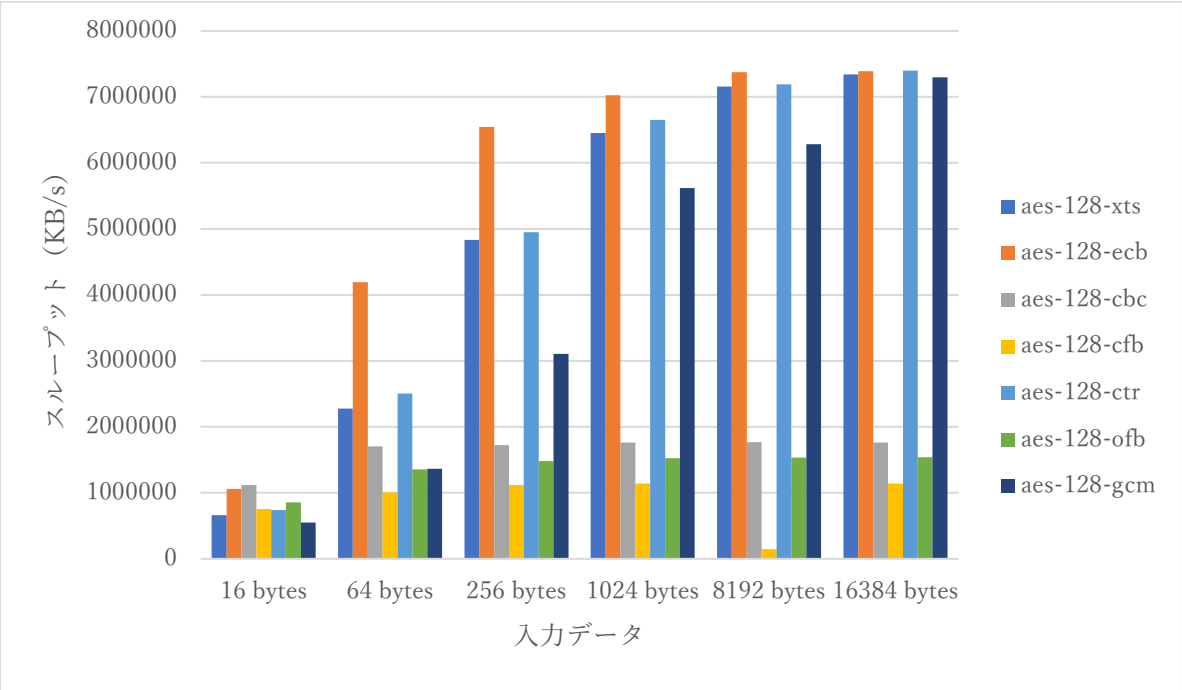


図 41　Linux 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

表 45　Linux 環境における暗号利用モード別実装性能（OpenSSL: 128bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	661516.04	2279209.56	4833391.02	6455110.66	7155864.92	7339769.86
aes-128-ecb	1061451.1	4193916.16	6546062.85	7024358.4	7376401.75	7391002.62
aes-128-cbc	1119322.87	1706853.65	1722613.42	1764965.03	1767754.41	1764027.05
aes-128-cfb	751704.38	1007290.47	1117634.47	1142679.21	146833.58	1142008.49
aes-128-ctr	737892.05	2506246.4	4947914.07	6649841.32	7189613.23	7401482.92
aes-128-ofb	853811.17	1355799.89	1479747.67	1522986.33	1534651.05	1538402.99
aes-128-gcm	551437.96	1367443.48	3106557.53	5617884.84	6284465.49	7296679.94

（単位は KB/s）

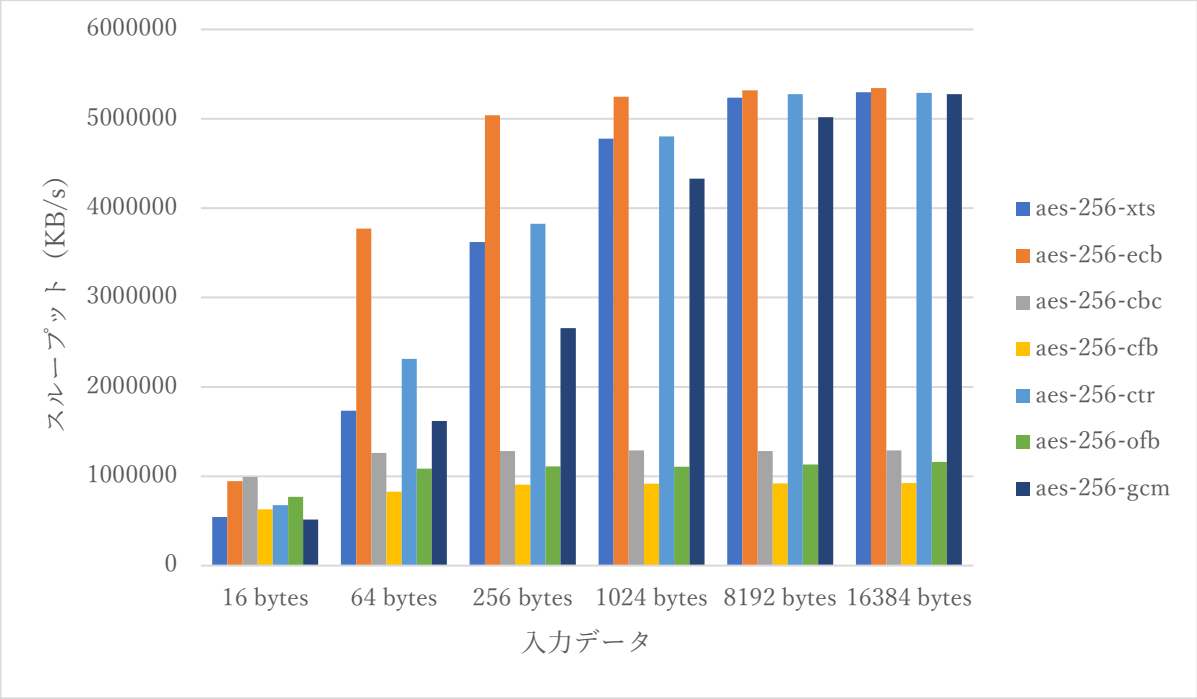


図 42　Linux 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

表 46　Linux 環境における暗号利用モード別実装性能（OpenSSL: 256bit 暗号化）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	543179.57	1732118.38	3621865.56	4778003.8	5237033.64	5296936.28
aes-256-ecb	944778.55	3770500.5	5040330.84	5248844.8	5318967.3	5345012.39
aes-256-cbc	993300.81	1260529.54	1283763.2	1288221.7	1283825.66	1287591.25
aes-256-cfb	631964.9	827079.64	905192.79	917515.26	920322.05	922894.34
aes-256-ctr	675295.37	2312978.03	3825245.95	4804376.58	5275538.77	5288645.97
aes-256-ofb	771115.41	1084587.01	1111855.53	1107193.86	1133259.43	1158818.47
aes-256-gcm	517057.63	1619436.8	2657961.73	4331651.41	5016505	5275314.86

（単位は KB/s）

＜Linux 環境：復号＞

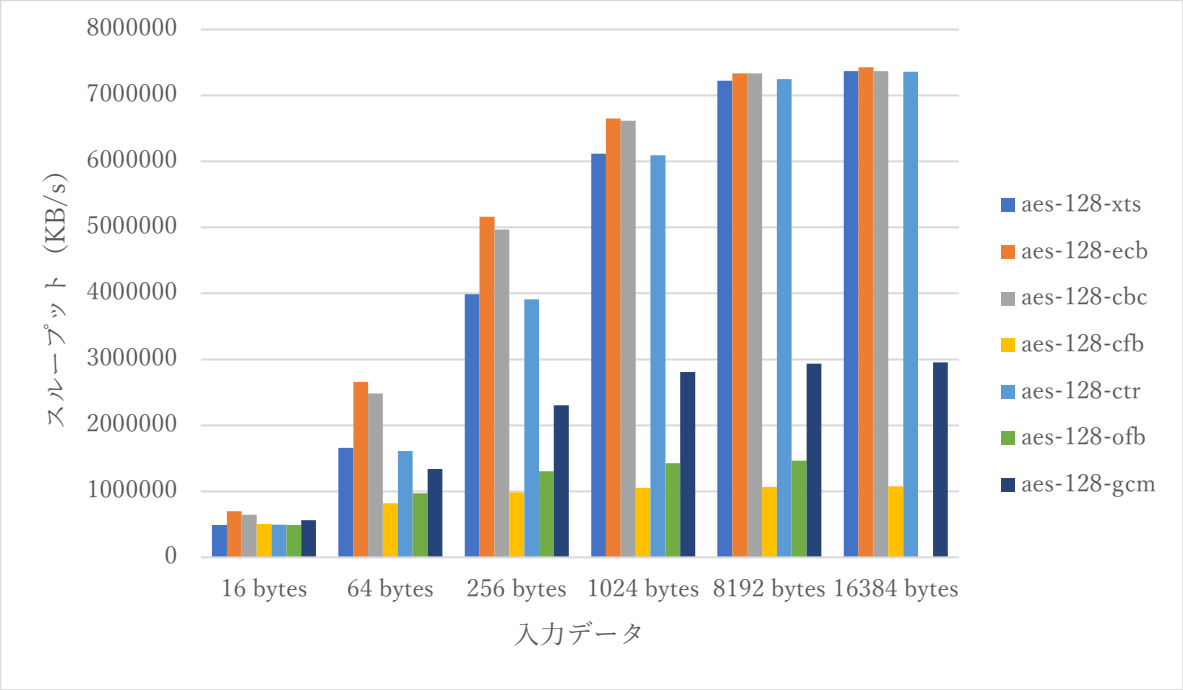


図 43　Linux 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

表 47　Linux 環境における暗号利用モード別実装性能（OpenSSL: 128bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-128-xts	488168.52	1659935.85	3990094.17	6119095.64	7220923.05	7370932.22
aes-128-ecb	700743.92	2658373.31	5160932.78	6652813.99	7335744.85	7425288.87
aes-128-cbc	647194.95	2484312.32	4968155.14	6615095.3	7333931.69	7368824.15
aes-128-cfb	504244.49	820187.39	985202.18	1051696.13	1066573.82	1075871.74
aes-128-ctr	493407.48	1611306.03	3908796.76	6094154.41	7245433.51	7360561.15
aes-128-ofb	491785.02	970106.94	1303616	1424346.11	1463282.35	1471354.2
aes-128-gcm	561432.51	1339588.46	2302199.89	2811140.44	2937307.14	2953172.31

（単位は KB/s）

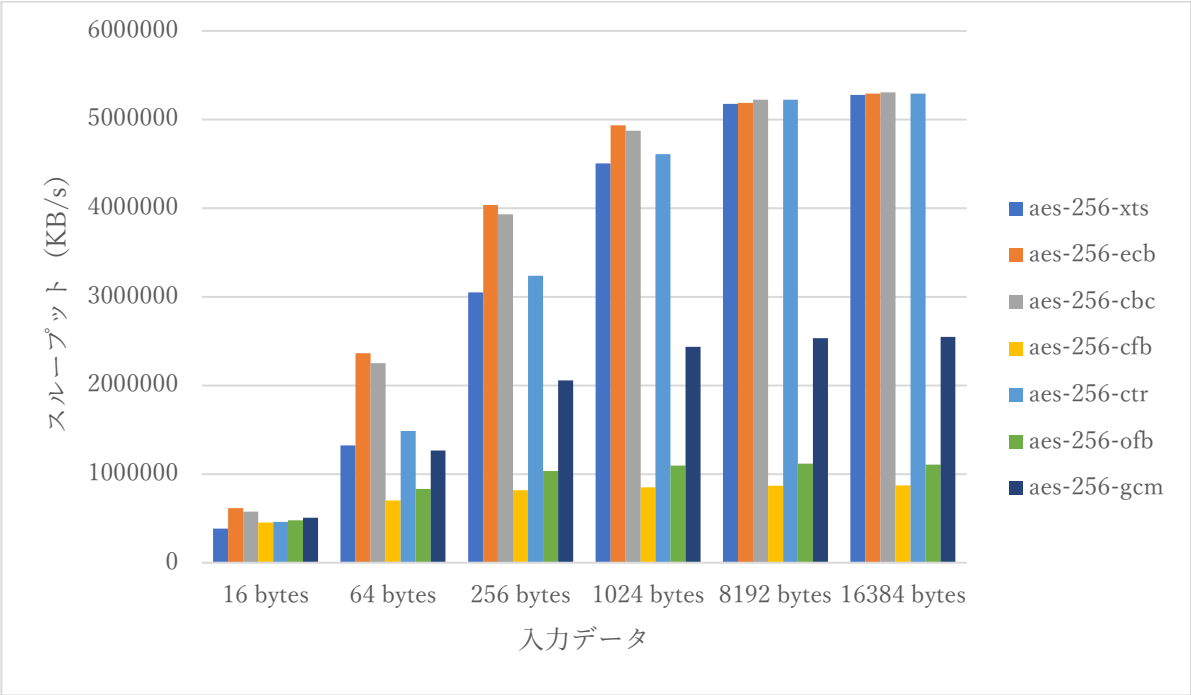


図 44　Linux 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

表 48　Linux 環境における暗号利用モード別実装性能（OpenSSL: 256bit 復号）

	16 bytes	64 bytes	256 bytes	1024 bytes	8192 bytes	16384 bytes
aes-256-xts	384665.19	1324694.98	3050718.81	4504600.58	5176423.77	5278963.03
aes-256-ecb	616197.9	2363642.99	4035974.23	4933271.55	5187674.11	5292894.89
aes-256-cbc	575186.49	2252488.51	3929712.13	4874141.35	5223746.22	5305805.48
aes-256-cfb	452018.73	702641.24	817115.65	852125.01	867647.49	872103.94
aes-256-ctr	461044.95	1484724.65	3235897.09	4610137.77	5224682.84	5294227.46
aes-256-ofb	477868.11	833128.98	1033522.69	1097309.18	1118524.76	1108530.52
aes-256-gcm	507229.18	1264452.95	2056969.39	2434975.74	2533695.49	2548154.37

（単位は KB/s）

WolfCrypt に関する実装性能調査結果

「表 16　暗号ライブラリでの実装を考慮した測定対象の暗号利用モード」において、CRYPTREC 暗号リストで採択されている暗号利用モードに対して、WolfCrypt を用いた実装性能測定を行った結果を示す。なお、鍵長については XTS において利用可能な鍵長である 128 ビットおよび 256 ビットの範囲とした。なお、WolfCrypt においてベンチマークの測定項目として対応していないもの（差異が発生する観点として、鍵長が 128 ビット/256 ビットに未対応および復号処理に関して未実装）については、スループットを 0 KB/s としている。

＜Windows 環境：暗号化＞

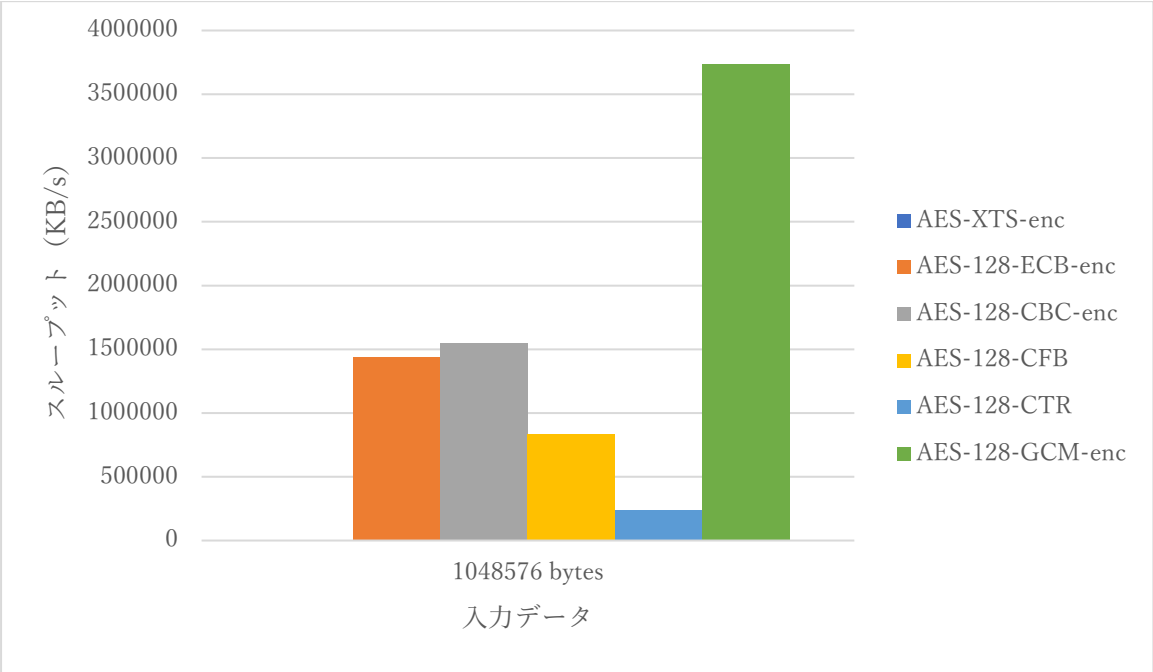


図 45　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

表 49　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

	AES-XTS-enc	AES-128-ECB-enc	AES-128-CBC-enc	AES-128-CFB	AES-128-CTR	AES-128-GCM-enc
1048576 bytes	0	1441730	1548909	831177	242341	3739314

（単位は KB/s）

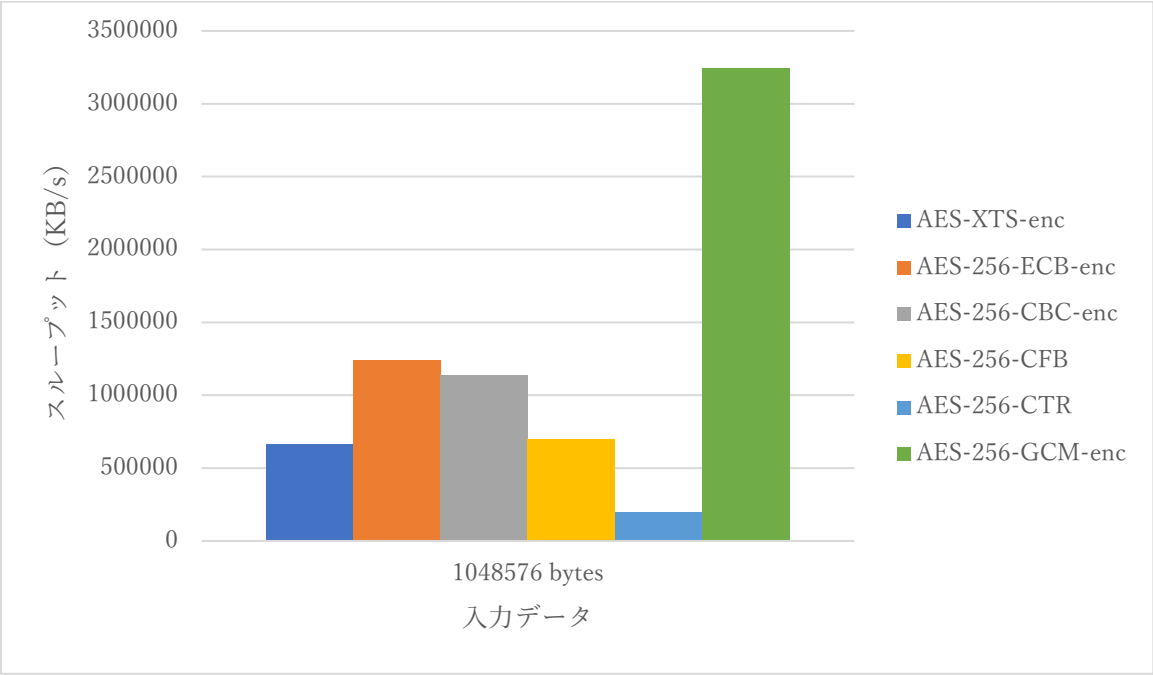


図 46　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

表 50　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

	AES-XTS-enc	AES-256-ECB-enc	AES-256-CBC-enc	AES-256-CFB	AES-256-CTR	AES-256-GCM-enc
1048576 bytes	661023	1238166	1136883	692510	195377	3242569

（単位は KB/s）

＜Windows 環境：復号＞

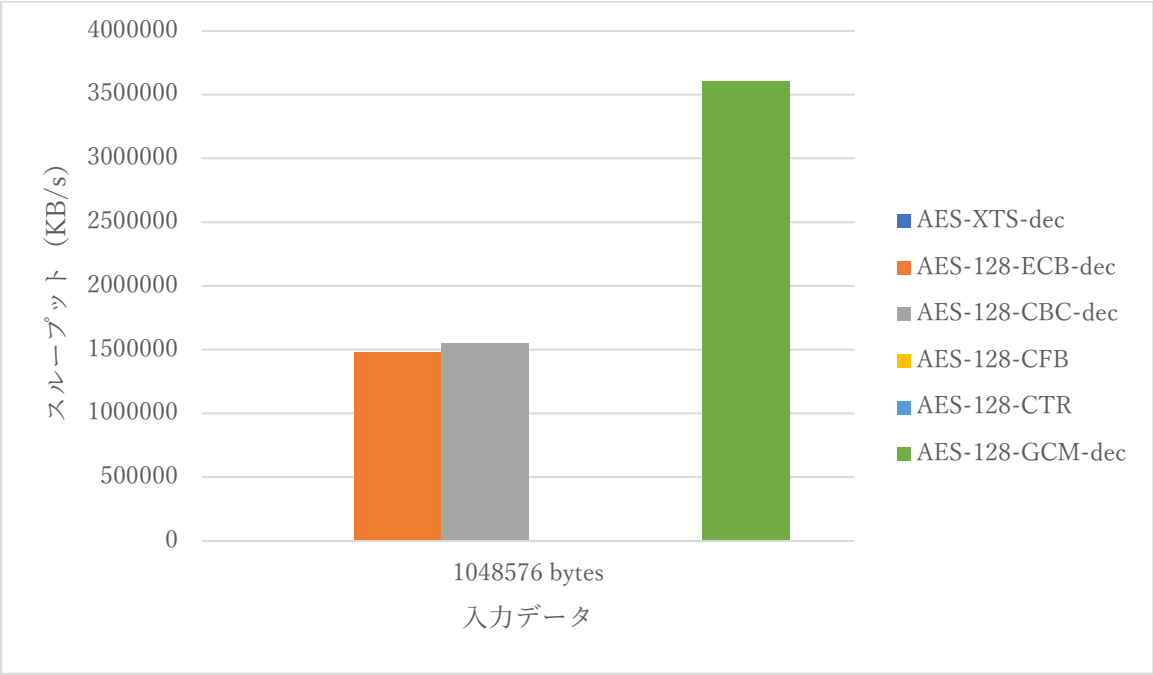


図 47　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

表 51　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

	AES-XTS-dec	AES-128-ECB-dec	AES-128-CBC-dec	AES-128-CFB	AES-128-CTR	AES-128-GCM-dec
1048576 bytes	0	1475929	1548909	0	0	3606427

（単位は KB/s）

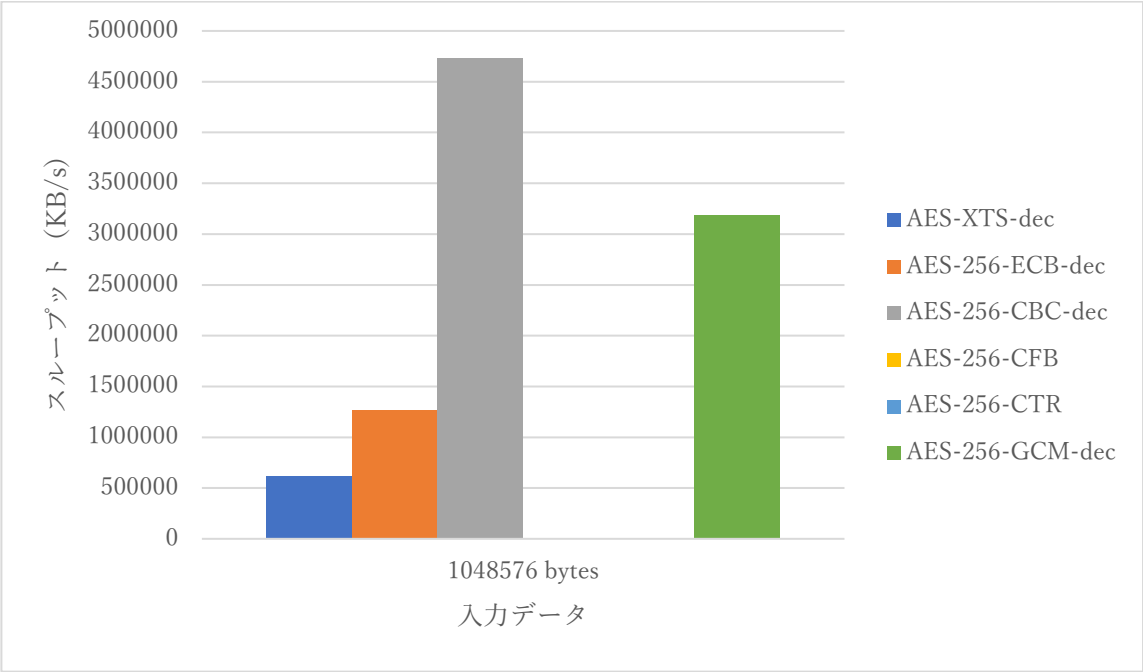


図 48　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

表 52　Windows 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

	AES-XTS-dec	AES-256-ECB-dec	AES-256-CBC-dec	AES-256-CFB	AES-256-CTR	AES-256-GCM-dec
1048576 bytes	614010	1264893	4728723	0	0	3189713

（単位は KB/s）

＜macOS 環境：暗号化＞

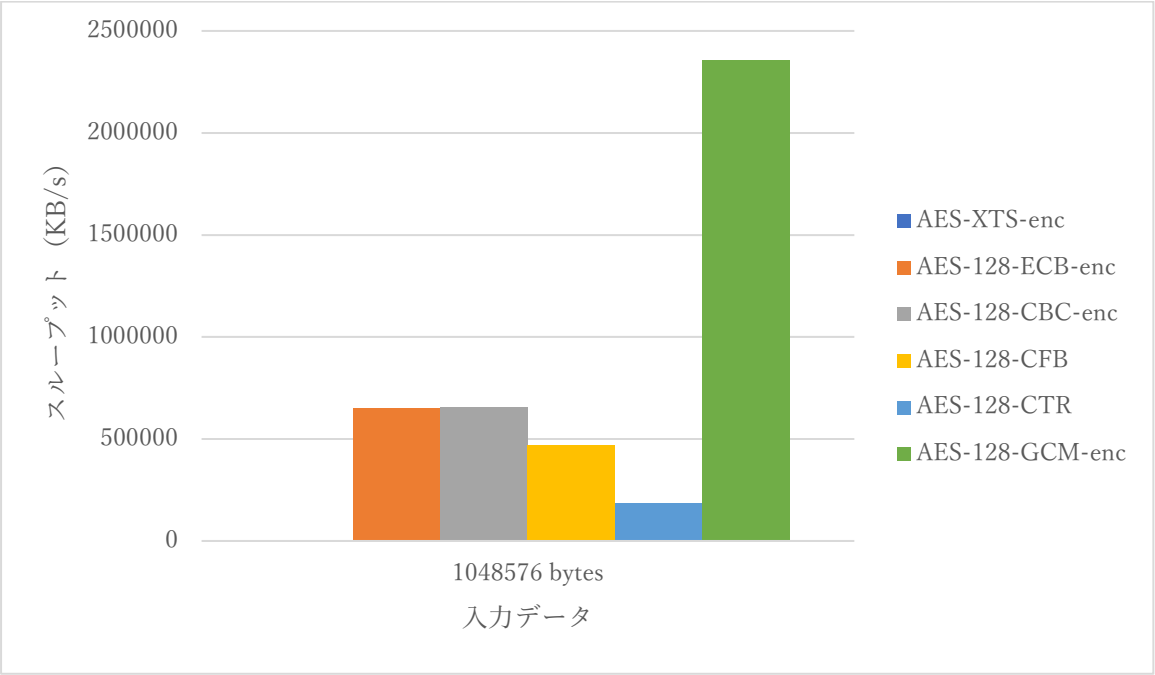


図 49　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

表 53　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

	AES-XTS-enc	AES-128-ECB-enc	AES-128-CBC-enc	AES-128-CFB	AES-128-CTR	AES-128-GCM-enc
1048576 bytes	0	650949	653942	467095	185870	2355569

（単位は KB/s）

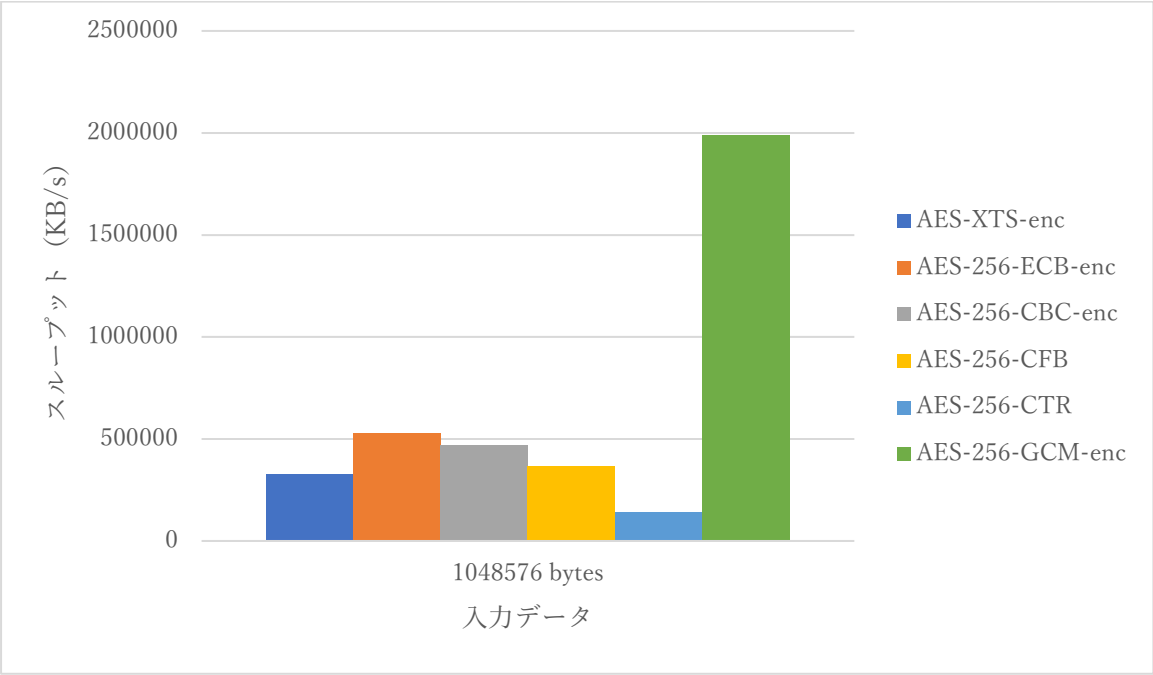


図 50　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

表 54　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

	AES-XTS-enc	AES-256-ECB-enc	AES-256-CBC-enc	AES-256-CFB	AES-256-CTR	AES-256-GCM-enc
1048576 bytes	326383	528321	469786	365140	142796	1989329

（単位は KB/s）

＜macOS 環境：復号＞

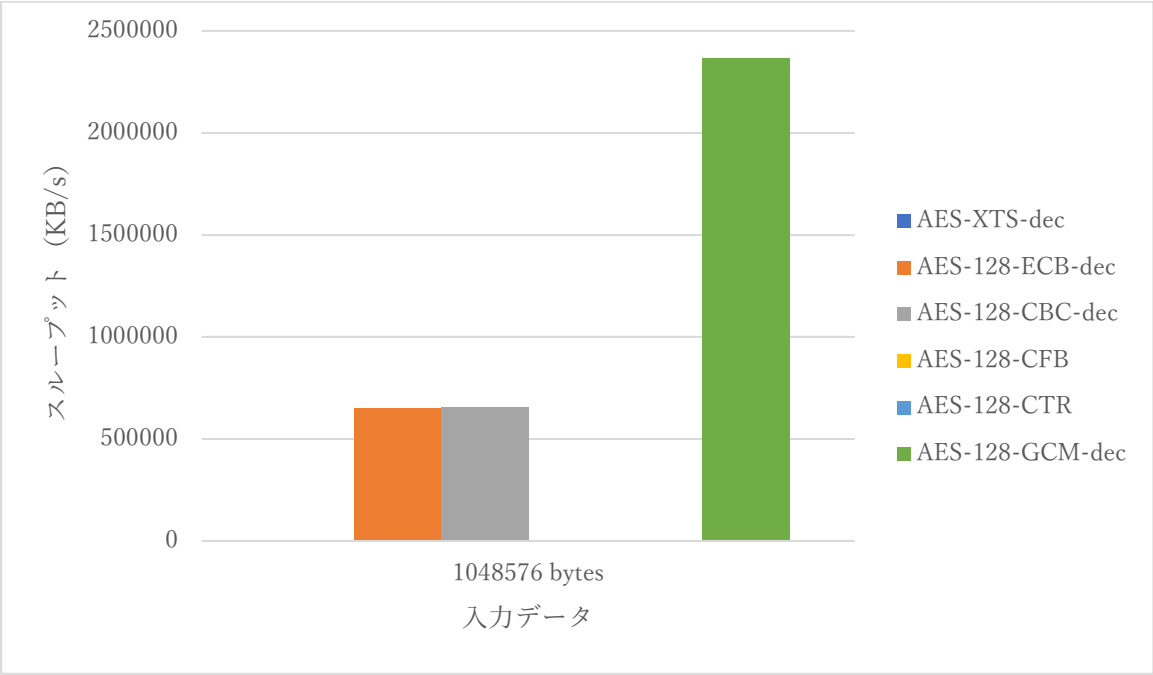


図 51　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

表 55　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

	AES-XTS-dec	AES-128-ECB-dec	AES-128-CBC-dec	AES-128-CFB	AES-128-CTR	AES-128-GCM-dec
1048576 bytes	0	649625	653942	0	0	2364187

（単位は KB/s）

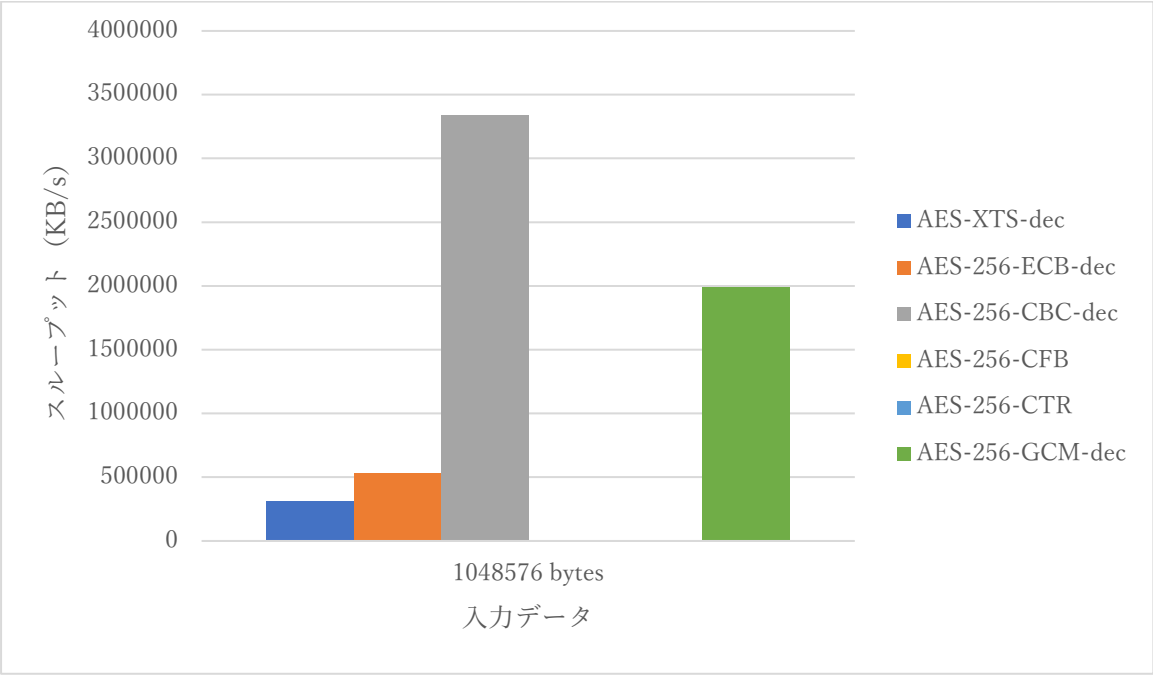


図 52　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

表 56　macOS 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

	AES-XTS-dec	AES-256-ECB-dec	AES-256-CBC-dec	AES-256-CFB	AES-256-CTR	AES-256-GCM-dec
1048576 bytes	312059	526088	3335104	0	0	1986771

（単位は KB/s）

＜Linux 環境：暗号化＞

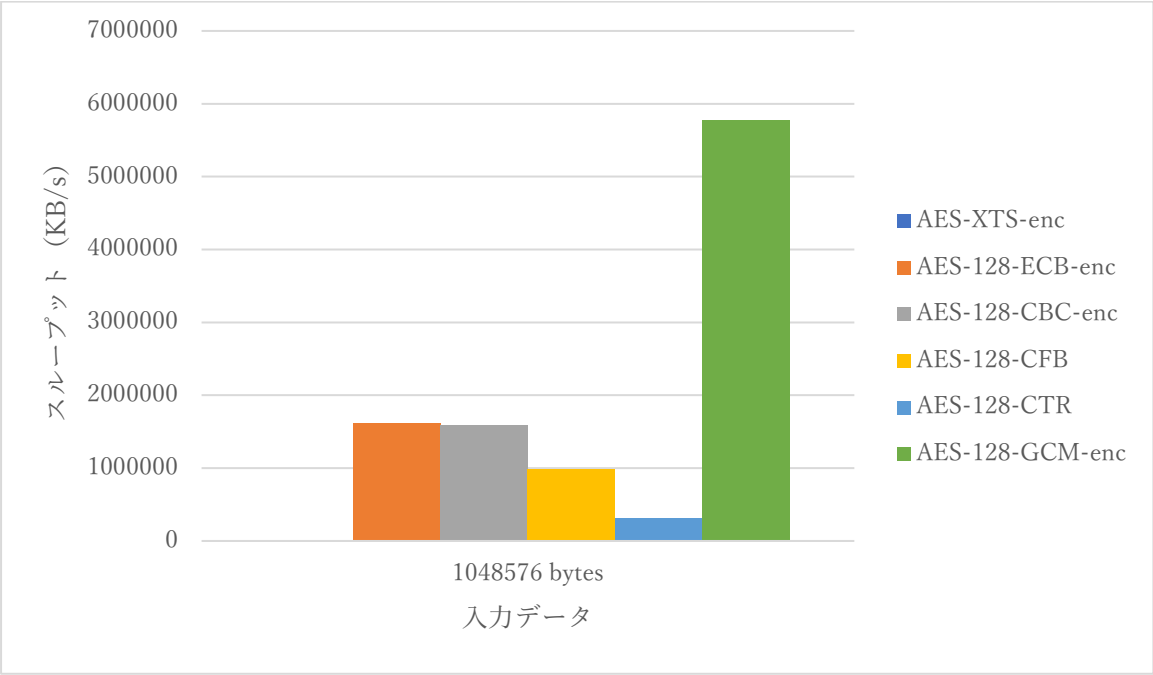


図 53　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

表 57　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 暗号化）

	AES-XTS-enc	AES-128-ECB-enc	AES-128-CBC-enc	AES-128-CFB	AES-128-CTR	AES-128-GCM-enc
1048576 bytes	0	1611741	1579842	978631	311652	5768223

（単位は KB/s）

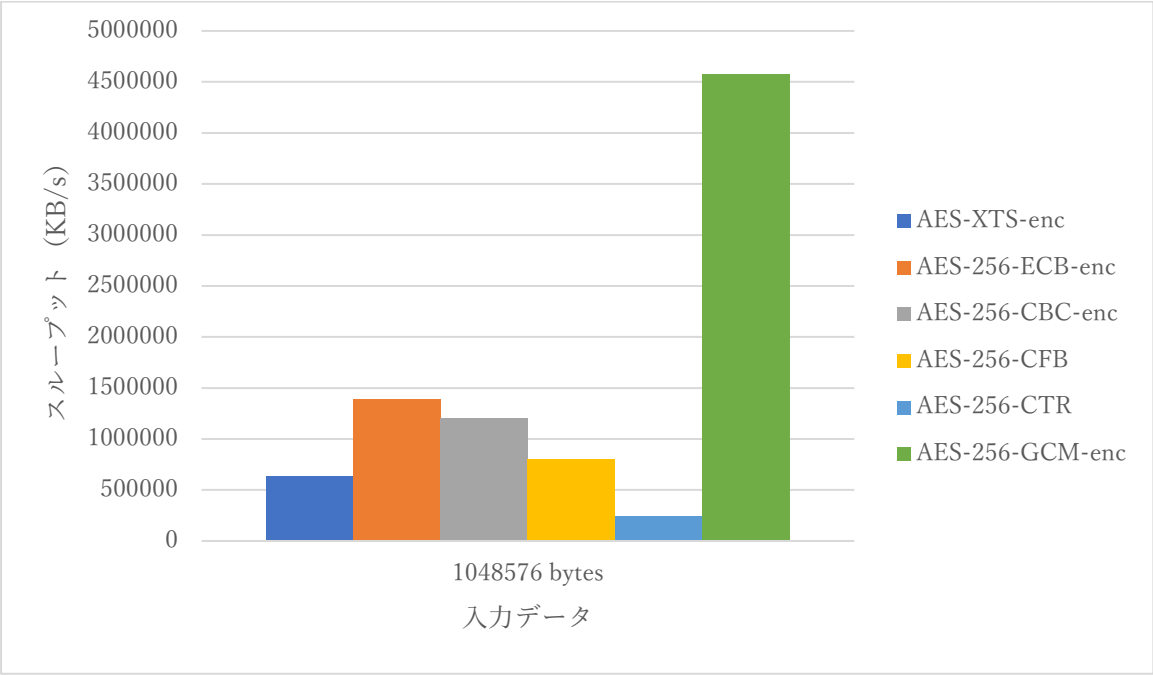


図 54　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

表 58　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 暗号化）

	AES-XTS-enc	AES-256-ECB-enc	AES-256-CBC-enc	AES-256-CFB	AES-256-CTR	AES-256-GCM-enc
1048576 bytes	630823	1381395	1197638	795793	239931	4574227

（単位は KB/s）

＜Linux 環境：復号＞

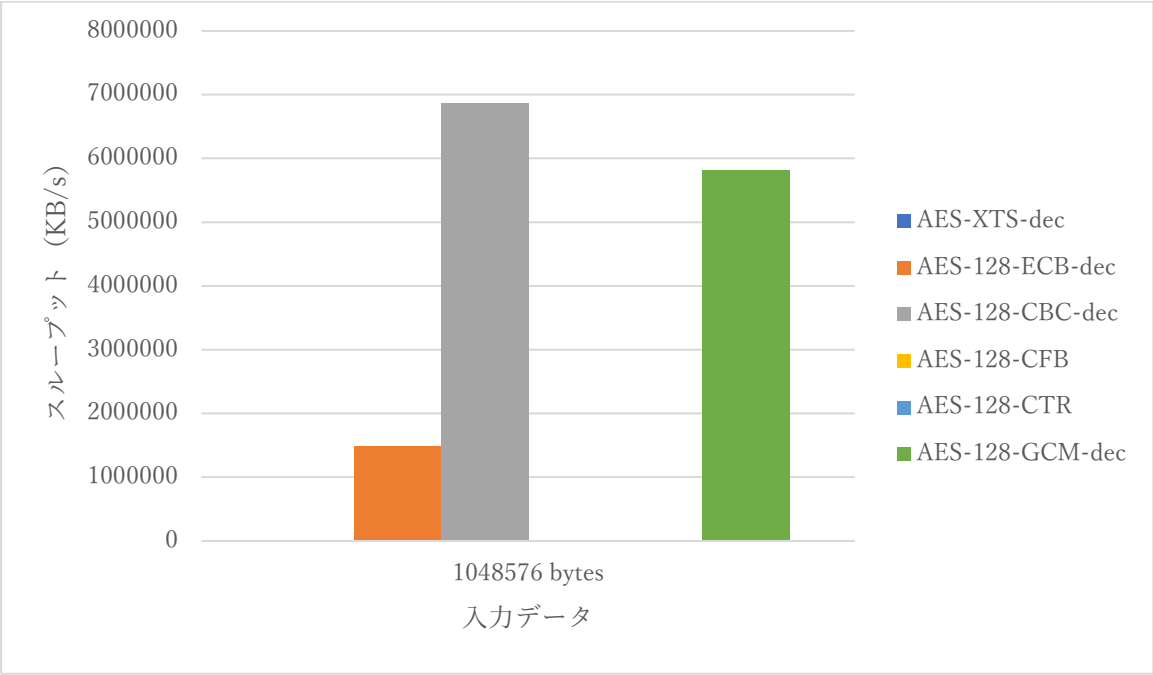


図 55　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

表 59　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 128bit 復号）

	AES-XTS-dec	AES-128-ECB-dec	AES-128-CBC-dec	AES-128-CFB	AES-128-CTR	AES-128-GCM-dec
1048576 bytes	0	1478008	6862302	0	0	5809065

（単位は KB/s）

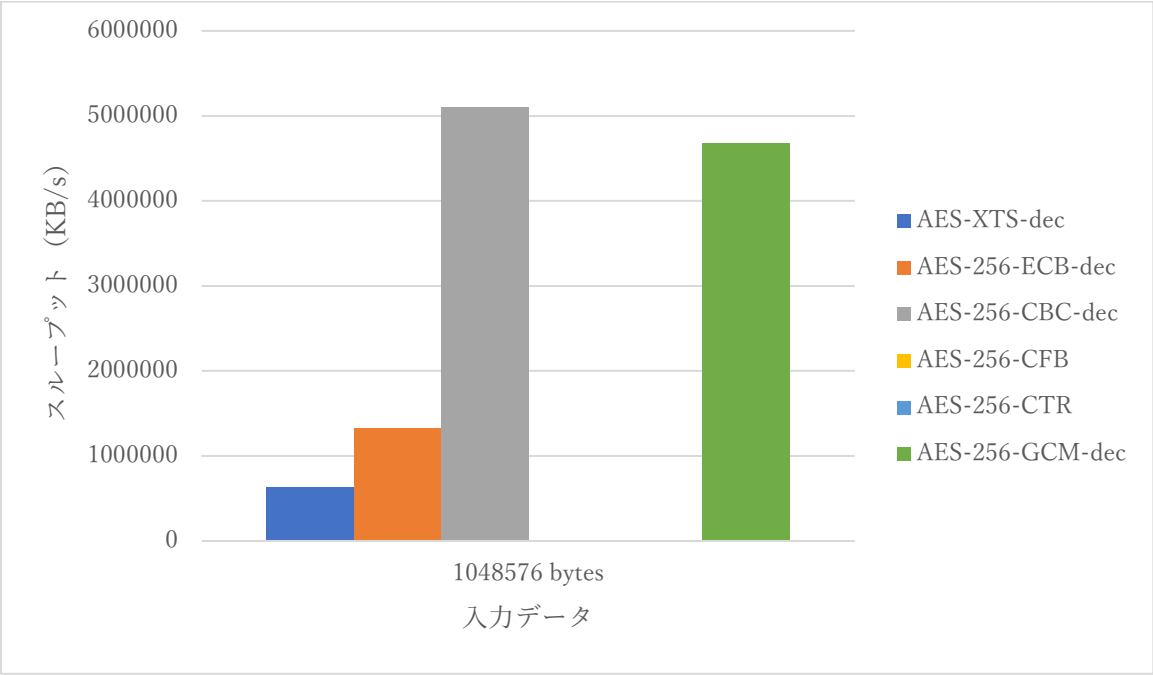


図 56　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

表 60　Linux 環境における暗号利用モード別実装性能（WolfCrypt: 256bit 復号）

	AES-XTS-dec	AES-256-ECB-dec	AES-256-CBC-dec	AES-256-CFB	AES-256-CTR	AES-256-GCM-dec
1048576 bytes	630177	1327715	5100741	0	0	4676740

（単位は KB/s）