Raspberry Piを用いたカメラ動画像のAES暗号化に対するリアルタイム処理性能評価

楊 淵[1,a)]　上竹 嘉紀[1]　小林 航也[1]　日下 卓也[1]　野上 保之[1]

概要：IoT 時代の到来に伴い, インターネットに接続する端末数が爆発的に増加すると予想されている. 計算資源に乏しい IoT 機器においても, 十分な安全性を担保することが求められており, 動画像の送受信における安全性が問題となっている. そこで本研究では,Raspberry Pi で撮影した動画像をリアルタイムで暗号化して送信する際の性能評価を行った. 検証の結果,1 秒で 30 フレームの画像を処理可能であり, 通信遅延を 1 秒に抑えた通信が可能である.

キーワード：IoT, AES, Raspberry Pi

A Performance Evaluation of Video Encryption by AES Cryptography on Raspberry Pi

YANG YUAN[1,a)]　Yoshinori Uetake[1]　Kazuya Kobayashi[1]　Takuya Kusaka[1]
Yasuyuki Nogami[1]

Abstract: With the advent of the IoT era, it is expected that the number of devices connected to the Internet will explosively increase. Even with IoT devices that have limited computing resources, there is a need to ensure sufficient security, and video data transmission has become a serious issue. In this research we evaluated the performance of transmitting and encrypting the video data in real-time based on Raspberry Pi. And as a result, we can process 30 frames per second and have only a one-second delay.

Keywords: IoT, AES, Raspberry Pi

1. はじめに

IoT 時代の到来に伴い,IoT 機器の普及が急速に進み,2018 年時点で世界中に約 70 億台もの IoT 機器が存在すると言われている. また, 今後もその端末数は爆発的に増加し,2020 年までに IoT 機器の総数は約 99 億台にも上ると予測されている [1]. こうした背景のなか, 多数の人がインターネット上に個人情報を保存しており, それらに対する情報の保護は極めて重要な課題と言える. 特に身近な IoT 機器としてウェブカメラやマイクなどが挙げられ, これらは取得した個人情報や機密情報などの重要なデータを攻撃者に不正利用される可能性がある. 例えば, 暗号化や認証などのセキュリティ対策を十分に実施していない場合, 攻撃者はインターネットを介して我々の身の回りのカメラやマイクに侵入可能であり, 情報漏えいの一因となりうる. また,Charge Coupled Device(CCD) の発展に伴い, 高解像度の画像が送信可能になったことで, 取り扱う秘密情報が増えた. これらの要因によって, 動画像を転送する際にデータを暗号化して保護する重要性が増していると言える.

本研究では,IoT 機器における動画像のセキュアな転送手法に着目し,Raspberry Pi で撮影した動画像を AES 暗号化した後に転送し, その性能評価を行った. 性能評価では, リアルタイム性と処理性能を確認するために撮影側の Raspberry Pi と受信側の PC 間における転送遅延と

[1] 岡山大学自然科学研究科
Granduate School of Natural Science and Technology, Okayama University, Japan
[a)] pmny2wnp@s.okayama-u.ac.jp

FPS(Frames Per Second, 毎秒処理可能なフレーム数) を測定した. 同時に,HLS(HTTP Live Streaming) 法 [2] を実装し, 本研究の方法と比較を行った.

評価結果として, 異なる 4 種類の画面サイズすべてにおいて遅延を 1 秒以内に抑えられることが分かった. 特に,VGA(Video Graphics Array) 基準で採用される 640 × 480 の画面サイズでは 30FPS 以上で処理可能であった.

2. AES

AES(Advanced Encryption Standard) は,DES に代わる新しい標準暗号として 2000 年に採用された共通鍵アルゴリズムである [3]. 鍵長のビット数に応じて,AES-128,AES-192,AES-256 の 3 種類に分けられる.

2.1 AES の構成

AES は行列操作を主演算として, 平文を暗号化するアルゴリズムである. この特徴は暗号化の際のメモリコストを抑制できるため, メモリ容量が限られた組み込みシステムにおいて有効的である.AES では, 平文を 128 ビット毎に区切り, さらに 8 ビット毎に区切ったものを 1 ブロックとし, これら 16 個のブロックを 4 × 4 の行列として考える. この行列は状態行列と呼ばれる.AES の行列操作は状態行列に基づいて行列の要素を変換する. **図 1** は AES の各ラウンドの流れを表したものである. 状態行列 0 に対し, 行列操作を N ラウンド行ったものは状態行列 N と呼ばれる. この状態行列 N は暗号文として出力される.

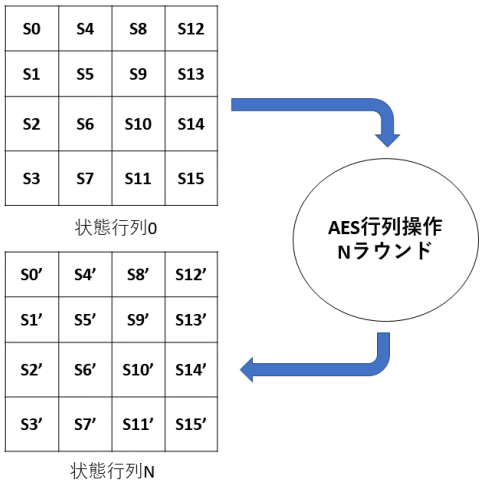

図 1 AES の流れ
Fig. 1 flow of AES

AES 暗号の行列操作は, SubBytes, ShiftRows, MixColumn, AddRoundKey の 4 つの操作に分けられる.SubBytes は S-Box による 1 バイト単位の置換操作を行う.ShiftRows は 4 バイト単位の第 $n(0 \leq n \leq 3)$ 行を n バイトのループ左シフトする. 例えば図 1 の状態行列 0 の第 1 行は $(S_1 S_5 S_9 S_{13})$ であり,1 バイトのループ左シフトした後は $(S_5 S_9 S_{13} S_1)$ になる.MixColumn は ShiftRows した行列 A を定数行列 M と式 (1) のように掛け算する.

$$A' = M \times A \tag{1}$$

この定数行列 M は式 (2) で示す. 式 (2) の数値は十六進法で表している.

$$M = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \tag{2}$$

AddRoundKey は AES を利用している鍵を状態行列の中身と XOR(排他的論理和) する.AES-128 はこれらの操作を 10 ラウンド繰り返し行うが, 最後の 1 ラウンドでは MixColumn は実行しない. **図 2** は AES-128 の流れを表したものである.

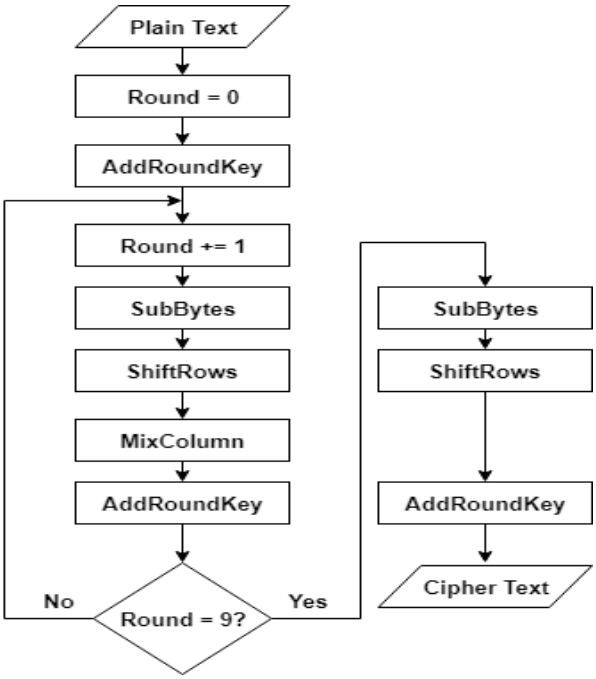

図 2 AES-128 の流れ
Fig. 2 flow of AES-128

2.2 AES の安全性

AES の AddRoundKey,ShiftRows,MixColumn は線形変換であるが,SubBytes は非線形変換であるから,AES の安全性を確保可能である.SubBytes は有限体 $GF(2^8)$ 上における 8 ビット平文の逆元を計算する. また, 現在知られる攻撃法に対して耐性をもつアフィン変換を利用した S-Box 構成手法が報告されている [3]. この S-Box をルックアップ

テーブルとして実装することで入力データを短時間のうちに非線形変換することが可能となる.S-Box の利用は AES の安全性を確保しつつ, 計算スピードも高速化可能になる。

$$f(x) * f^{-1}(x) = 1 \mod p(x) \tag{3}$$

式 (3) は逆元の定義式であり,$f^{-1}(x)$ は $f(x)$ の逆元である. AES で用いられる $p(x)$ は既約多項式と呼ばれる. これを式 (4) に示す.

$$p(x) = x^8 + x^4 + x^3 + x + 1 \tag{4}$$

8 ビットの平文を二進法で表すと,$b_7b_6b_5b_4b_3b_2b_1b_0$ のようになり, 第 n 次の項の係数を b_n とし,n 次の多項式として式 (5) のように表す.

$$\begin{aligned} f(x) =& b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 \\ &+ b_3x^3 + b_2x^2 + b_1x^1 + b_0 \end{aligned} \tag{5}$$

以上の式より 8 ビット平文 $f(x)$ の逆元である $f^{-1}(x)$ が得られる. この方法で,256 個の各逆元を求め, アフィン変換を行った後に,16×16 の行列として S-Box を構成する.

現在,AES に対する最も有効な攻撃の一つとして,2011 年に Bogdanov らが発表した Biclique 攻撃と呼ばれる攻撃法が知られている [4]. また, サイドチャネル攻撃や AES の数学的な構造を利用する攻撃方法が存在すると指摘されている [5]. しかし, いずれの攻撃法も AES 暗号の安全性を直ちに脅かすものではないと考えられており, 正しい実装を行えば運用上十分な安全性を担保できる.

3. 提案手法と実装

本研究は,IoT 機器における動画像のセキュアな転送に注目しており, **図 3** の模式図に示す手法を提案した.

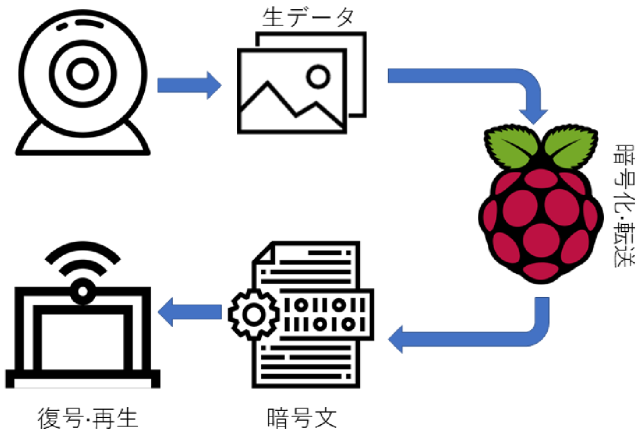

図 3 実験の流れ
Fig. 3 Flow of implement

まず,Raspberry Pi に接続されたカメラで撮影した動画像を AES で暗号化する. 次に, 暗号化したデータをクライアントで復号して再生する. 性能評価として,640 × 480/800 × 600/1024 × 768/1280 × 960 の 4 つの異なる画面サイズに対して,AES の 3 種類の鍵長における転送遅延と FPS を比較する. フレームを一枚ずつ暗号化してクライアントに送信するため, 処理速度が速くなればリアルタイムで動画像が視聴可能である. この手法では, 暗号化に相当な計算力が必要であり,Raspberry Pi の計算資源では画面のサイズが小さいほど遅延が小さくなり,FPS が高くなると予想される.

本研究で利用している圧縮技術は非可逆圧縮方式の JPEG(Joint Photographic Experts Group) であり,Raspberry Pi に実装する JPEG は 640 × 480 サイズの画像を約 50KB に圧縮可能である. この方法は画像の高速的な処理に有効な手法である.

本研究では Raspberry Pi 3B を利用し, 画像に関する処理は opencv を用いた. また,Raspberry Pi とコンピューターの通信フレームワークの構築には Flask を用いた. 利用している AES は Dwayne Litzenberger が開発した暗号化パッケージ PyCrypto の AES ツールであり [6],AES のモードは ECB モードを用いている. **表 1** は本研究で利用した各種機器と実装に用いたツールの一覧である.

表 1 機器とツール
Table 1 Devices and Tools

コンピューター	Windows10 i5-6500 CPU 3.20GHz
Raspberry Pi 3B	ARMv7 Processor rev 4 (v7l)
カメラ	Camera module for Raspberry Pi
プログラム言語	python3
画像処理ツール	opencv3
通信フレームワーク	Flask
AES	PyCrypto.Cipher.AES AES_MODE_ECB

4. 評価実験

本研究の評価実験では,3 節で提案した手法を実装し, 各状況での性能評価を行った. また, 比較対象として AES-128 を利用する HLS 手法を実装し, 提案手法の結果と比較を行った.

4.1 実験環境

実験では Raspberry Pi と通信しているコンピューターで再生する動画像の遅延と FPS の評価を行った. カメラでストップウォッチを撮影し, 同時にコンピューターで再生する画面内のストップウォッチの時間と比較することで遅延時間を測定した. また,1 秒間で処理可能なフレーム数を記録し,FPS を測定した. 測定した結果は 4.2 節で説明する. 異なるネットワークによる遅延の影響を排除するため,Raspberry Pi とクライアントを同じ LAN に接続し実験を行った.

4.2 性能評価

本研究では 640 × 480/800 × 600/1024 × 768/1280 × 960 の異なる 4 つの画面サイズを設定し, それぞれ実験を行った. そして, 画面の遅延と 1 秒に処理するフレーム数を評価対象として,AES 暗号化した画像の転送性能を評価した.

4 種類の画面サイズにおいて 3 種類の鍵長の AES を適用した評価結果を**表 2** から**表 4** にまとめる. さらに, 暗号化が動画像の転送へ及ぼす影響を明確にするため, 暗号化せずに画面転送を行った結果を**表 5** に示す.

結果より, 画面サイズが大きいほど, 遅延が大きくなり, 画面サイズが小さいほど FPS が高くなることが分かった. この結果は 3 節での予想と一致している. 今回 4 種類のサイズの動画像の転送遅延がいずれも 1 秒以内であり, 特に 640 × 480 サイズの動画像は 30FPS 以上で処理可能であることが分かった. また,1280 × 960 画面サイズの AES-256 において遅延が最大となり,0.64 秒であった.

表 2 AES-128 評価結果
Table 2 The result of AES-128 test

size(pixels)	delay(s)	FPS
640 × 480	0.19	33.7
800 × 600	0.49	18.4
1024 × 768	0.51	7.3
1280 × 960	0.54	7.2

表 3 AES-192 評価結果
Table 3 The result of AES-192 test

size(pixels)	delay(s)	FPS
640 × 480	0.20	31.7
800 × 600	0.50	18.3
1024 × 768	0.52	7.3
1280 × 960	0.59	6.9

表 4 AES-256 評価結果
Table 4 The result of AES-256 test

size(pixels)	delay(s)	FPS
640 × 480	0.25	30.9
800 × 600	0.52	17.3
1024 × 768	0.55	6.5
1280 × 960	0.64	6.3

表 5 暗号化しなかった評価結果
Table 5 The result without encryption

size(pixels)	delay(ms)	FPS
640 × 480	0.17	36.8
800 × 600	0.37	18.5
1024 × 768	0.50	8.0
1280 × 960	0.53	7.6

4.3 考察

比較対象として,HLS(HTTP Live Streaming) を Raspberry Pi 上に実装し, 評価を行った.HLS は Apple が自社 iOS 向けに開発した,HTTP ベースのストリーミングプロトコルである.HLS は録画するビデオを複数に分割し, それぞれを独立的に処理することでリアルタイムに動画像を送信するアルゴリズムである.HLS で再生する際, 自動的に生成された m3u8 形式のファイルに基づいて再生順番を決め, それに従いビデオファイルを一部分ずつ Web ページで再生する [2].HLS も AES-128 を利用して動画像の暗号化を行っている.4.2 節と同じ状況で HLS の遅延と FPS を評価した結果を**表 6** に示す.

表 6 AES-128 を用いた HLS 評価結果
Table 6 The result of HLS test using AES-128

size(pixels)	delay(s)	FPS
640 × 480	5.47	30
800 × 600	5.50	30
1024 × 768	5.53	27
1280 × 960	5.57	27

ここで,HLS の設定は 2 秒毎に 1 つのビデオファイルを生成し, 毎 6 秒で 1 サイクルとなる. ビデオファイルを順番に再生する際, 理論的には 2 秒の遅延しか生じないが, 本研究で用いた Raspberry Pi の計算力が乏しく,1 サイクルにおける三つ目のビデオファイルの生成段階で一つ目のビデオが再生可能になる. つまり,2 秒以上 6 秒未満の遅延が発生する.HLS は 2 秒毎に 1 つのビデオファイルを録画し, 暗号化した後, 転送する流れである. 本研究で提案した手法はフレームを処理しつつ転送をするのに対して, ビデオにおける処理とフレームにおける処理方法は違うため,HLS は高い FPS が得られた.

本研究と HLS ではビデオに対して処理する手法が異なるため, 実験結果には大きな相違がある. 結果から, 本研究が利用するプログラムでは低遅延, 小画面サイズにおいて高 FPS となるのに対し,HLS ではいずれの画面サイズでも高 FPS が得られるが大きな遅延が発生している.

5. まとめ

本研究は, 暗号化した動画像データを別端末に転送して, 復号再生する手法を採用し,Raspberry Pi 上に実装した. そして,3 種類の鍵長の AES 暗号を利用して,640 × 480/800 × 600/1024 × 768/1280 × 960 の画面サイズでリアルタイム画像の転送遅延と FPS の評価を行った.

性能評価の結果より,4 種類の画面サイズにおける転送遅延を 1 秒以内に抑えられることが分かり, 最大遅延が 1280 × 960 画面サイズで 0.64 秒となった. また,640 × 480 画面サイズの動画像では 30FPS 以上で処理可能であることが分かった. しかし,AES-256 では 640 × 480 サイズの転

送遅延への影響が大きいことも分かった.

本研究で用いた手法を拡張することで, 画像に識別用のラベルを埋め込んだ後に暗号化及び転送を行うことができる. これを検索可能暗号と呼ばれる暗号方式と組み合わせることで, 動画像をリアルタイムで暗号化可能かつ検索可能なシステムが開発可能であると考える. 今後の展望として,AES 暗号を利用した検索可能暗号の実装を挙げる.

参考文献

[1] https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/

[2] https://developer.apple.com/documentation/http_live_streaming/

[3] Federal Information Processing Standard Publication 197: Announcing the ADVANCED ENCRYPTION STANDARD(AES), (2001.11.26).

[4] A. Bogdanov, D. Khovratovich, and C. Rechberger: Biclique Cryptanalysis of the Full AES, ASIACRYPT 2011, pp. 344-371, (2011).

[5] 中井 綱人, 汐崎 充, 藤野 毅: 電力・電磁波解析攻撃におけるオンチップ・キャパシタの影響評価, 電子情報通信学会 (2013).

[6] https://www.dlitz.net/software/pycrypto/