音声・映像機器の暗号技術

Secure Methods for Audio and Video

田 中 裕 之 Hiroyuki Tanaka
松 尾 正 克 Masakatsu Matsuo
豊 永 三 朗 Saburo Toyonaga

要 旨

インターネットに対応した組込み機器では暗号機能が必須となっているが，暗号は処理負荷が大きく音声・映像機器では音とびや映像コマ落ちの原因となる．そこで筆者らは，音声・映像機器などのリアルタイム処理が求められる組込み機器向けに暗号モジュールを開発した．また，SSL（Secure Socket Layer）通信でサーバに集中する暗号の処理負荷を分散させる負荷分散技術や，高速なUDP（User Datagram Protocol）暗号方式を開発している．本稿では，音声・映像機器に適した暗号技術に関して，これらの取り組みを説明する．

Abstract

The embedded devices which support the Internet are required to have cryptographic functions. However, the processing load of encryption becomes a cause of dropped frames and jumpiness in audio and video devices. Accordingly, we have developed a cryptographic module for embedded devices, such as audio and video devices that require real-time processing. Furthermore, we have been developing a high-speed User Datagram Protocol (UDP) encryption method and load balancing techniques for Secure Socket Layer (SSL). In this paper, we explain some activities in relation to cryptographic technology suitable for audio and video devices.

1. はじめに

当社の主力商品の1つに，IP（Internet Protocol）電話やネットワークカメラなど，音声・映像機器がある．近年，クラウド連携などで，これら機器のインターネット通信は増加の一途である．これに伴い，強力な暗号・認証機能が，特にビジネスユースで必須になりつつある[1].

そこで当社では，組込み機器，特に音声・映像機器に適した暗号・認証技術の研究を行い，この研究成果を反映した暗号・認証モジュールを開発している．本稿では，これらのうち，暗号に関する研究を以下に解説する．

2. 要求事項

はじめに，音声・映像機器の暗号実装に対する要求事項を下記に示す．

(1) 高速化・省メモリ化

暗号は計算処理の複雑さを安全の根拠にしている．十分な安全強度を得るには，PCと同等の暗号強度が必要とされるため，組込み機器にもPC並みのパフォーマンスが求められる．しかし，音声・映像機器は，安価なCPU（Central Processing Unit）を搭載していることが多くCPU処理能力が劣るうえ，リアルタイムで大量の音声・映像データを処理するので，暗号実装では，通常の組込み機器より一層の高速化・省メモリ化が要求される．

(2) リアルタイム処理

音声・映像機器では，暗号処理中も音声・映像が途切れない，リアルタイム処理が求められる．

(3) 開発効率の向上

音声・映像機器は，性能差による派生モデルが非常に多い．そのため，暗号実装では異なる環境に対応できるように高い移植性が必要となる．

3. 当社での取り組み内容

暗号実装に対する当社のこれまでの取り組み内容を簡単に紹介する．

3.1 高速化

当社では，ソフトウェアアルゴリズムによる暗号処理の高速化手法を研究し，音声・映像機器に多い低価格・低消費電力CPUでの高速な暗号処理を実現している．この手法には企業秘密が含まれるため説明は割愛するが，**第1表**に，当社とOpenSSL（オープンソースのSSLツールキット）のRSA（Rivest Shamir Adleman）[2]鍵生成・ 暗号・復号処理時間の比較を示す．

3.2 省メモリ化

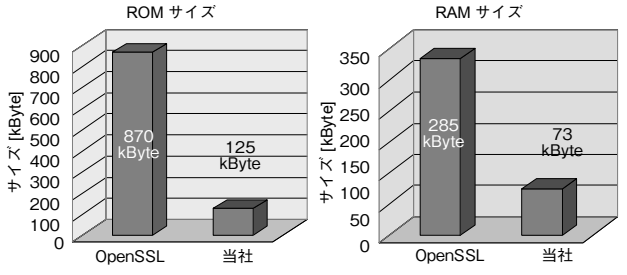
暗号通信では複数の暗号機能を組み合わせて実現しているため，多くのメモリを消費する．当社では，組み合わせる暗号機能の処理順番などを工夫し，暗号処理で使用するメモリの共用化を可能とした．また，暗号通信の機能を部品化し，搭載するモデルごとに必要な機能を必要最小限の構成になるようカスタマイズ可能にした．

第1図に，当社のSSL[3] /TLS（Transport Layer Security）[4]モジュールとOpenSSLでのROM（Read Only Memory），RAM（Random Access Memory）サイズの比較を示す.

第1表　RSA鍵生成と暗号処理時間比較
Table 1　RSA key generation and cipher speed comparison

鍵長	処理	OpenSSL	当社
1024 bit	鍵生成	10.4 s	4.13 s
	暗号	29.1 ms	3.90 ms
	復号	462 ms	144 ms
2048 bit	鍵生成	91.2 s	27.3 s
	暗号	82.8 ms	11.4 ms
	復号	2490 ms	852 ms

計測環境：UniPhier-S (注)　[MN2PS00003RF]
CPU クロック周波数：132 MHz
OpenSSL version：openssl-1.0.1c



計測環境：UniPhier－S [MN2PS00003RF]
CPU クロック周波数：CPU:132 MHz
OpenSSL version：openssl-1.0.1c
※クライアント・サーバ有効で，1 接続のみ行う場合で計測

第1図　ROM/RAMサイズ比較
Fig. 1　ROM/RAM size comparison

3.3　リアルタイム処理

音声・映像機器のOS（Operating System）は，ノンプリエンプティブ・マルチタスクが多い．このシステムでリアルタイム処理を実現するために，当社では暗号処理を短時間処理可能なブロックに分割し，このブロックの実行回数とタスクのディスパッチの実行を調整可能とする実装を行っている.

3.4　開発効率の向上

当社では，OS・CPUを仮想化して暗号モジュールが動作するようにし，OSやCPUの差異を吸収している．これにより高い移植性を実現している.

特に，暗号モジュールの一部であるSSL モジュールでは，**第2図**に示すように下位プロトコルの仮想化により，下位プロトコルの差異を吸収している．このため，TCP（Transmission Control Protocol）だけでなく，SSLモジュールに手を加えることなく，UDPやEAP（Extensible Authentication Protocol）も選択可能である．したがって，DTLS（Datagram Transport Layer Security）[5]やEAP-TLS [6]も容易に追加し，実現できる.

（注）当社の登録商標

第2図　インターフェース構成図
Fig. 2　Interface framework

さらに，上位アプリケーションとSSLモジュール間のインターフェースを，ソケットインターフェースに準拠することで，上位アプリケーションとの間でも仮想化を実現している．このため，上位アプリケーションに特殊な変更を加えることなく，SSL通信が利用できる.

これにより，コードサイズ，開発期間，テスト工数，開発コストを削減するとともに，品質向上も実現している．なお，暗号の専門知識も不要となるため，プログラマーの教育費も削減できる.

4.　当社で取り組み中の暗号技術

この章では，現在取り組み中の暗号技術を紹介する.

4.1　SSL通信の負荷分散技術

近年，音声・映像機器などのネットワーク機器でも，音声・映像のデータ解析などにクラウドを活用し，高い拡張性をもつようになっている．さらにクラウドに音声･映像データなどの情報を保存することで耐障害性をもったデータの保守も可能となる．ただし，これら多くの利点があるクラウドとの通信には音声･映像などのデータを秘守するための強力な暗号通信が必要となる．これには通常，さまざまな研究機関に評価され信頼性が高く，インターネット標準の暗号通信仕様であるSSL通信が利用される．このSSL通信における鍵データの交換には通常はRSA暗号が利用される．SSL通信プロトコルは，サーバのCPUがクライアントのCPUよりパフォーマンスが高いことを前提にしているため，SSLクライアント（機器側）に処理の軽いRSA暗号処理を，SSLサーバ（クラウド側）に処理の重いRSA復号処理を割り当てている.

しかし，サーバ（クラウド側）には非常に多数の機器が

接続されるため，この前提は必ずしも適切とは言えない．機器の接続台数が増加すれば，サーバはRSA復号処理に忙殺される．これは，サーバの台数を増やしたり，RSA復号処理を代行するSSLアクセラレータを導入したりすることで緩和できるが，それではサーバ運用コストの増大や，クラウドの利点である拡張性を失う可能性もある．

この問題点を解決するため，当社では，ソフトウェアによるSSL通信の負荷分散技術を研究している．

〔1〕SSL通信を負荷分散するための課題

クラウドのRSA復号処理を機器に代行させれば，SSL通信の負荷分散を安価に実現できる．しかし，単純にクラウドをSSLクライアント，機器をSSLサーバとすれば，以下の2つの課題が生じる．

(1) NAT（Network Address Translation）超え

機器から通信を開始しないと，機器のIPアドレスやポート番号が決定されないので通信できない．

(2) ファイアーウォール超え

クラウドから通信を開始すると，ファイアーウォールがこれを攻撃と見なし通信を遮断する．また，登録されていない通信プロトコルなどを制限するファイアーウォールも存在する．

〔2〕当社の負荷分散方式の実現手段

当社は，先に示した負荷分散技術における課題を以下の手順で解決しようとしている．

① **第3図**に示すように，クライアント（機器）は，ClientHelloメッセージで，通常の暗号方式と同時に，ベンダーユニーク値を利用して，『非暗号』を暗号方式としてサーバに提示する．

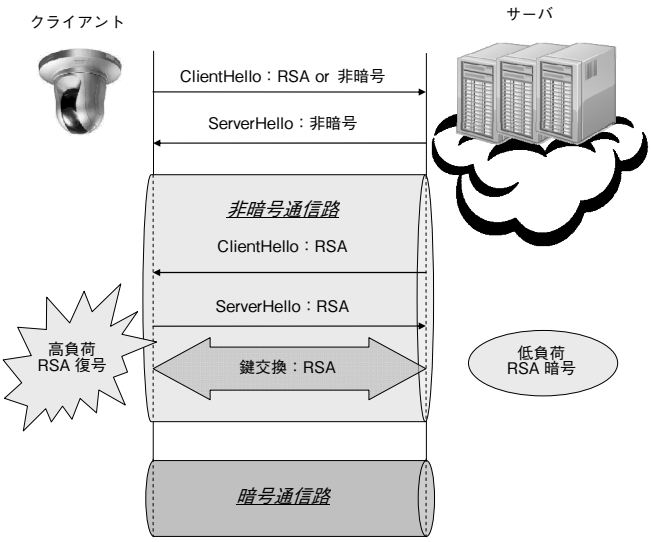
② 次に，サーバ（クラウド）は，過剰な負荷が生じた場合に，ServerHelloメッセージで『非暗号』の利用をクライアントに指示する．

③ この後，サーバ・クライアントは通常のSSLハンドシェークを行うが，『非暗号』なのでRSA暗号・復号処理は行われず，『非暗号』の暗号通信路が確立される．

④ 次に，この暗号通信路でクラウドをSSLクライアント，機器をSSLサーバとして，通常のSSLハンドシェークを実行する．これは第三者には，暗号通信路上のデータ送受信に見えるため，ここでもファイアーウォール超えの問題は生じない．

このように『非暗号』でSSL通信を確立し，この通信路上で逆向きのSSL通信を行えば，安全・安価にSSL通信の負荷分散を実現できる．

ただしこの手法では，機器の負担増加が新たな課題となる．しかし，3.1節で説明したように，当社の暗号モジュールであれば十分高速にこれに対応可能である．また，このプログラミングをOpenSSLで行うと煩雑になるのに対し，3.4節で説明したように，当社のSSLモジュールでは，アプリケーションの特殊な変更は不要である．このように，この手法は3章で説明した取り組み成果がベースとなっている．



第3図 SSL負荷分散
Fig. 3 SSL load balancing

〔3〕当社負荷分散技術の評価

ここでは，先に示した実現手段で構築した当社の負荷分散技術を評価する．

(1) DoS（Denial of Service）攻撃に対する安全性

SSL接続要求を執拗（しつよう）に行うDoS攻撃や，SSL接続の確立後に再ネゴシエーション要求を連続して行うDoS攻撃は，サーバの負荷が著しく増大するため，サーバダウンの原因になるなど，大きな脅威となっている．本手法を用いれば，SSL接続に伴うサーバ側の負荷が軽いので，この被害を軽減できる．

(2) なりすまし攻撃に対する安全性

『非暗号』のSSLハンドシェーク中では，SSLサーバ証明書を検証しないため，サーバ認証が実施されない．これを補うには，逆向きSSLハンドシェーク中でサーバをクライアントとしたクライアント認証を実施すれば良いが，それでは，結局，サーバ側でRSA復号処理が動作することになり，負荷分散が成り立たない．そこでこれを解決すべく，現在，なりすまし攻撃を排除可能なパスワード認証の研究開発を進めている．

4.2 UDP暗号技術

音声・映像機器では，リアルタイム性を求められるため，TCPと比べて高速なUDP通信が利用される．

そのため，当社では UDP暗号技術の研究を行い，3.3節で

説明したリアルタイム性のさらなる向上に努めている.

UDP暗号技術には処理が高速な, 共通鍵暗号が適している. 共通鍵暗号方式には大きく分けてストリーム暗号とブロック暗号があるが, UDP暗号技術として用いる場合にはそれぞれ以下に示す問題点がある.

(1) ストリーム暗号の問題点

ストリーム暗号とは, 擬似乱数生成器で乱数列（鍵データ）R = {R1, R2 …}を生成し, （1）式に示すように, これと平文データM = {M1, M2 …}とをXOR （eXclusive OR）処理して, 暗号データC = {C1, C2 …}を生成する暗号化方式である. 復号は, （2）式に示すように, この逆の処理を行う.

《暗号》 $Ci = Mi \oplus Ri \quad (i = 1, 2, \ldots)$ ・・・・・・・・・・ （1）

《復号》 $Mi = Ci \oplus Ri \quad (i = 1, 2, \ldots)$ ・・・・・・・・・・ （2）

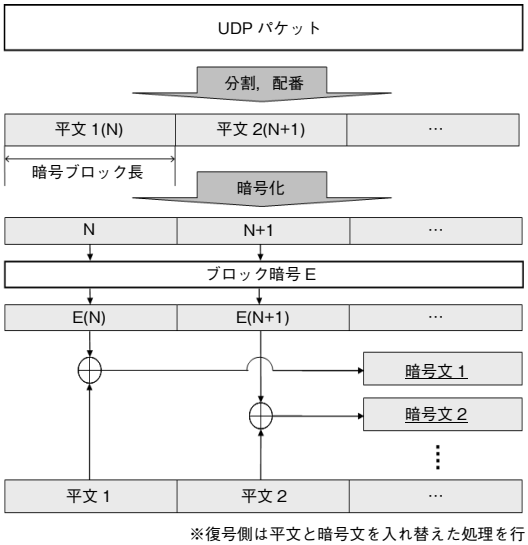
（注）⊕は排他的論理和を示す

しかし, ストリーム暗号は単純にUDP暗号に利用できない. 例えば, N番目のUDPパケットがロスした場合, 受信側では, N番目のUDPパケットに適用すべき乱数列で, N+1番目のUDPパケットを復号してしまい, 復号に失敗する.

(2) ブロック暗号のカウンタモードの問題点

パケットロスの対策として, UDP暗号には通常, ストリーム暗号の一種であるブロック暗号のカウンタモードが利用される.

第4図に, この具体的処理の1例を示す. 各UDPパケットをブロックサイズごとに分割し, 各ブロックにN, N+1 …とユニークなカウンタ番号を振る. このカウンタ番号を, ブロック暗号で暗号化し, この暗号化されたカウンタ番号を乱数列として, ストリーム暗号を行う. このように, この手法においては, ブロック暗号はストリーム暗号における乱数生成器の役割を果たす.



第4図 ブロック暗号カウンタモード
Fig. 4 Block cipher counter mode

この手法では, ブロックごとに鍵データの同期を合わせるので, パケットロスが生じても, 鍵データの同期ずれは起こらず, 復号に失敗しない. しかし, ブロック暗号を幾度も利用するため高速処理は難しい.

〔1〕UDP暗号技術実装の課題

先に示した各共通鍵暗号方式の問題点に示すとおり, UDP暗号技術の実装には大きく2つの課題がある.

① リアルタイム性の向上には高速処理が必要となる.

② UDP通信にはTCP通信のような再送制御がなく, パケットロスを考慮する必要がある.

〔2〕当社のUDP暗号方式実現手段

当社は, この課題を以下の手法で解決しようとしている.

① UDPパケットの暗号・復号には, 高速処理が可能なストリーム暗号を利用する.

② パケットロスによる鍵データの同期ずれを防止するため, パケットごとに乱数生成器をリセット（新たなシードを入力）し, パケットごとに鍵データの同期を合わせる.

③ 予測不可能なシードを生成するため, シードはブロック暗号のカウンタモードで生成する. なお, ブロック暗号はブロック単位ではなく, パケット単位で利用するので, 処理速度への影響は小さい.

ただしこの手法では, ストリーム暗号とブロック暗号の状態を通信セッションごとに記憶しなくてならず, 多数の通信セッションを利用する音声・映像機器では多くのメモリを消費する. しかし, 3.2節で説明したように, 当社の暗号モジュールは省メモリ化されていて, 少ないメモリで対応できる. また, 3.4節で説明したように, 当社のSSLモジュールであれば, 下位プロトコルにUDPを利用できるので, 鍵交換を含めて, プログラミングは容易である. このように, この手法も3章で説明した取り組み成果がベースとなっている.

次に, **第5図**にこの手法の具体的処理の1例を示す.

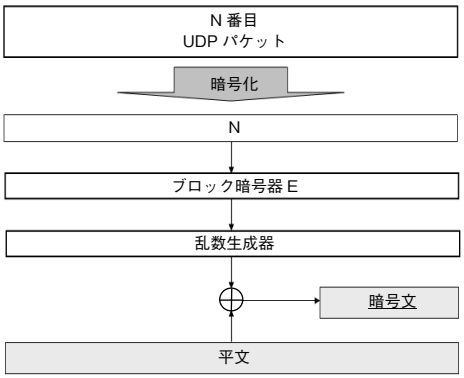
① 送信側は, パケットごとにカウンタ番号を付与し, これをUDPパケットに付加する. なお, このカウンタ番号はパケットごとにユニークな値であればどのような値でも良い.

② カウンタ番号をブロック暗号で暗号化する.

③ この暗号化されたカウンタ値を, 乱数生成器のシードとして設定する. なお, 乱数生成器のリセットを複数パケットごとに行うことでさらに高速化が可能である.

④ 乱数生成器が生成した乱数列と平文データとをXOR処理し, これをUDPパケットに付加する.

⑤ 受信側は，UDPパケットに付加されたカウンタ番号を取得し，ブロック暗号の暗号器で暗号化する．
⑥ この暗号化されたカウンタ値を乱数生成器のシードとして設定する．
⑦ 乱数生成器が生成した乱数列とUDPパケットの暗号データとをXOR処理して平文データを取得する．



第5図　当社のUDP暗号方式
Fig. 5　Our UDP cryptographic method

なお，本手法は，SRTP（Secure Real-time Transport Protocol）[7]やDTLSにおける1つの暗号スイートとして実装可能である．また，鍵交換に関しては前述のDTLSによる鍵交換のほかにも，独自の鍵交換方式や，MIKEY（Multimedia Internet KEYing）[8]による鍵交換によって実現が可能である．

〔3〕当社UDP暗号方式の評価

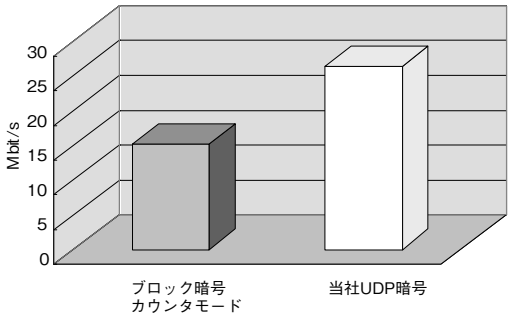
筆者らは，安全性を損なわないようにするため，ブロック暗号ではAES暗号，ストリーム暗号ではArcfour暗号というような安全性の高い，逆に言えば処理の重い暗号方式を採用した。このような対応を行ったとしても，なおブロック暗号だけの暗号方式に比べれば高速に処理可能である．実際，**第6図**に示すように，AES暗号のカウンタモードに比べれば，本方式は，約1.7倍高速である．

ただし，実績のあるストリーム暗号でも，乱数生成器のリセット直後は，乱数パターンに偏りがあることが知られているので，実際の実装においては，リセット後の最初の数百バイトの乱数を破棄するようにした．

5. まとめ

当社は，組込み機器，特に音声・映像機器に適した暗号・認証技術の研究を行うことで，安全，かつコストパフォーマンスの高い暗号・認証モジュールを開発してきた．

当初は，この暗号・認証モジュールは音声・映像機器向けであったが，コードサイズ，開発期間，テスト工数，開発コストなどの削減効果が大きく，今では，当社のさまざまなネットワーク商品に，このモジュールが搭載されるに至った．近年では暗号の2010年問題にもいち早く対応するなど，暗号強度の向上にも努め，現在では，当社のクラウド戦略を支える1つの基盤になっている．



計測環境：MIPS32 bit（200 MHz）
ストリーミング暗号：Arcfour
ブロック暗号：AES(Advanced Encryption Standard)
パケットサイズ：1024 Bytes

第6図　処理速度比較
Fig. 6　Processing speed comparison

これからは，現在研究中のSSL通信の負荷分散技術やUDP暗号技術など，新たな暗号方式の提案も行い，より安全で快適なネットワーク・クラウドソリューションを実現したい．

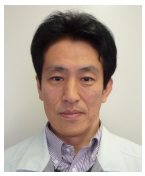
また，研究成果を異なる用途に活用することで活躍の場を広げたいと考えている．例えば，最近はBYOD（Bring Your Own Device）ソリューションへの期待が膨らみつつある．これには端末認証が必須で，そのためには高速な鍵生成処理が必須となる．当社の高速な鍵生成処理技術であれば，これに貢献できると考えている．また，モバイル端末では，暗号処理に要する消費電力を削減することで，駆動時間の改善も図りたい．すでに一部で展開しているが，さらに多くのモバイル端末に当社の暗号モジュールを搭載していきたい．

参考文献

[1] 警察庁，“平成23年中のサイバー犯罪の検挙状況等について，”平成24年3月15日 広報資料．http://www.npa.go.jp/cyber/statics/h23/pdf01.pdf, 参照 Oct. 21. 2013.
[2] Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, IETF RFC 3447, pp. 6-27, 2003.
[3] The Secure Sockets Layer (SSL) Protocol Version 3.0, IETF RFC 6101, pp.12-39, 2011.
[4] The Transport Layer Security (TLS) Protocol Version1.2, IETF

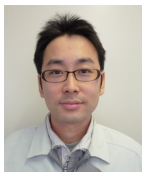
RFC 5246, pp.15-65, 2008.
[5] Datagram Transport Layer Security Version 1.2, IETF RFC 6347, pp.5-27, 2012.
[6] The EAP-TLS Authentication Protocol, IETF RFC 5216, pp.4-23, 2008.
[7] The Secure Real-time Transport Protocol (SRTP), IETF RFC 3711, pp.5-33, 2004.
[8] MIKEY: Multimedia Internet KEYing, IETF RFC 3830, pp.7-52, 2004.

執筆者紹介



田中　裕之　　Hiroyuki Tanaka
パナソニック システムネットワークス（株）
先行技術開発センター
Advanced Technology Development Center,
Panasonic System Networks Co., Ltd.

松尾　正克　　Masakatsu Matsuo
パナソニック システムネットワークス（株）
先行技術開発センター
Advanced Technology Development Center,
Panasonic System Networks Co., Ltd.



豊永　三朗　　Saburo Toyonaga
パナソニック システムネットワークス（株）
先行技術開発センター
Advanced Technology Development Center,
Panasonic System Networks Co., Ltd.