CRYPTREC 暗号技術ガイドライン
(SSL/TLS における近年の攻撃への対応)

平成 26 年 3 月

独立行政法人情報通信研究機構
独立行政法人情報処理推進機構

目次

1. 序章

1.1 本ガイドラインの目的

SSL/TLS について、近年、プロトコルの仕組みの脆弱性やソフトウェアの脆弱性を複合的に利用する攻撃がいくつか公開されている。また、プロトコル内で用いる暗号として RC4 を選択することができるが、RC4 は運用監視暗号リストに位置づけられており、安全性に係る問題のある暗号技術として、互換性維持以外の目的での利用が推奨されていない。さらに、RC4 に対する攻撃が適用できる環境下では SSL/TLS の安全性が保てなくなることが示されている。このような状況を踏まえ、本ガイドラインでは、近年示されている攻撃の解説を行うとともに、SSL/TLS を安全に利用するため近年注目されている攻撃に対して推奨される対応を示すことを目的としている。

本文では、プロトコルの仕組みを利用した攻撃として、BEAST、TIME、CRIME、Lucky Thirteen などについて解説するとともに推奨される対応策を示す。また、プロトコル内で用いる暗号として RC4 を用いた場合の実際の攻撃方法、事例を示す。この場合は攻撃を回避する効果的な対応策がないため、RC4 を選択しない利用方法の推奨などを述べている。

1.2 総論

SSL/TLS に関して、 (1) プロトコルの仕組みを利用した攻撃に起因する脆弱性と、 (2) プロトコル内で用いる暗号として RC4 を用いた場合に、RC4 のアルゴリズムの弱さに起因する脆弱性とが指摘されている。

(1) に分類される脆弱性：BEAST は、プロトコルで CBC モードを用いた場合に CBC モードの脆弱性として知られる特性を利用した攻撃である。具体的には、特定のブロックの平文を意図した値に差し替えられる攻撃者が、別のブロックの解読が容易になるという脆弱な性質を利用しており、 SSL/TLS のプロトコルの仕様との複合的事象として、Java アプレット実行環境が脆弱なブラウザにおいて攻撃が発生することが指摘されている。ただし､プロトコルそのものを変更しなくても平文を 1 対 (N-1) の分割を行うことで回避できる可能性が示されている。また、Java アプレットのパッチを当てることでも回避することができるとされている。これらの状況から、この攻撃をもって、 SSL/TLS において、ブロック暗号を利用しないという結論には至らない。 CRIME、TIME、BREACH、Lucky Thirteen は、圧縮データのサイズの差異や、実行時間の差異を利用して暗号解読の攻撃を行う、いわゆる実装攻撃に属する攻撃であるが、これらの攻撃は、一般の実装攻撃への対策と同様の考え方で、圧縮機能の無効化、データや実行時間の平準化やランダム化などの回避策が示されている。その他、圧縮機能を無効化せずに，回避する方法も検討されはじめている。 これらの攻撃を鑑みても SSL/TLS において、ブロック暗号を利用しないという結論には至らない。

(2) に分類される脆弱性： RC4 は、同じデータに対して異なる鍵を用いて生成された暗

号文を複数入手できる Broadcast Setting や同じデータをセッションごとに同じ位置で、異なる鍵で暗号化して送信する Multi-Session Setting の環境が攻撃者に与えられた場合、効率的に攻撃が実現できることが知られている。 SSL/TLS で用いる暗号として RC4 を選択した場合、攻撃者に効率的に RC4 に対する攻撃が適用できる環境を提供してしまうことになる。近年の解析結果では、現実的なコスト、および起こりうる確率で平文が回復できることが示されている。(一例としては、同じメッセージに対して 2^{34} の暗号文が集められた場合、メッセージの先頭から約 1000 T byte を非常に高い確率（0.97）で復元可能であることが示されている［1]）。RC4 の攻撃を適用できる環境として利用されている Broadcast Setting は、BEAST、TIME、CRIME 等の攻撃の中でも利用されており、この攻撃のみで想定している特殊な環境ではない。また、 HTTPS + basic 認証（例：ネットワーク利用者認証、グループ利用の Web ページ）を利用する際に攻撃者に繰り返し re-negotiation をさせられてしまう場合や JavaScript のバグを攻撃者が悪用し、攻撃者のサーバに大量の暗号文を送らされてしまう場合等には、比較的容易に整えられる環境であり、PC 版の Internet Explorer、 Firefox、 Opera、 Safari などのブラウザに対してブロードキャスト状態にするのは十分に実行可能な設定条件であるといえる。ゆえに、RC4 を用いた場合の解析結果は現実的な脅威として配慮すべきである。

SSL/TLS にはいくつかのバージョンが存在する。推奨される設定として、TLS 1.0 より古いバージョンについては、新しいバージョンへアップデートすることが推奨される。TLS 1.0 については、CBC モードを用いた場合の脆弱性に対してパッチが提供されているため、Java 等のソフトウェアを最新版に更新した上で、CBC モードを選択することが推奨される。TLS 1.1 については、CBC モードを用いた場合の脆弱性が解消されていることから、CBC モードを選択することが推奨される。TLS1.2 については、CBC モード、CCM モード、GCM モードが選択できるため、それらを使うことが推奨される。

1.3 本ガイドラインの構成

2 章に、本文中で取り扱っている技術説明/用語説明を記す。3 章に、プロトコルの仕組みを利用した攻撃を記す。4 章に、RC4 の脆弱性に基づく攻撃を記す。

1.4 注意事項

本ガイドラインは状況の変化に伴い、改訂される場合がある。

2. 技術説明 / 用語説明

SSL/TLS

SSL（Secure Socket Layer）、TLS（Transport Layer Security）は、ネットワーク上のアプリケーションに対して通信相手の認証と暗号化された通信を提供するプロトコル。SSL は Netscape Communications 社が開発し、その仕様を引き継ぐ形で IETF において TLS として標準化されている。

https

アプリケーションにおいて、SSL/TLS を用いて通信を行う際に使われる URI のスキームのこと。

Deflate

可逆データ圧縮アルゴリズムである。SSL などのプロトコル内の圧縮で使われるケースでは 16KB ごとの境界が存在するため、攻撃対象の Cookie の値がちょうどこの境界上に来るようにパディングのサイズを調節することによる 1 バイトずつのブルートフォース攻撃などに利用される。

Cookie

HTTP プロトコルで通信する、ウェブブラウザとウェブクライアントの間で、主に状態管理のために情報を保存するために使われるプロトコル、およびこのプロトコルによって保存される情報そのもの。

3. プロトコルの仕組みを利用した攻撃

3.1 CBC モードの構成を利用した攻撃：BEAST

BEAST [2] は 2011 年に開発された攻撃ツールであり、SSL3.0/TLS1.0 の CBC モードの脆弱性を利用して選択平文攻撃を行い、Cookie（平文）を得る。BEAST の概要は公開されているが、ツールは非公開のため、詳細については不明な部分が多く、以降の説明には一部推測が含まれる。

まず、図 1　に、SSL3.0/TLS1.0 における CBC モードの処理概要を示す。

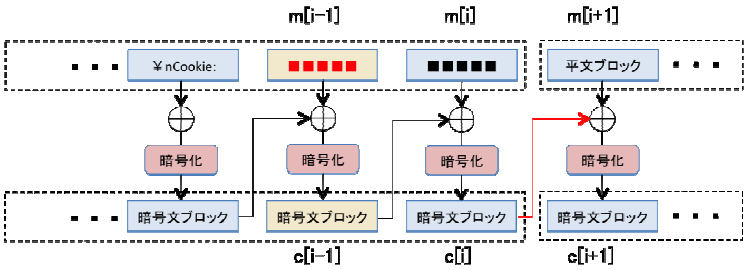

図 1　SSL3.0/TLS1.0 における CBC の概要

ここでのポイントは、初期化ベクトルとして直前の最終暗号文ブロック c[i] を使用している点である。これにより、一つ前の平文ブロック m[i-1]（Cookie に対応）に対して以下の選択平文攻撃が可能となる。

1. 攻撃者は平文 m[i-1] の推測 M[i-1] を生成
2. 次の平文の最初のブロックとして M[i+1]=M[i-1] XOR c[i-1] XOR c[i] を設定
3. 対応する暗号文 C[i+1] と c[i-1] を比較
4. 異なっていれば、1 からやり直し、なお、

C[i+1]=E(M[i+1] XOR c[i])
=E(M[i-1] XOR c[i-1])
=E(m[i-1] XOR c[i-1]), if M[i-1]=m[i-1]
=c[i-1]

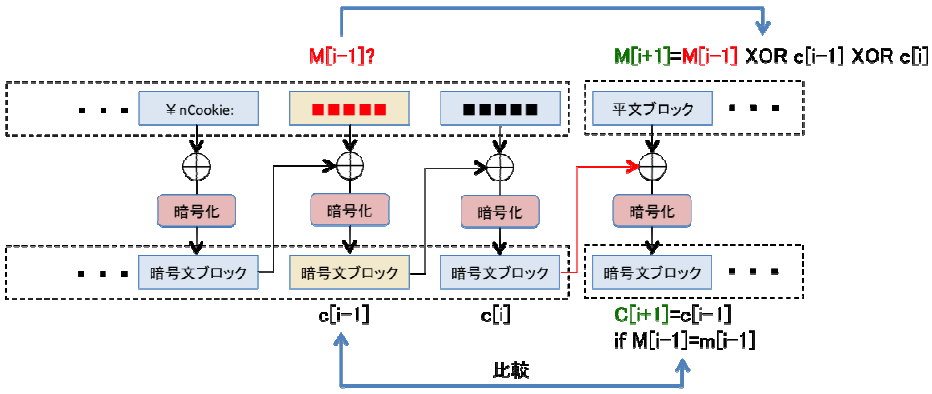

図 2　SSL3.0/TLS1.0 への平文選択攻撃

図 2 の攻撃では、平文ブロックをブロック全体で全数探索しており、特定に（最大）2^{128} 回の平文選択が必要となり脅威は小さい。

それに対して BEAST では攻撃の効率向上のため、ブロック単位ではなくバイト単位で全数探索することで、特定に必要な平文選択を $2^8 \times 16$ と大幅に削減した。具体的には、アクセス先の URL を変更し、図 3 に示すように Cookie の 1 バイト目が平文ブロックの最後となるようにした上で、選択平文攻撃でこの 1 バイトを特定する。そしてさらに、URL を 1 バイト短くすることで、Cookie の 2 バイト目がブロックの最後となるようにし、同様に処理を繰り返し、バイト単位で特定する。これにより、攻撃の効率が飛躍的に向上し、実際に適用可能となった。

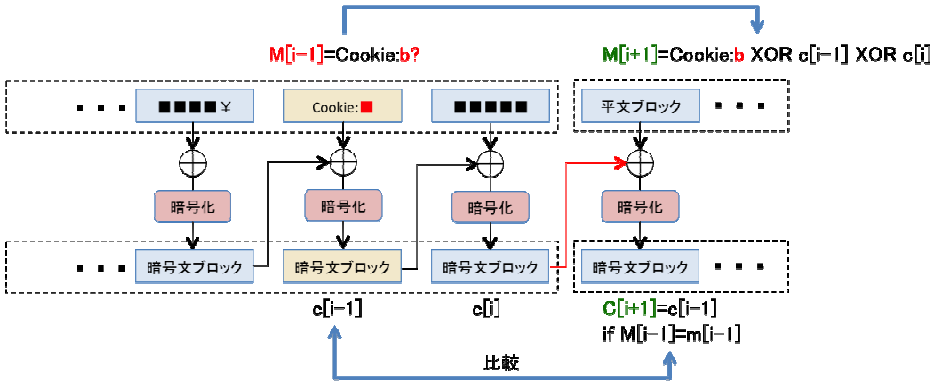

図 3　BEAST におけるバイト単位の平文選択攻撃

BEAST への対策には、TLS1.0 で実施可能な対策と、TLS1.1 以降への移行で実施可能となる対策の 2 種類がある。前者の対策として、セキュリティパッチの適用が挙げられる。現時点のセキュリティパッチ [3]（1/n-1 レコード分割，1/n-1 Record Splitting Patch と呼ばれる）については、その安全性が [4]で評価されており、ある条件下で BEAST 系の平文選択攻撃に対して識別不能性を満たすことが証明されている。ここで条件には、CBC モードでの暗号化の前に平文に付け加えられる MAC（後述の 3.3 節の図 6 参照）の長さがブロ

ック長より短いことが含まれる。よって、ブロック長より長い MAC を生成する Truncated HMAC（RFC6066 [5]）を使う場合には、必ずしも安全性が保証されるわけではないため、注意が必要である。一方、後者の TLS1.1 以降で実施可能な対策として、改良された CBC モード（初期化ベクトルに直前のブロックは使わず、リフレッシュする）の使用が挙げられ、さらに TLS1.2 以降であれば、新たに追加された認証付き秘匿モード(GCM モード、CCM モード)の使用も対策となる。なお、BEAST のデモでは、実装に Java アプレットが使用されているが、その理由は Java アプレットの脆弱性を使用するためと言われている。よって、上述のいずれの対策でも、Java を最新に保つことが不可欠となる。

短期的な対策として共通鍵ブロック暗号の代わりにストリーム暗号 RC4 を使うことが挙げられるが、RC4 の脆弱性が数多く報告されており、長期的な対策としては推奨できない。

3.2 圧縮処理部分の観測に基づく攻撃

3.2.1 CRIME

CRIME（Compression Ratio Info-Leak Mass Exploitation） [6] は、2012 年のセキュリティカンファレンス Ekoparty において、Rizzo と Duong によって発表された攻撃である。SSL/TLS において、入力データに対する圧縮後のパケット長の違いから平文である Cookie を解読する攻撃である。一般的にデータ圧縮の技術では、頻度が高いデータが多いほど圧縮後のデータ長は短くなる。この性質を利用し、圧縮後のメッセージの長さを参照しながら解読を行う。解読の例を図 4 に示す。

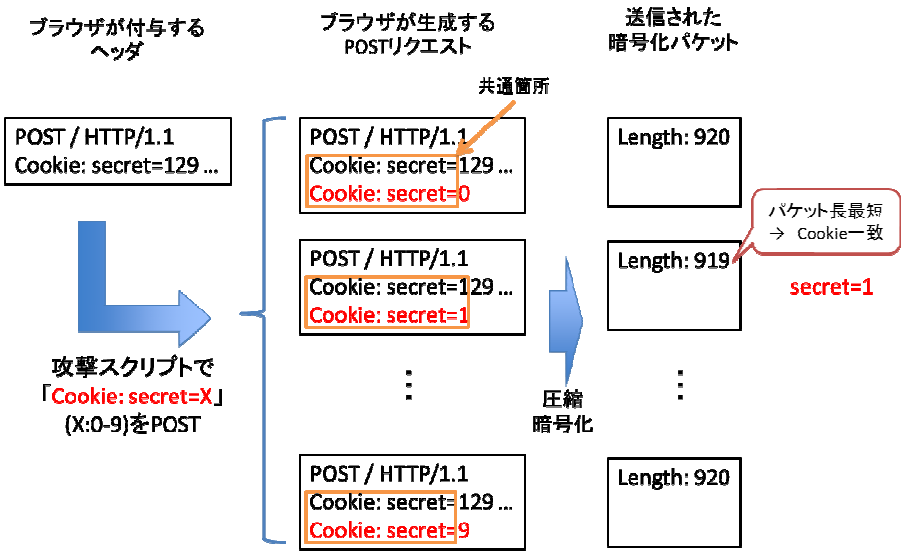

図 4 CRIME の攻撃の流れ

この例では、 Cookie の中に、secret という属性値がセットされているが、攻撃スクリ

プトを利用し、secret=X という形で X を 0 から 9 に変化させたデータを SSL/TLS のデータとして送る。その結果として圧縮されたデータのパケット長から、secret の値を類推することができる。

この攻撃は、SSL/TLS において使われているデータ圧縮の機能に依存するものであり、SSL/TLS の圧縮機能を使わないことで対応できる。Web ブラウザの Internet Explorer ではもともと圧縮機能に対応していなかったため本攻撃は適用が出来なかった。また、Google Chrome ではバージョン 21.0.1180.89、Firefox ではバージョン 15.0.1、Opera では 12.01、Safari ではバージョン 5.1.7（Windows）、5.1.6（MacOS）で圧縮機能が無効化されており、これら以降のバージョンでは本攻撃の影響はない。

また、 Web サーバソフトウエアの Apache2.2 with MOD_SSL ではデフォルトで圧縮機能を利用しており機能の無効化の設定はないが、 Apache2.4 with MOD_SSL ではデフォルトで圧縮機能を利用しているものの無効化も可能となっている。また、Microsoft IIS (Internet Information Services) ではもともとすべてのバージョンで圧縮機能が存在せず、Amazon ELB (Elastic Load Balancing) ではデフォルトで圧縮機能は無効となっている。

3.2.2 TIME

TIME（Time Info-leak Made Easy） [7] は 2013 年に Liu らによって発表された攻撃で、 CRIME と同様に、 SSL/TLS の圧縮機能を用いて Cookie などの値を解読する攻撃である。図 5 に攻撃の流れを示す。 CRIME がデータ長を推定に利用したことに対して、TIME ではブラウザにおいての処理時間の差によって攻撃に必要な情報を収集するため、攻撃者による中間者攻撃が必要であった CRIME に比べて、攻撃の実現性が高いことが特徴である。TIME では、HTTP レスポンスの圧縮結果を用いていることが攻撃の原因となっており、圧縮機能の無効化が攻撃を回避する有効な方法である。しかし、現実のアプリケーションにおいては性能上要件により圧縮機能の無効化が受け入れられない場合があり、このような場合においては HTTP レスポンスの圧縮を無効化という対策を講じることは難しいのが現状である。

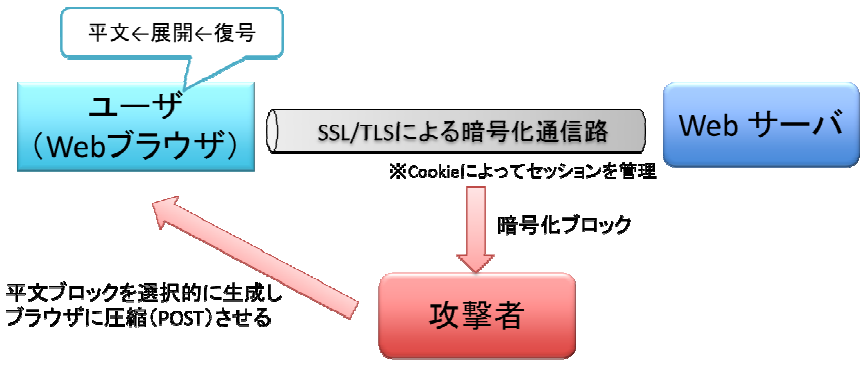

図 5 TIME の攻撃の流れ

3.2.3 BREACH

BREACH [8] は、2013 年に行われた BlackHat において Prado らによって発表された攻撃である。基本的な考え方は、 CRIME と同様に HTTP リクエストメッセージをコントロールして、圧縮データのデータ長の違いにより暗号文を解読する攻撃である。CRIME との違いは、 http レスポンスに含まれる情報を奪うことと、 SSL/TLS のデータ圧縮機能を用いるのではなく、アプリケーション層における圧縮機能、例えば Web アプリケーションによる gzip を用いた圧縮においても攻撃が成功するため、SSL/TLS の設定変更では対策にならないという点である。一方で、攻撃成功の条件は限定的であり、gzip 圧縮の他に、レスポンスの平文にリクエストの情報そのものと、レスポンス自体に CRSF Token などの秘密情報が含まれることが必要である。前述の通り、SSL/TLS の設定変更では対処できないため、 SSL/TLS を用いるアプリケーションでの対応が必要となる。

3.3 MAC-then-Encryption の構成を利用した攻撃：Lucky Thirteen

Lucky Thirteen [9] は 2013 年に発見された TLS が使用する HMAC 付き CBC モード(MAC-then-Encrypt、以下 MEE-CBC-TLS) の脆弱性を利用した中間者攻撃であり、攻撃者は復号処理の処理時間差(ハッシュ関数の計算回数の違い)を特定することで平文を得る。図 6 に、TLS における MEE-CBC-TLS の処理概要を示す。

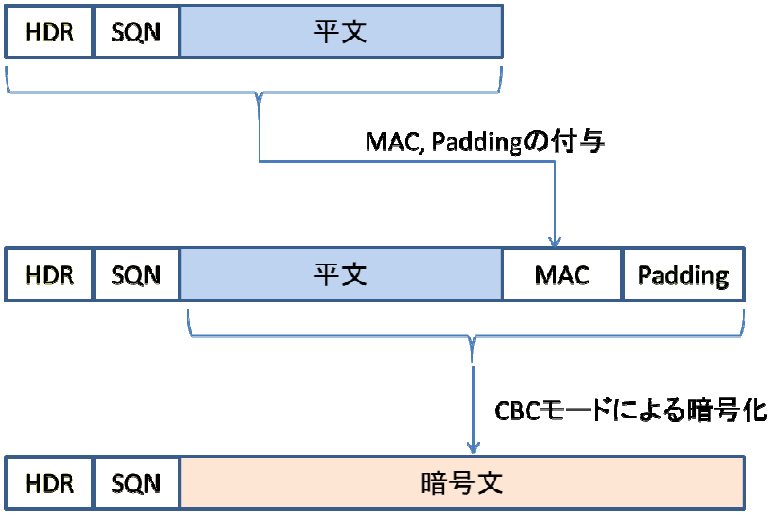

図 6 TLS における MEE-CBC-TLS の概要

ここで MAC の生成に HMAC-SHA1 を用いる場合、ハッシュ対象のデータ長により MAC の生成に用いる SHA-1 の実行回数が異なることが知られている。 SHA-1 の処理回数は[((64+M) +1+8)/64]で表現されるため、具体的には M mod 64 が 55 か 56 になるかで SHA-1 実行回数が変化する。

実際の攻撃は次の通りである。

1. 暗号文を入手する
2. MAC エラーが発生するような攻撃用の暗号文を用意し、サーバに送付する
3. 意図的に MAC エラーを発生させ、エラー発生のタイミング（SHA-1 実行回数の変化）から平文を得る
4. エラー発生箇所を変更し、2、3 を繰り返す

図 6 に示す通り、MAC の対象は平文にヘッダ（HDR）5 byte とシーケンス番号（SQN）8 byte の合計 13 byte を加えたデータとなるため、この 13 byte を加えた平文ブロックのデータ長を変化させる。

また、実際に攻撃を行うためには、攻撃者はネットワーク越しに MEE-CBC-TLS 復号の処理時間差を厳密に測定する必要があるため、実際に攻撃を適用することは難しい。Lucky Thirteen の提案者は、 OpenSSL 及び GnuTLS を使用しているサーバに対して同一セグメント内からの攻撃に成功した実験結果を示している。

Lucky Thirteen に対する対策としては、認証付き暗号利用モード(GCM モード、CCM モード）を利用することである。これは TLS 1.2 以降でサポートされている。

3.4 Renegotiation を利用した攻撃

3.4.1 攻撃方法

Renegotiation を利用した攻撃とは、2009 年に発見された SSL/TLS のハンドシェイクにおいて確立された暗号アルゴリズムと鍵長を更新(Renegotiation)する際の脆弱性を利用した中間者攻撃である [10]。

Renegotiation は、SSL/TLS が確立され暗号通信を行っているセッションを更新して、新たにセッションを確立させる手法である。Renegotiation の概要を図 7 に示す。Renegotiation は最初のハンドシェイクにおいて確立された暗号チャネルを使用して、新規にハンドシェイクを行うことで実施されるため、新たに確立された暗号チャネルが既存の暗号チャネルに置き換わる。なお、---は平文データ、===は暗号化データを示す。

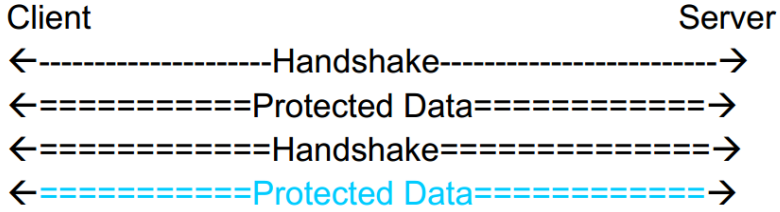

図 7 Renegotiation の概要（[11]より引用）

実際の攻撃は次の通りである。

1. 攻撃者はクライアントのハンドシェイクを受信し、パケットを保持しておく
2. 攻撃者とサーバの間で通常のハンドシェイクを行い、サーバと暗号通信を行う
3. 攻撃者は Renegotiation を要求し、クライアントとサーバの間でのハンドシェイクに対して、1 で保持していたパケットをサーバに送信する

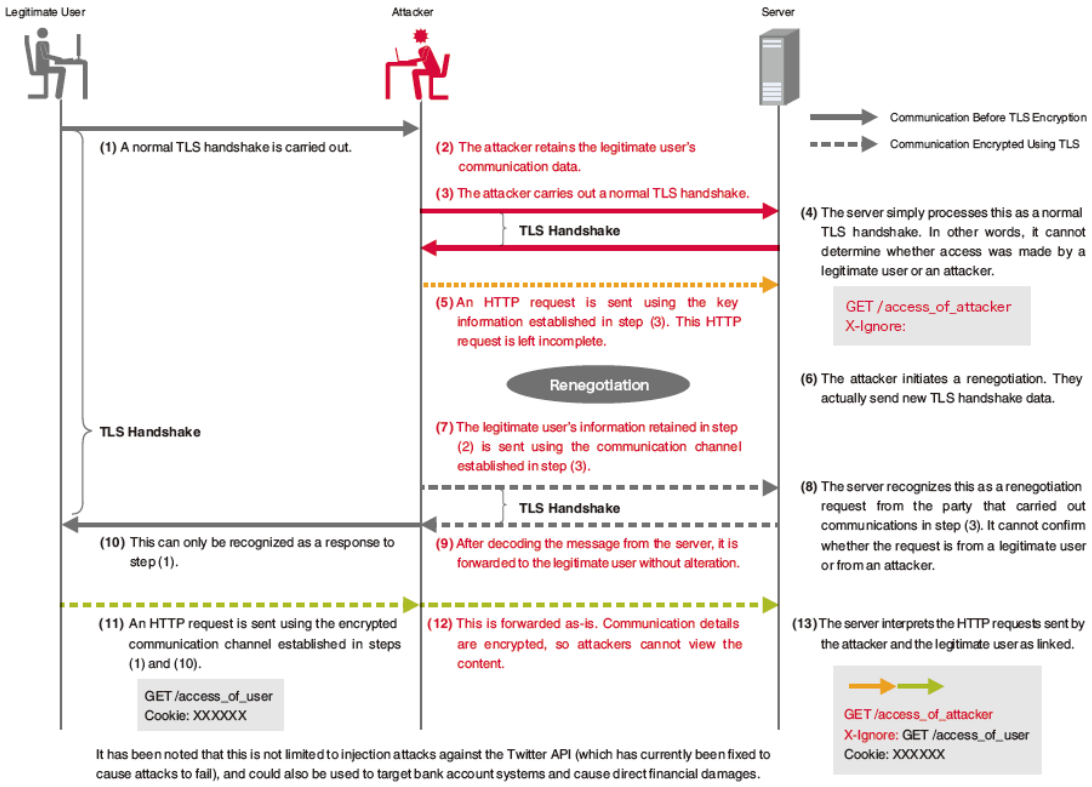

図 8 Renegotiation 攻撃の概要（[12]より引用）

図 8 に示すように、Renegotiation されたデータは暗号化されているため、攻撃者が内容を参照することはできないが、サーバはクライアントからのパケットと攻撃者からのパケットを区別することができない。具体的な攻撃としては、サーバ認証のハンドシェイクをクライアント認証(相互認証)に切り替える例が考えられている。

3.4.2 対策方法：RFC5746

Renegotiation の対策として RFC5746 が提案されている。 RFC5746 では TLS 1.2 で定義されている TLS connection state に対して、secure_renegotiation フラグの追加と、client_verify_data と server_verify_data が追加されている。

追加された内容の詳細は次の通りである。これにより、 Renegotiation を安全に行う実

装がなされていることをサーバとクライアントの間で共有することが可能となる。

① secure_renegotiation フラグ：セキュアな Renegotiation が使用されているかを示す
② client_verify_data：直前のハンドシェイクにおいてクライアントから送信された Finished メッセージ
③ server_verify_data：直前のハンドシェイクにおいてサーバから送信された Finished メッセージ

また、 SSLv3、 TLS 1.0/TLS 1.1 に対して本対策は適用できないため、 RFC5746 では Cipher Suite に TLS_EMPTY_RENEGOTIATION_INFO_SCSV を追加することでハンドシェイクを中断する方法も提案されている。

なお、これらの実装が行われていない Renegotiation が行われた場合に、正しい相手からのリクエストであるかを安全に確認する手段がないため、この実装が行われていない Renegotiation を拒否することが推奨されている。

4. RC4 の脆弱性に基づく攻撃

4.1 RC4 に対する攻撃

本節では、ストリーム暗号 RC4 において現在までに指摘されている脆弱性について示す。

RC4 は、1987 年に Ronald Rivest によって開発されたストリーム暗号である。1 バイトから 256 バイトの鍵から、鍵スケジューリングアルゴリズム（KSA）により 256 バイトの内部状態を作り出し、内部状態からキーストリーム生成アルゴリズムにより 1 バイト単位のキーストリームを出力する。内部の処理はバイト単位であり、鍵は 1 バイトから 256 バイトの可変長（推奨値は 16 バイト）、内部状態は 256 バイトの配列と、2 つのインデックスからなる。

ストリーム暗号の安全性評価としては、以下の 4 つの攻撃を想定する。

1) 鍵回復攻撃:出力されたキーストリームから、ストリーム暗号に対する入力鍵（の一部）を求める攻撃
2) 内部状態復元攻撃：出力されたキーストリームから、内部状態を推定する攻撃
3) 出力予測攻撃：出力されたキーストリームから、将来出力されるキーストリームを予測する攻撃
4) 識別攻撃：出力されたキーストリームと真性乱数を 1/2 以上の無視できない確率で識別する攻撃

これらの攻撃に対し、それぞれ、入力の鍵長を x とした場合、2^x 以下の計算量で推定ができれば攻撃成功となる

鍵回復攻撃においては、入力の鍵長を推奨値である 128 ビットにした場合、FSE2013 において発表された Sepehrdad らの攻撃により無線 LAN の暗号・認証プロトコルである WEP において、19,800 パケットを収集することで鍵回復攻撃が成立することが示されている。また、弱鍵の性質を用いる Weak Key Attack については、長尾らの攻撃 [13] [14] [15] により、$2^{96.36}$ の計算量、$2^{-18.75}$ の確率で鍵回復攻撃が成立することが示されている。

内部状態回復攻撃においては、 CRYPTO2008 における Maximov らの発表により、2^{241} の計算量で内部状態の復元を行うことが示されている。このため、 RC4 においては、鍵長を 241 ビットよりも長くしても安全性は向上しないことが示されている。

出力予測攻撃においては、 EUROCRYPT2005 の Mantin らの攻撃により、2^{45} バイトのキーストリームから、85%の確率で 1 ビットの出力を予測できることが示されている。

識別攻撃においては、同じく EUROCRYPT2005 の Mantin らの攻撃により、$2^{26.5}$ バイトのキーストリームを用いることで、真性乱数との識別ができることが示されている。

また、複数の鍵を用いた場合、 FSE2001 の Mantin らの攻撃により、2^8 バイトのキーストリームを用いることで真性乱数との識別ができることが示されている。このような攻撃は、4.2 に示すように攻撃環境が整った場合には、RC4 に対する攻撃を適用することに

より SSL/TLS のメッセージに対する平文回復攻撃が可能となることが示されている。

4.2 RC4 の攻撃を SSL/TLS に適用した場合の攻撃事例

4.1. に記載のとおり RC4 のアルゴリズム自体の脆弱性は多く示されている。SSL/TLS の中で RC4 を選択した場合、その脆弱性を利用した攻撃が示されている。RC4 の攻撃が適用できる条件として、同じデータに対して異なる鍵を用いて生成された暗号文を複数入手できるような環境下を想定している。そのような環境は比較的容易に得られることが出来る。一例として、図 9 に示すような Broadcast Setting と呼ばれる環境が相当する。Broadcast Setting は、複数のユーザが同じファイルを取得する場合や同じファイル(=平文) を繰り返し送信するような場合に得られる環境である。例えば、ネットワーク利用者の認証やグループ利用の Web ページへのログインなどのように、https の中で basic 認証を行うケースなどで Broadcast Setting の環境は整えることができてしまう。また、OS イメージの配布などの場合でも、Broadcast Setting の環境は準備可能である。その他、図 10 に示す Multi-Session Setting と呼ばれる SSL/TLS で通信を行う際に異なるセッションで同じデータを同じポジションで送信する場合(攻撃対象となるデータ以外の平文は毎回任意のデータで構わない)なども RC4 の攻撃が適用できる条件を満たす。この場合、攻撃対象となるのは、例えば cookie やパスワードといった情報になる。このように RC4 の攻撃が適用できる条件は特殊な利用環境というわけではなく、一般的に存在しうる環境であるといえる。

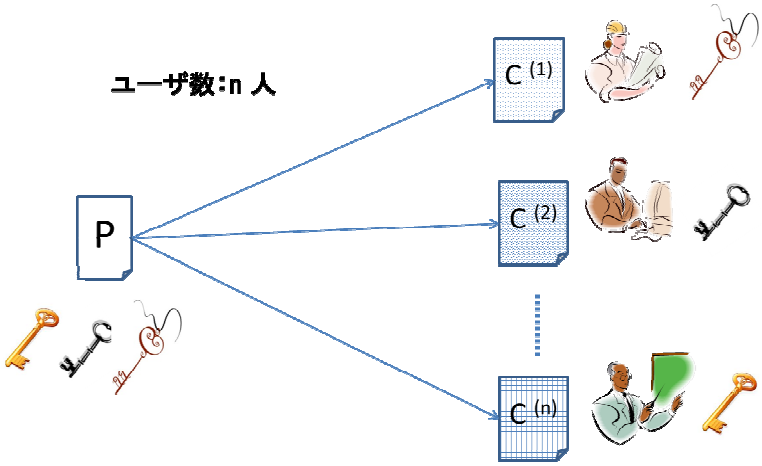

図 9 Broadcast Setting

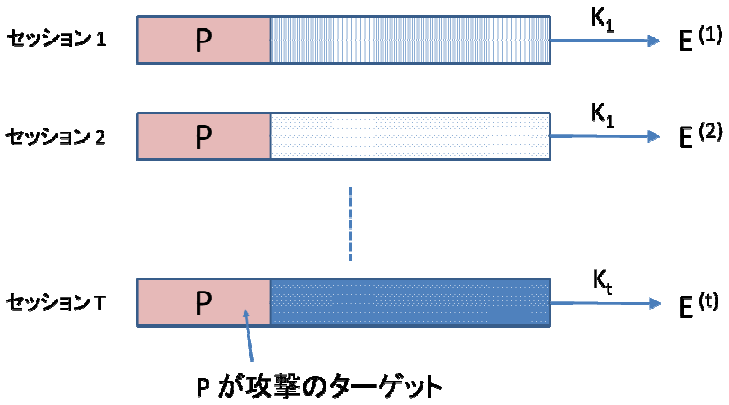

図 10 Multi-Session Setting

近年の結果として [1] [16] では、RC4 の解析を行いキーストリームにおける新しい bias が発見され、それが解析に有効であることを具体的に示されており、RC4 の攻撃が適用できる環境下で、先頭 1000 T バイトの平文を 2^{34} 個の暗号文から 0.97 以上の確率で復元できてしまうことが示されている。さらに [17]では、平文となり得る候補が絞れる場合は、より効率的に解析が行えることが示されている。例えば，平文が PIN code などの場合、入力に使われる文字の種類は 10 種類に限られる。この場合、平文の先頭 257 バイトを 2^{23} の暗号文からランダムに推定する場合よりも高い確率で平文を回復することができる。SSL/TLS の場合、先頭 36 バイトはセッションごとに変化するため、RC4 の解析により、先頭 257 バイトのうち 221 バイトが復元可能となり、入力の種類が限られる場合、より現実的な脅威となることが示されている。

RC4 の攻撃に関しては、 [18] にまとめられている。近年示された強力な攻撃の結果としては、2^{32} の暗号文が集まれば平文の初期 257 byte の任意 byte を確率 0.5 以上で推定

が可能であることが示されている [1] [16]。この結果を鑑みても、SSL/TLS で RC4 を用いる場合のリスクがより高くなっているといえる。

[1] [16] などで示されている解析は、RC4 のキーストリームの先頭の n バイト（推奨 n = 768、理想的には n = 3072）を捨てることにより回避することができる。しかしこのような対応をした場合であっても、回避できない攻撃があることが [19] により示されている。具体的には平文の一部(連続した 6 バイト程度) が知られてしまっている場合、同じ平文に対して 2^{34} の暗号文が集められてしまうと，連続した 1 ペタバイトの平文が 0.6 以上の確率で復元されてしまう．また、平文の情報が一切知られていない場合であっても、同じ平文に対して 2^{35} の暗号文が集められてしまうと，平文のどの位置であっても 1 に近い確率で復元されてしまうことが示されている。

また、 [20] では基本的な攻撃方針としては [1] と同様の手法を用い、成功確率を上げるための最適化を施した解析結果を示している。 具体的には、Broadcast セッティングが実現できるいくつかの具体的な事例を実際に実装し SSL/TLS で RC4 を用いることが現実的な脅威になりうることを示している。事例 1) Java スクリプトの脆弱性を利用し、不正な JavaScript をユーザに使わせることにより、その Java スクリプトを使って大量のターゲットメッセージの Cookie を暗号化して送信させることにより、Broadcast セッティングの環境を実現させる。この不正な Java スクリプトを用いた Broadcast セッティングは、具体的には攻撃者の Web サイトから Java スクリプトマルウェア をダウンロードさせ、その上で https リクエストを大量にリモートサーバに送らせることにより実現できる。事例 2) IMAP(Internet Message Access Protocol；メールサーバ上の電子メールにアクセスし操作するためのプロトコル) で送られるパスワードをターゲットとし、IMAP サーバにアクセスする際に暗号化されたパスワードが送られる仕組みに着目し、暗号化されたパスワードが送られた後に TCP コネクションをリセットし、暗号化されたパスワードを繰り返し送らせることにより Broadcast セッティングを実現させている。具体的に示されている結果として、先頭 256 バイトの bias を実験的に調べ、同じ平文に対して 2^{26} の暗号文を集められると毎回変化する 36 バイトを除いた 40 バイト が 0.5 以上の確率で復元されてしまうことが示されている。 また、同じ平文に対して 2^{32} の暗号文を集められると毎回変化する 36 バイトを除いた 220 バイト が 0.96 以上の確率で復元されてしまうことが示されている。また、ターゲットとなる平文の直前の平文が知られている場合、そのターゲットとなっている平文について、$16 \cdot 2^{30}$ の暗号文を集められるとおおよそ 1 の確率で復元されてしまうことが示されている。

このように、RC4 の攻撃が適用できる条件が整う環境下では、RC4 のアルゴリズムの攻撃は現実的に実現し得るものであり、攻撃者は暗号文を集めれば攻撃を試みることができてしまう。3 章に示された数々の攻撃に対してはそれらを防止する対処策を施すことができる一方、 RC4 の攻撃は対処策がないため、SSL/TLS を運用する選択肢として、RC4 を用いることは、現実的な脅威を招く原因となり得る。

謝辞

本ガイドラインの執筆にあたり、国立大学法人広島大学 大東俊博助教、株式会社富士通研究所 伊豆哲也様、株式会社インターネットイニシアティブ 須賀祐治様、ソニー株式会社 五十部孝典様より、SSL/TLS 及び RC4 に対する攻撃および安全性に関する知見のご提供とご助言をいただきました。ここに深く感謝申し上げます。

引用文献

[1] T. Isobe, T. Ohigashi, Y. Watanabe , M. Morii, “Full Plaintext Recovery Attack on Broadcast RC4,” FSE, 2013.

[2] J. Rizzo and T. Duong, BEAST: Surprising Crypto Attack against HTTPS, http://www.ekoparty.org/eng/2011/thai-duong.php.: ekoparty, 2011.

[3] “Bug 665814,”: https://bugzilla.mozilla.org/show_bug.cgi?id=665814#c59.

[4] 黒川 貴司, 野島 良, 盛合 志帆, “TLS1.0 における CBC モードの安全性について,” 第 31 回暗号と情報セキュリティシンポジウム (SCIS2014), 2014.

[5] “Transport Layer Security (TLS) Extensions: Extension Definitions.,”: http://tools.ietf.org/ html/rfc6066..

[6] J. Rizzo and T. Duong, “The CRIME Attack,” ekopary, 2012.

[7] T. Be'ery and A. Shulman, “A Perfec Crime? Only TIME Will Tell,” BlackHat, 2013.

[8] Y. Glick, N. Harris and A. Prado, “BREACH: REVIVING THE CRIME ATTACK,” BlackHat, 2013.

[9] AlFardan and Paterson, “Lucky Thirteen: Breaking the TLS and DTLS, IEEE Security&Privacy,” 2013.

[10] “JVNVU#120541:SSL および TLS プロトコルに脆弱性,” 11 2009.: http://jvn.jp/cert/JVNVU120541/.

[11] S. Joe , R. Eric, “TLS Renegotiation Vulnerability,: http://tools.ietf.org/agenda/76/slides/tls-7.pdf.

[12] Internet Initiative Japan, "1.4.2 MITM Attacks Using a Vulnerability in the SSL and TLS Renegotiation", Internet Infrastructure Review vol.6, 2010.

[13] A. Nagao, T. Ohigashi, T. Isobe and M. Morii, “New Classes of Weak Keys on RC4 using Predictive State,” Computer Security Symposium 2012 (CSS2012), 2012.

[14] A. Nagao, T. Ohigashi, T. Isobe and M. Morii, “Expanding Weak-Key Space of RC4,” 2013 年暗号と情報セキュリティシンポジウム(SCIS2013), 2013.

[15] A. Nagao, T. Ohigashi, T. Isobe and M. Morii, "Expanding Weak-Key Space of RC4," Journal of Information Processing, vol.22, no.2, 2014.

[16] T. Isobe, T. Ohigashi, Y. Watanabe and M. Morii, “Comprehensive Analysis of Initial Keystream Biases of RC4,” IEICE Trans. on Fundamentals of Electronics, Comm. and Computer Sciences, 2014.

[17] Y. Watanabe, T. Isobe, T. Ohigashi, M. Morii, “Vulnerability of RC4 in SSL/TLS,”

情報通信システムセキュリティ(ICSS)研究会, 2013.

[18] CRYPTREC, “「CRYPTREC Report 2012 暗号方式委員会報告書」,” 2012.

[19] T. Ohigashi, T. Isobe, Y. Watanabe , M. Morii, “How to Recover Any Byte of Plaintext on RC4,” SAC, 2013.

[20] N. AlFardan, D. J. Bernstein, K. G. Paterson , J. C. Schuldt, “On the Security of RC4 in TLS,” USENIX, 2013.

[21] “CVE Details,”: http://www.cvedetails.com/cve/CVE-2011-3389.

[22] “Mozilla Firefox,”: https://developer.mozilla.org/en-US/docs/Security_in_Firefox_2.

[23] “Google Chrome,”: https://code.google.com/p/chromium/issues/detail?id=90392.

[24] “Microsoft Security Bulletin - MS2-006 - Important,”: http://technet.microsoft.com/en-us/security/bulletin/ms12-006.

[25] “Oracle,”: http://www.oracle.com/technetwork/topics/security/javacpuoct2011-443431.html.

[26] “List of browsers support for different TLS version,”: https://en.wikipedia.org/wiki/Transport_Layer_Security#Web_browsers.

発行日　2014年 7月14日　第1版
発行者
・〒184-8795
東京都小金井市貫井北町四丁目2番1号
独立行政法人 情報通信研究機構
（ネットワークセキュリティ研究所 セキュリティ基盤研究室、
セキュリティアーキテクチャ研究室）
NATIONAL INSTITUTE OF
INFORMATION AND COMMUNICATIONS TECHNOLOGY
4-2-1 NUKUI-KITAMACHI,KOGANEI
TOKYO,184-8795 JAPAN

・〒113-6591
東京都文京区本駒込二丁目28番8号
独立行政法人 情報処理推進機構
（セキュリティセンター 暗号グループ）
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
2-28-8 HONKOMAGOME,BUNKYO-KU
TOKYO,113-6591 JAPAN