WHITE PAPER:

White Paper

SSL を理解するための基礎ネゴシエーション

暗号化通信がはじまるまで

Symantec

Norton SECURED
powered by Symantec

CONTENTS

SSL の概要

ネゴシエーションのしくみ

個人情報を送信するフォームや、ID、パスワードなどを入力するフォームで、入力データがどのような仕組みで暗号化されるのか、気になったことはありませんか?

https から始まるアドレスであれば、ブラウザに南京錠のアイコンが表示され、通信データは SSL によって暗号化されます。大切なデータを目的の相手に安全に届けるために暗号化は重要ですが、SSL が通信を始める段階では、まだ暗号化を開始できません。平文で通信を開始しながら通信対象の存在を確かめ、暗号の共通鍵を共有するに至るまでの一連の工程を経て、はじめて暗号化通信を開始します。この一連の工程を SSL ネゴシエーションと言います。

本ホワイトペーパーでは、はじめに SSL の概要を説明してから、SSL の醍醐味と言っても過言ではない「ネゴシエーション」の段階で交わされる通信の中で、通信の安全性を確立していく様子を解説いたします。

SSL の概要

SSL が必要とされる背景

なぜ SSL が広く普及し、インターネットの安全な取引に欠かせないツールとなっているのでしょうか。言うまでもなく、ネットショッピング、ネットバンキング、ネット予約など、インターネットを利用した取引は広く行われています。

こうした取引は、一般にインターネットのウェブサイトを通じて行われますが、クレジットカード番号が盗聴されたり、振込金額が改ざんされたり、あるいは「なりすましサイト」によって情報を盗み取ろうとするフィッシング詐欺の危険性がある状況では、安心して利用できません。

個人に限らず企業や政府機関など、インターネットを利用するすべての人にとって、通信の安全性を確保する必要性がますます高まっています。

ウェブを安全に使うための SSL

ウェブを安全に使うための技術として 1990 年代中頃に発表されたのが SSL（Secure Sockets Layer）です。当時 SSL を開発したネットスケープコミュニケーションズ社（Netscape Communications Corporation）は、改良を続けて 1995 年には SSL 3.0 を公開しました。

1999 年 に は IETF（The Internet Engineering Task Force）というインターネットで利用される技術を標準化する団体が SSL の技術仕様として RFC2246 を発行し、SSL と同じ仕組みを使うオープンな標準規格として TLS を定めました。TLS は 2008 年発行の RFC5246 で、Version 1.2 となっています。

現在では SSL/TLS がセキュアな通信のデファクトスタンダードとなっており、SSL と TLS は厳密には異なりますが、基本的には同じ仕組みで動いています。

安全を確保するしくみ

インターネット通信の主なリスクには、「盗聴」、「改ざん」、「なりすまし」が挙げられます。これらの被害を防ぐためには、次のような方法を使います。

「盗聴」対策

インターネットでは、いくつものサーバを経由してデータを転送するため、第三者が通信を傍受することが比較的簡単です。盗聴による被害を防ぐには、万が一傍受されてもデータを解読できないように「暗号化」して送る事で、情報が漏れないようにすることです。その際、暗号は簡単に解かれてしまわないように、強力な暗号方式を使うことが重要です。近年では無線 LAN を使ったネット環境が充実して、物理的につながっていない状態でも盗聴できる事があるので注意が必要です。

「改ざん」対策

通信経路の途中で、データを書き換えられてしまう場合の対策には、「ハッシュ関数」(別名 : ダイジェスト関数) を応用します。

ハッシュ関数とは、大きなデータを、一意の短いデータに要約する関数です。

ハッシュ関数から出力された結果を「ハッシュ値」と言います。このハッシュ値は、入力データの少しの違いで結果が大きく異なるという性質と、「出力結果から元のデータを逆計算できない」という性質を併せて持っています。

ハッシュ関数とハッシュ値の性質を応用すると、データが改ざんされていないかどうかを検出できるようになります。例えば、電子メールを送信した時のハッシュ値が、相手が受信した時に変わっているような場合には、受信されるまでの途中で第三者が内容を書き換えた、ということが分かります。ハッシュ関数のアルゴリズムである MD2、MD5 は既に強度が低下しており、現在は SHA-1 アルゴリズムが主流です。近い将来は更に安全なアルゴリズムへの移行が必要になると議論されています。

「なりすまし」対策

通信相手が本物かどうかを確認するためには、「SSL サーバ証明書」に含まれている情報を使って相手を確認します。SSL サーバ証明書は、前述のハッシュ関数や後述の公開鍵暗号方式を用いてその正当性を検証する仕組みを備えています。SSL サーバ証明書が「信頼」されるためには、SSL サーバ証明書の発行元にあたるルート認証局証明書と、それを運用する認証局、さらには認証局が SSL サーバ証明書を発行するための審査プロセスが信頼できる必要があります。

暗号の種類

暗号方式には共通鍵暗号方式と公開鍵暗号方式がありますが、SSL では両方の暗号方式を使ってデータの暗号化処理を行います。

共通鍵暗号方式

送信者と受信者が同じ鍵を使って暗号化通信をします。暗号鍵を送信者と受信者だけの秘密として共有することから「共有鍵暗号」とも呼ばれます。

共通鍵暗号方式は、公開鍵暗号方式に比べると暗号化と復号の処理が高速ですが、通信を始める前に共通鍵を第三者に知られることなく共有しておかなければなりません。インターネットで初めて通信する相手とは共通鍵を共有していないので、いきなり暗号化通信を開始することはできませんから、使い方に工夫が必要です。

公開鍵暗号方式

公開鍵と秘密鍵と呼ぶ 2 つの異なる鍵を使って、暗号化通信をします。一方の鍵で暗号化したデータを復号するには、暗号化に用いた鍵と対になる他方の鍵を使う必要があります。このような特徴から「非対称暗号」とも呼ばれています。

公開鍵暗号方式では、公開鍵を第三者に知られても、秘密鍵が秘密になっている限り、暗号化されたデータの内容を第三者は知り得ないので、一方の鍵（公開鍵）を公開しながら暗号化通信ができるという点で、共通鍵暗号にはない利便性があります。

ネゴシエーションのしくみ

2 段階になっている SSL の通信

SSL の通信では、初めての相手でも通信できるように、最初は暗号を使わずに通信を開始し、公開鍵暗号を使って相手を確認しながら共通鍵を双方で共有する状態に移り、最終的に共通鍵暗号方式によって安全にデータを送受信するという工程を辿ります。

通信開始から共通鍵を双方が共有するまでの段階を、ネゴシエーションと呼びます。

そして、共有した共通鍵を使って、目的のデータを暗号化して通信する段階があります。

以下では、ネゴシエーションの中でデータ通信に使う共通鍵を、第三者に悟られないように共有するまでの工程を説明します。

ネゴシエーションの目的

認証した相手とデータを暗号で通信する前の段階で、「認証」と「アルゴリズムの決定」を目的とする「ネゴシエーション」が行われます。

認証

通信を始めるにあたって、先ず通信する相手が本物であるかどうかを確認する必要があります。相手が偽物であったのでは、どんなに優れた暗号を使って通信しても情報が安全にやり取りされているとは言えません。そのために、証明書を交換して、通信する相手を確認（認証）します。

アルゴリズム

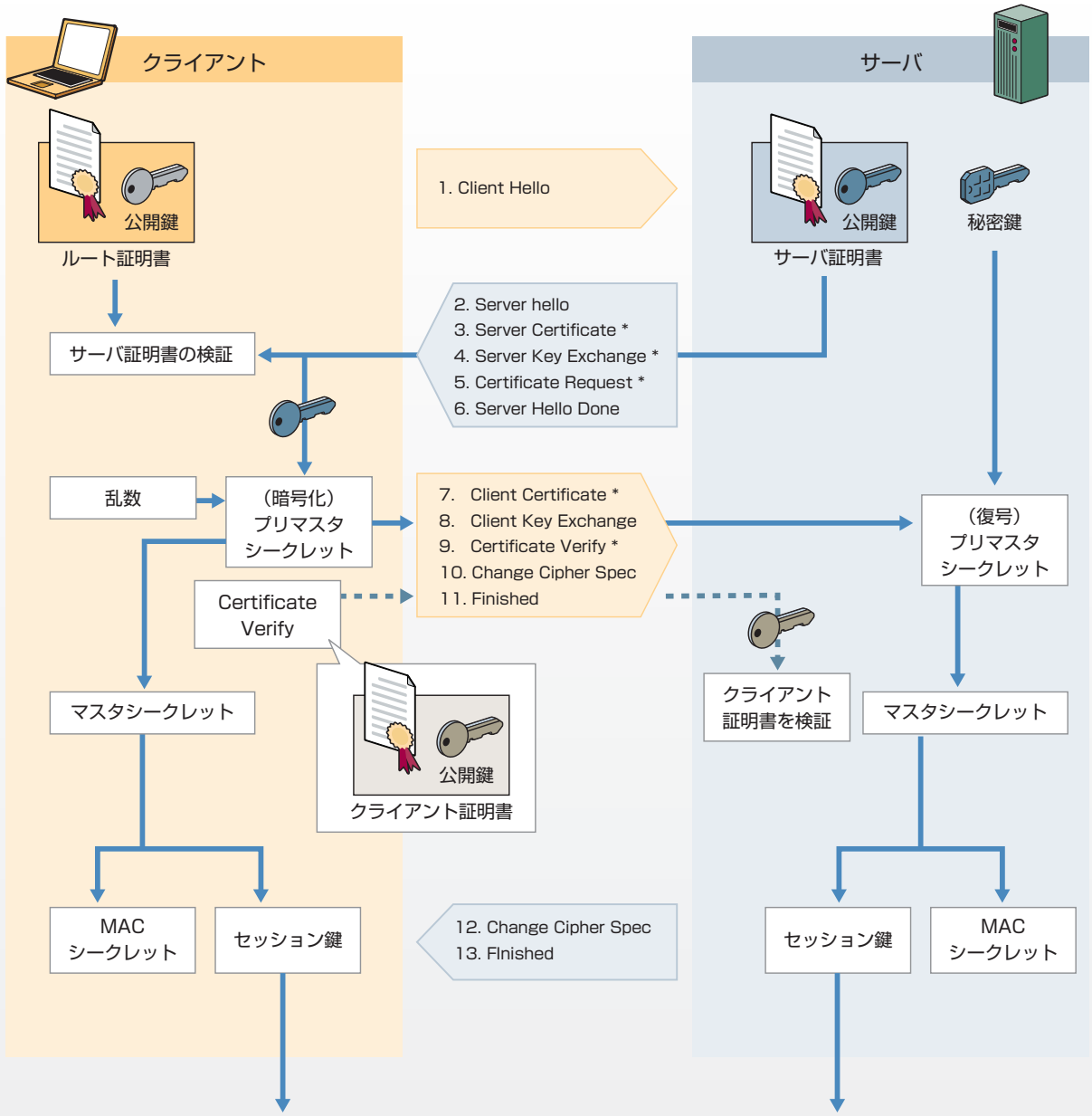
認証によって相手を確認できたら、暗号やハッシュ関数を処理する手順（アルゴリズム）を決定します。利用する暗号強度やハッシュ関数は、サーバとクライアント（ブラウザなど）の組み合わせによって主に決定されます。

ネゴシエーションの手順

ネゴシエーションの間にも、交わされる通信内容が第三者に読み取られてしまっては安全を保てません。

次の図は、ネゴシエーション中のクライアントとサーバの動作の概要を表しています。図の中央に番号順に並んでいるのは、クライアントとサーバの間で交わされるメッセージです。「*」が付いている メッセージは、状況に応じて省略できる場合があります。

ネゴシエーションの動作を、メッセージの順を追って説明します。



以上で SSL ネゴシエーションが終了となります。以下、共通鍵（セッション鍵）を使ってアプリケーションが必要とするデータの暗号化、復号化が繰り返されます。

1.Client Hello

クライアントが、通信の開始をサーバに通知します。このメッセージの中には、次の内容が含まれています。

- プロトコルのバージョン(SSL/TLSのバージョン)
- 乱数(後で共通鍵の算出に使用)
- セッションID(直前にアクセスしていれば、セッションを再開してネゴシエーションを省略可能)
- クライアントが利用できる暗号化方式やデータの圧縮方法の一覧

プロトコルにある SSL2.0 は脆弱性が指摘されていることから、安全性を確保するためには、SSL3.0 以上のバージョンを使うことを推奨します。また、一旦は SSL3.0 で繋がっても、サーバ側に脆弱性があると第三者の干渉で SSL2.0 に切り替えられてしまう危険があるため、クライアント側においても SSL2.0 の設定を無効にしておくことが推奨されています。

また、暗号アルゴリズムについても、共通鍵暗号方式のうち、鍵長が 40bit や 56bit の暗号は短くて解読されやすいので、安全なアルゴリズムのみを利用可能としておくことを推奨します。

2.Server Hello

サーバは Client Hello を受けて、これから使う暗号とハッシュ関数のアルゴリズムを決定し、クライアントに通知します。アルゴリズムは、クライアントから送信された一覧の中から選択します。このメッセージの中には、次の内容が含まれています。

- プロトコルのバージョン(SSL/TLSのバージョン)
- 乱数(後で共通鍵の算出に使用)
- セッションID(再接続してネゴシエーションを省略する場合に使用)
- 決定した暗号化方式やデータの圧縮方式

通信の安全性を確保するためには、暗号強度の高いアルゴリズムを使う必要があります。Client Hello の説明でも述べた通り、プロトコルやアルゴリズムについては、サーバ側でもなるべく安全性の高い方式を選択する必要があります。

3.Server Certificate(省略されることがあります)

サーバはクライアントに向けて、SSL サーバ証明書を送信します。このメッセージには、その証明書を発行した認証局の証明書や、さらに上位の認証局があればその証明書も含むように、ルート証明書までの証明書のリスト（証明書チェーン）を含んでいます。

4.Server Key Exchange(省略されることがあります)

SSL サーバ証明書を持っていないサーバの場合、または Server Certificate で送信した SSL サーバ証明書に公開鍵が含まれない場合に、共通鍵を交換するための公開鍵を送信するメッセージです。

サーバは一時的な公開鍵を生成し、サーバの署名と共に送信します。

5.Certificate Request(省略されることがあります)

クライアントを認証する必要がある場合に、サーバがクライアントに対してクライアント認証用の証明書を送るように要求するメッセージです。

このメッセージは、サーバが信頼するルート証明書のリストを含んでいます。

6.Server Hello Done

クライアントに、Server Hello から始まる一連のメッセージが完了したことを通知します。

クライアントはこれを受けて、再びメッセージを送信する側になります。

7.Client Certificate（省略されることがあります）

サーバから Certificate Request があった場合に、クライアント証明書をサーバに送ります。前述の SSL サーバ証明書と同様に、ルート証明書までの証明書のリスト（証明書チェーン）も送信します。

サーバから Certificate Request が無かった場合には省略します。

8.Client Key Exchange

ここまでのやりとりで、クライアントは Server Certificate 中の SSL サーバ証明書に含まれている公開鍵を得ています。しかし、これらの公開鍵は、第三者にも傍受可能な状態で通信されていたので、別の第三者がクライアントになりすませてしまうという危険性が残っています。

そこで、クライアントはサーバとクライアントだけが知り得る共通鍵を作り出すために、「プリマスタシークレット」と呼ばれる乱数情報を生成します。

生成したプリマスタシークレットを、サーバの公開鍵を使って暗号化してサーバに送信します。この暗号化によってプリマスタシークレットは、サーバだけが持っている秘密鍵でしか解読できない状態になります。このようにクライアントとサーバだけが共有する状態を作り出します。

このプリマスタシークレットを共有する工程で、公開鍵暗号方式が用いられています。

9.Certificate Verify（省略されることがあります）

Client Certificate を送信した場合に送る、クライアント証明書に対する署名データです。

クライアントは署名 (Certificate Verify) を生成し、サーバに送信します。Certificate Verify を受け取ったサーバは、クライアントから受け取った証明書 (Client Certificate) を使って署名を検証します。

サーバから Certificate Request が無かった場合には、Client Certificate と共に省略します。

10.Change Cipher Spec

ここまでの通信で、クライアントとサーバだけが、プリマスタシークレットを共有した事になります。

そこで、クライアントとサーバはそれぞれ、Client Hello と Server Hello に含まれていた乱数とプリマスタシークレットを使って「マスタシークレット」を生成します。 クライアントとサーバは同じ方法でマスタシークレットを生成するので、できあがったマスタシークレットは、再度交換する必要はありません。

さらに、マスタシークレットから、以降の暗号化通信に用いるための共通鍵（セッション鍵とも呼ばれます）を生成します。

ここまでで、以後の暗号化通信に必要な準備は揃いました。これ以後は、この暗号で通信することを通知するために、クライアントはサーバに対して Change Cipher Spec を送信します。

11.Finished

クライアントがサーバ認証に成功し、共通鍵を共有できたことをサーバに通知します。ここまでで得られた共通鍵で、メッセージを暗号化して送信します。

12.Change Cipher Spec
サーバもまた、受け取ったプリマスタシークレットと乱数を元にして、マスタシークレットを生成し、さらに共通鍵を生成します。この準備が整ったことをクライアントに通知するために、サーバはクライアントに対して Change Cipher Spec を送信します。
13.Finished
サーバがクライアント認証に成功し、共通鍵を共有できたことをクライアントに通知します。ここまでで得られた共通鍵で、メッセージを暗号化して送信します。

「11. Finished」と「13. Finished」を互いに解読できたならば、両者は間違いなく共通鍵を共有できていることになります。

これで、SSL ネゴシエーションが完了しました。以後は、アプリケーションが必要とするデータ通信を、暗号化して送受信する段階に移ります。

まとめ

SSL ネゴシエーションでは、通信相手が本物かどうかを認証し、後に続くデータ通信の安全を確保するためのプロトコルを決定します。

証明書を使って相手を確かめる場合、証明書があるというだけでは不十分です。証明書が信頼されるためには、認証局が発行する時にどのような審査をして、何を証明しているのかが重要です。第三者である認証局の発行基準がどのようなものであるのか確認しておくと安心です。

また、SSL が実現する通信の暗号強度は、ネゴシエーションの段階で決定するプロトコルや暗号アルゴリズムに依存しています。

SSL が発表されてから既に 20 年近くが経過し、脆弱性を克服するために改良が重ねられてきていますが、プロトコルや暗号アルゴリズムを適切に選択しなければ、本来の安全性を発揮することはできません。

暗号強度が強い鍵は計算負荷が大きいという考え方から、処理速度を上げるためのトレードオフとして暗号強度の弱い鍵を選択することは、結果として危険性を高めることになってしまい、SSL を使う意味を毀損させてしまいます。

SSL 全体のレスポンスを改善するためには、SSL アクセラレータを導入したり、ネゴシエーションの回数を管理するためにサーバの Keep-alive 設定を見直すことも有効です。

SSL の安全性を保つためには、ネゴシエーションの工程で選択されるプロトコルやアルゴリズムに注意を払い、常にサーバの設定が適切になるように考慮する事が重要です。

合同会社シマンテック・ウェブサイトセキュリティ
https://www.jp.websecurity.symantec.com/
〒104-0028　東京都港区赤坂1-11-44赤坂インターシティ
Tel : 0120-707-637
E-mail : websales_jp@symantec.com