数理モデル化と問題解決　23－5
（1999．2．19）

SSLプロトコルの形式仕様記述と検証

田中 俊行[1,2]　Chris George[3]　張 漢明[2,4]　荒木 啓二郎[1,2]
[1]九州大学大学院システム情報科学研究科
[2]九州システム情報技術研究所
[3]国連大学国際ソフトウェア技術研究所
[4]奈良先端科学技術大学院大学

通信の安全性が保証されていないインターネットのようなネットワーク上で，安全な通信を実現するために「セキュリティプロトコル」が利用されている．セキュリティプロトコルが安全な通信を実現していることを示すには，プロトコルがどのような前提条件を想定していて，何を保証しているのかを明確にすることが重要である．本研究の目的は，セキュリティプロトコルの安全性を明確にし，形式的手法によるセキュリティプロトコルの仕様記述法および検証法を確立することである．本研究では，形式手法RAISEを用いてセキュリティプロトコルの仕様記述を行い，プロトコルが保証している安全性を明確にし，安全性に対する形式的な証明を与えることで安全性の保証を図る．本稿では，SSLプロトコルを対象として，RSLによるプロトコルの仕様記述，およびSSLプロトコルが想定しているサーバへのなりすまし攻撃に対する安全性の検証法を示す．セキュリティプロトコルの仕様を形式的に厳密に定義することにより，プロトコルが想定している前提条件が明示的に記述され，プロトコルが保証している安全性が明確になる．

Formal Specification and Verification of the SSL Protocol

Toshiyuki Tanaka[1,2], Chris George[3], Han-Myung Chang[2,4],
and Keijiro Araki[1,2]
[1]Graduate School of Information Science and Electrical Engineering, Kyushu University
[2]Institute of Systems & Information Technologies/KYUSHU
[3]International Institute for Software Technology of United Nations University
[4]Nara Institute of Science and Technology

Security protocols are used to establish secure connections between computers over an insecure network such as the Internet. One of the most important issues is to make clear what kind of precondition a security protocol assumes and what kind of property is guaranteed by the protocol. Our goal is to make security properties clear and to present formal methods for specification and verification of security protocols. We intend to describe formal specification of a security protocol using the RAISE method, and to make security properties clear which the protocol intends to guarantee. We also aim to give formal proof of the properties and guarantee that the protocol has them. In this report, we specify the SSL Protocol using RSL and verify its security against the man-in-the-middle attack which is supposed to be done to the protocol. As we describe formal and rigorous specification of the protocol, we can obtain clear preconditions which the protocol assumes, and make clear what kind of property the protocol guarantees.

1　はじめに

インターネットは通信の安全性を保証していないため，通信している情報が第三者によって覗き見され，改ざんされるといった問題がある．「セキュリティプロトコル」は暗号技術を利用して，インターネットのようなネットワーク上で安全な通信路を確立するためのプロトコルである．セキュリティプロトコルを開発する際は，通信の安全性が確保されていることを検証する必要がある．

通信の安全性に関しては，暗号の強さ，秘密鍵の保管方法，あるいはプロトコル実装時のソフトウェアのバグなどが要因として考えられる．セキュリティプロトコルの安全性を議論するためには，プロトコルがどのような前提条件の元で，何を保証しているのかを明確に記述しなければならない．本研究の目的は，セキュリティプロトコルの安全性を明確にし，セキュリティプロトコルの形式的な仕様記述や検証の方法を確立することである．

本研究では形式手法としてRAISE (Rigorous Approach to Industrial Software Engineering) を導入し，具体的な例題におけるプロトコルの仕様記

述や検証を通して，安全性の明確化を図る．RAISEはRSL (RAISE Specification Language) という仕様記述言語や，RAISE Development Methodという段階的な開発手法，および計算機による支援ツールから構成されている．

本研究ではインターネット上のセキュリティプロトコルの一つであるSSLプロトコルバージョン2（Secure Sockets Layer Protocol version 2）の形式仕様をRSLで記述した．また，SSLプロトコルが想定しているネットワーク攻撃の一つであるサーバへのなりすまし攻撃をRSLで記述し，その攻撃が防がれていることを検証した．

プロトコルの仕様を形式的に記述することで，

- 想定されている前提条件の明示
- プロトコルが保証している安全性の明確化

といった効果が得られる．自然言語による仕様の曖昧な前提条件を明示することにより，プロトコルが本質的に保証している安全性とは何かを明確にする．本稿では，SSLプロトコルの形式仕様において必要な性質を示し，プロトコルを形式的に記述するための指針を与える．

本稿では2章で形式手法RAISEについて，3章でSSLプロトコルについて概説する．4章で通信をモデル化し，RSLによるSSLプロトコルの形式仕様を与える．5章ではSSLプロトコルが保証している性質を明確にし，その性質についての検証を行う．6章で本研究に関する考察を行い，7章で関連研究について紹介する．最後に，8章で本研究のまとめを行い，今後の展開について述べる．

2 形式手法RAISE

形式手法RAISEは次の3つから構成されている．

- 仕様記述言語: RSL
- 開発手法: RAISE Development Method
- 支援ツール: RAISEツール

RSLは仕様記述能力の高い言語である [8]．例えば，抽象型を用いて関数をシグネチャと公理で定義する，抽象度の高い適用型（applicative）スタイル，あるいは，具体的な型を利用した変数への代入によってシステムの状態を変化させる関数（手続き）を定義する，抽象度の低い手続き型（imperative）スタイルといった様々なスタイルで仕様を記述することが可能である．

表1: SSL-HPのメッセージ

	方向	メッセージ名	内容
1.	C → S	CLIENT-HELLO	s_id, chall
2.	S → C	SERVER-HELLO	s_id_hit, s_cert, c_id
3.	C → S	CLIENT-MASTER-KEY	k_{sp}\{m_key\}
4.	C → S	CLIENT-FINISHED	k_{cw}\{c_id\}
5.	S → C	SERVER-VERIFY	k_{sw}\{chall\}
6.	S → C	SERVER-FINISHED	k_{sw}\{s_id\}

RAISE Development Methodは，抽象度の高い仕様を詳細化し，実装に近い仕様を段階的に生成するための開発手法である [9]．各開発段階の仕様の間には満足すべき関係があり，正しい開発を行っているかどうかの検証手段を示している．

RAISEでは，RAISE Development Methodに沿ったシステム開発を支援するために，計算機を利用したRAISEツールが用意されている．RAISEツールは，RSLを用いた仕様記述や検証，そして最終的な仕様をプログラミング言語へと変換することを支援する．

3 SSLプロトコル

SSLプロトコルは，インターネット上のセキュリティプロトコルの1つである [3]．SSLプロトコルはTCPのような信頼できるトランスポートプロトコルを必要とするが，アプリケーションプロトコルには依存しないので，SSLプロトコルを利用するHTTPなどのサーバやクライアントが開発されている．

SSLプロトコルはSSLハンドシェイクプロトコル（SSL-HP）とSSLレコードプロトコルという2つのプロトコルから構成されるが，本研究ではSSL-HPに着目する．SSL-HPではクライアントとサーバが表1のようなメッセージをやりとりして，クライアントはサーバを認証し，暗号形式と秘密情報を共有することができる．つまり，SSL-HPが安全な通信路を確立する役割を担っている．

表1でCはクライアントを，Sはサーバを表す．k\{data\}はdataを暗号鍵kで暗号化したものを意味する．暗号鍵として，サーバの公開鍵（k_{sp}）とセッション鍵（k_{cw},k_{sw}）がある．セッション鍵はSSLプロトコルで行う対称鍵暗号のための鍵で，SSL-HPのセッション毎に乱数（chall, c_id, およびm_key）を用いて作成される．s_idは2回目以降のセッションでm_keyを再利用し，プロトコル

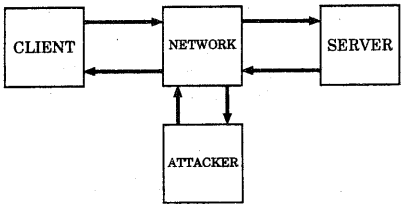

図 1: 通信のモデル

を簡略化するための情報である.

4 RSLによる仕様記述

システムを形式的に記述する際は，対象を適切にモデル化することが必要である．本章ではSSLプロトコルを利用してクライアントとサーバが行う通信をモデル化し，SSL-HPの仕様とサーバへのなりすまし攻撃をRSLで記述する．本稿では抽象度の高い適用型の記述スタイルを採用する．

本稿ではRSLによる仕様記述について，その一部のみを取り上げている．すべてのRSL記述についての詳細は，文献 [7] を参照されたい．

4.1 通信のモデル化

SSLプロトコルはTCPといった信頼できるトランスポートプロトコルを必要とする．また，インターネットのようなネットワークでは，第三者によって通信内容を覗き見されたり，改ざんされたりすることがある．そこで，クライアントとサーバが行う通信を図1のようにモデル化した．

図1のCLIENTとSERVERは，それぞれクライアントとサーバを表す．また，ATTACKERはネットワーク攻撃を試みる攻撃者を表し，NETWORKを介してCLIENTとSERVERの間で行われている通信の邪魔をすることが可能である．

図1のモデルの特徴は通信路を表すNETWORKにあり，その性質を変化させることによって，様々なネットワークの形態を表現することができるようになっている．

4.2 基本的な性質

最初にSSL-HPを利用する上で前提としている基本的な性質をRSLで記述する．以下の記述は公開鍵暗号の性質の一部である．

```
axiom
  [decrypt_key_OK]
    ∀ key, key1, key2 : Key •
      match_key(key1, key2) ⇒
      decrypt_key(encrypt_key(key, key1), key2) =
        ok_key(key)
```

[decrypt_key_OK] により，公開鍵（key1）で暗号化された鍵（key）が，公開鍵に対応した秘密鍵（key2）によって復号化できる性質を定義している．

4.3 クライアントの記述

本節では，SSL-HPを行うクライアントの仕様をRSLで記述する．RSLによる仕様では，クライアントが保持する情報や，各メッセージに対する処理，およびクライアントの振る舞いを定義した．以下はその一部である．

```
scheme
  CLIENT(N : NETBASE(M)) =
    class
      value
        c_next :
          Client × N.Network →
            Client × N.Network
        c_next(c, n) ≡
          case state(c) of
            Client_0 → send_client_hello(c, n),
            Client_1 → recv_server_hello(c, n),
            Client_2 →
              send_client_master_key(c, n),
            Client_3 → send_client_finished(c, n),
            Client_4 → recv_server_verify(c, n),
            Client_5 → recv_server_finished(c, n),
            Client_ok → (c, n),
            Client_error → (c, n)
          end, .....
```

c_next関数はSSL-HPにおけるクライアントの振る舞い — クライアントはSSL-HPを開始する時（Client_0）にCLIENT-HELLOをサーバに送信する（send_client_hello）といったこと — を定義する．

4.4 サーバと攻撃者の記述

サーバの仕様は4.3節と同様，サーバが保持する情報や，各メッセージに対する処理，およびサーバの振る舞いを定義する．その一部を以下に示す．

```
scheme
  SERVER(N : NETBASE(M)) =
    class
      value
        recv_client_hello :
          Server × N.Network →
            Server × N.Network
        recv_client_hello(s, n) ≡
          let (msg, n′) = N.s_get(n) in
            case msg of
              M.raw_msg(M.C_hello(s_id, chall))
              →
                let
                  s′ =
                    change_s_id
                      (s_id, change_chall(chall, s))
                in
```

```
        (change_state(Server_1, s'), n')
      end,
    M.nil → (s, n'),
    _ → send_raw_error(s, n')
  end
end, .....
```

recv_client_hello 関数は CLIENT-HELLO の受信処理を定義する．サーバは CLIENT-HELLO を受信すると，chall と s_id の情報を保持し，CLIENT-HELLO 受信後の状態（Server_1）になる．受信したメッセージが CLIENT-HELLO 以外の場合は，エラーが発生する（send_raw_error）ことを明示している．

サーバになりすます攻撃者は，クライアントがサーバに要求する証明書に関する部分を除いて，サーバと同様に記述することができた．

4.5 通信路の記述

図 1 の NETWORK に対する RSL 記述を変えることで，様々なネットワークの形態を表現することができる．本節では，ネットワーク攻撃を受けない場合，およびサーバへのなりすまし攻撃を受ける場合の通信路の振る舞いを RSL で記述する．

ネットワーク攻撃を受けない通信路

ネットワーク攻撃を受けない場合，図 1 の NETWORK はクライアントの送信した情報をサーバへ届け，同様にサーバの送信した情報をクライアントへ届ける．そのような通信路の振る舞いを RSL で記述した．以下にその一部を示す．

```
scheme
  OK_NET(M : MSGBASE) =
    extend NETBASE(M) with
      class
        axiom
          [cput_sget_ax]
            ∀ msg : M.Message, net : Network •
              s_get(c_put(msg, net)) = (msg, net),
          .....
```

[cput_sget_ax] により，クライアントが通信路（net）に送信した（c_put）情報（msg）は，すべてサーバが受信できる（s_get）ことが分かる．

サーバへのなりすまし攻撃を受ける通信路

攻撃者がサーバへのなりすまし攻撃を試みる場合，NETWORK はクライアントと攻撃者の間で通信させるように振る舞う．そのような通信路の振る舞いを RSL で記述した．以下に示すのはその一部である．

```
scheme
  BAD_NET(M : MSGBASE) =
    extend NETBASE2(M) with
      class
        axiom
          [cput_aget_ax]
            ∀ msg : M.Message, net : Network •
              a_get(c_put(msg, net)) = (msg, net) ,
          .....
```

[cput_aget_ax] により，クライアントが通信路（net）に送信した（c_put）情報（msg）を，攻撃者が受信できる（a_get）ことが分かる．

5 SSL-HP の振る舞いの検証

4 章では SSL-HP とサーバへのなりすまし攻撃を RSL で記述した．適用型のスタイルによって記述された仕様に関する性質は抽象度が高く，その証明が容易になることが多い．

本章では，SSL-HP は攻撃を受けない時に正常終了するという性質，およびサーバへのなりすまし攻撃を防ぐという性質を RSL で記述する．

RAISE ツールを用いて行ったこれらの性質の証明については，文献 [7] を参照されたい．

5.1 SSL-HP が正常終了する性質

SSL-HP が攻撃を受けない時に正常終了するという性質は，次のように表現することができる．

```
theory SYS_OK_TH :
  axiom
    in
      extend
        class object N : OK_NET(M) end
      with
        SYSTEM(N)
    ⊢
      ∀ c : C.Client, s : S.Server, n : N.Network •
        C.state(c) = C.Client_0 ∧
        S.state(s) = S.Server_0 ∧
        N.c_get(n) = (M.nil, n) ∧
        N.s_get(n) = (M.nil, n) ⇒
        let (c', s', n') = run(c, s, n) in
          C.state(c') = C.Client_ok ∧
          S.state(s') = S.Server_ok
        end
end
```

前提として，クライアントとサーバが SSL-HP を開始する時の状態（C.Client_0, S.Server_0）で，通信路には何も送信されていないとする．その前提で SSL-HP を行うと（run(c, s, n)），最終的にクライアントとサーバの状態がそれぞれ C.Client_ok および S.Server_ok になることを意味する．ここで C.Client_ok と S.Server_ok は SSL-HP が正常に終了した時のクライアントやサーバの状態である．

5.2 攻撃を防ぐ性質

サーバへのなりすまし攻撃を防ぐ性質は，次のようにRSLで記述することができる．

```
theory SYS_A1_ERR_TH :
  axiom
    in
      extend
        class object N : BAD_NET(M) end
      with
        SYSTEM_A1(N)
    ⊢
      ∀
        c : C.Client,
        a : A.Attacker,
        s : S.Server,
        n : N.Network
      •
        C.state(c) = C.Client_0 ∧
        A.state(a) = A.Attacker_0 ∧
        S.state(s) = S.Server_0 ∧
        N.c_get(n) = (M.nil, n) ∧
        N.a_get(n) = (M.nil, n) ⇒
        let (c′, a′, s′, n′) = run(c, a, s, n) in
          C.state(c′) = C.Client_error
        end
  end
```

前提として，クライアント，サーバ，および攻撃者がSSL-HPを開始する時の状態（C.Client_0, S.Server_0, およびA.Attacker_0）で，通信路には何も送信されていないとする．その前提の元でSSL-HPを行うと（run(c, a, s, n)），最終的にクライアントの状態がC.Client_errorになるという性質である．C.Client_errorはSSL-HPの途中でプロトコルを止めたクライアントの状態を意味している．クライアントがSSL-HPを最後まで行わないため，攻撃は防がれている．

6 考察

RSLの特徴の一つは，抽象度の高い仕様から，詳細で抽象度の低い仕様まで，一つの言語によって記述できることである．つまり，抽象度の高いRSL記述を利用して，プロトコルの性質についての厳密な証明を行うことも可能であり，また，抽象度の低いRSL記述を利用して，プロトコルの実装を目指すことも可能である．したがって，RAISEのフレームワークを利用することは，実際のプロトコルを開発する際に十分有効であると考えられる．

通信モデル

本稿では，SSLプロトコルを利用するクライアントとサーバの間で行われる通信を図1のようにモデル化した．図1の通信路の性質を変更することで，様々なネットワークの形態に対応することが可能である．図1はSSLプロトコルに限らず，一般的なプロトコルを記述する際の通信のモデルとして利用することが可能であると考えられる．

本稿では，SSL-HPの形式仕様を図1のモデルに沿ったモジュールに分割して記述している．新たなネットワーク攻撃を考慮した仕様を記述する際は，通信路と攻撃者のモジュールを変更するだけでよく，クライアントとサーバの仕様を再利用できる．このように，モデルに沿ってモジュールに分割した仕様を記述することにより，仕様の再利用性を高めることができる．

適用型の記述スタイルの有効性

適用型のスタイルで仕様を記述することにより，抽象度の高いプロトコルの仕様が得られた．例えば，暗号については，アルゴリズムを規定せず，その性質のみを利用してプロトコルに関する検証を行うことができる．この暗号に関する記述には汎用性があり，暗号を利用する新たなプロトコルの仕様を形式的に記述する際に利用することができる．

本研究で得られた仕様に対する性質を記述し，支援ツールを用いて証明を行うことは容易であった．適用型のスタイルで仕様を記述することは，プロトコルの性質について厳密に議論する際に有効であると考えられる．

RSLのscheme, class, およびobjectは，関数型言語MLのファンクタ，シグネチャ，およびストラクチャに似ているので[8]，適用型のRSL仕様を関数型言語MLによって実装することが容易であると予想される．実際にMLで記述したプログラムを動かすことは，プロトコルがどのように振る舞うのかを実感できるので，プロトコルの理解を深める際に有効であると考えられる．

プロトコルが保証する性質

本研究では，SSL-HPに関する2つの性質について，RSLを用いて明確に記述し，検証することができた．セキュリティプロトコルが想定している前提条件と，満たすべき性質を形式的に記述すること，およびその性質について厳密に検証することは，セキュリティプロトコルの開発において重要である．本研究では，セキュリティプロトコルのSSLプロトコルを対象としているが，仕様記述や性質の検証はセキュリティプロトコルに限定

する必要はなく，一般的なプロトコルの仕様や満足すべき性質をRSLを用いて厳密に記述し，検証することも重要である．

7 関連研究

セキュリティプロトコルの解析のために，プロセス代数π-calculus [4, 5] を拡張し，spi calculusが設計された [1]．spi calculusでは通信ポートを隠すことによって通信路の安全性を表現しているため，ネットワーク攻撃を表現することができないと考えられる．

文献 [2] では，セキュリティプロトコルの仕様記述言語と，セキュリティプロトコルに望まれる性質について述べられている．さらに，仕様を記述されたセキュリティプロトコルが，望まれる性質を持っているかどうかを自動的に検証するプロトコル解析器を紹介している．

以上の研究では，抽象度の高い記述によってセキュリティプロトコルの性質を解析するという方針を採用している．抽象度の高い仕様は，その性質についての厳密な証明には有効であるが，実際のプロトコル開発に適用することは困難である．

我々は，既にRSLを用いて，並列性を持った手続き型のスタイルでSSL-HPとサーバへのなりすまし攻撃を記述している [6]．並列性を持った手続き型のスタイルで記述された仕様についての性質は，変数（状態）への代入や並列性のために，証明を行なうことが難しい．そのため，SSL-HPが正常に終了したり，攻撃を防いでいることについては，RSL記述の振る舞いを追って確認しただけである．しかし，並列性を持った手続き型のスタイルで仕様を記述することにより，そのプロトコルに対する理解を深め，既存の仕様書で曖昧になっている点を明確にすることができた．つまり，既存のプロトコルを理解し，その本質を抽出する際に有効であると考えられる．

8 おわりに

本稿では，SSLプロトコルにおけるクライアントとサーバの通信をモデル化し，そのモデルに基づいて，SSL-HPの仕様とSSLプロトコルに対して想定されているサーバへのなりすまし攻撃をRSLで記述した．これらの仕様に対して，ネットワーク攻撃を受けない場合はSSL-HPが正常に終了すること，また，サーバへのなりすまし攻撃を受けた場合はその攻撃を防ぐことについて，RAISEツールを用いて厳密に検証した．適用型のスタイルで書かれた仕様に対して性質を記述し，RAISEツールを用いて証明を行うことは容易であった．セキュリティプロトコルの仕様をRSLで形式的に記述することにより，プロトコルが想定している前提条件が明示的に記述され，プロトコルが保証している安全性が明確になる．

今後は，SSLプロトコルが想定しているサーバへのなりすまし以外のネットワーク攻撃についての形式仕様記述や検証を行い，ネットワーク攻撃に対してセキュリティプロトコルが満足すべき性質を明確にしたい．また，セキュリティプロトコルの安全性として，他にどのような性質が必要なのかを検討することも課題として挙げられる．

参考文献

[1] Abadi, M. and Gordon, A. D.: A Calculus for Cryptographic Protocols: The Spi Calculus, *The 4th ACM Conference on Computer and Communications Security*, ACM Press, pp. 36–47 (1997).

[2] Brackin, S. H.: An Interface Specification Language for Automatically Analyzing Cryptographic Protocols, *ISOC Symposium on Network and Distributed System Security*, IEEE Computer Society Press, pp. 40–51 (1997).

[3] Hickman, K. E. B.: *The SSL Protocol*, Netscape Communications Corp. (1995).

[4] Milner, R., Parrow, J. and Walker, D.: A Calculus of Mobile Processes, Part I, *Information and Computation 100 (1)*, Academic Press, pp. 1–40 (1992).

[5] Milner, R., Parrow, J. and Walker, D.: A Calculus of Mobile Processes, Part II, *Information and Computation 100 (1)*, Academic Press, pp. 41–77 (1992).

[6] Tanaka, T., Chang, H.-M., Taguchi, K. and Araki, K.: Formal Specification and Verification of Security Protocol in RSL, *International Symposium on Future Software Technology '97*, Software Engineers Association, pp. 143–150 (1997).

[7] Tanaka, T. and George, C.: Proving Properties of a Security Protocol Specified in RSL, Technical Report 143, International Institute for Software Technology of the United Nations University (UNU/IIST) (1998). ftp://ftp.iist.unu.edu/pub/techreports/report143.ps.gz.

[8] The RAISE Language Group: *The RAISE Specification Language*, Prentice Hall International (1992).

[9] The RAISE Method Group: *The RAISE Development Method*, Prentice Hall International (1995).