National Institute of
Advanced Industrial Science
and Technology
AIST

AIST-PS-2007-006

ソフトウェア更新システムプロトコルの BAN Logic による安全性検証 (Preliminary Version)

吉田聡　山形頼之

独立行政法人 産業技術総合研究所
システム検証研究センター

算譜科学研究速報
Programming Science
Technical Report

Research Center
for Verification and Semantics

独立行政法人 産業技術総合研究所 システム検証研究センター 発行
Published by Research Center for Verification and Semantics
National Institute of Advanced Industrial Science and Technology

ソフトウエア更新システムプロトコルの BAN logic による安全性検証

吉田聡　　山形頼之

概 要

BAN logic は認証プロトコルの分析を行うために与えらた論理の形式体系である. 本論文では, BAN logic を食品産業向け商品処理装置のために開発中であるソフトウエア更新システムのプロトコルに適用し, その安全性を検証した事例を紹介する. 本事例では 3 つのプロトコルについて考察する. まず最初のプロトコル α についてその仕様から帰結できる性質を示す. 次に, 通信経路上に悪意のある振る舞いを持つ主体が存在しても「商品処理装置は最新版の更新プログラムを受信するか, もしくはエラー通知を受信（送信）する」という性質を満たすような, プロトコル α を変更することによって得られる新たな 2 つのプロトコル β と γ を与える. [1]

1　はじめに

本研究では, 工場内のネットワーク中継装置を介して, 工場内の商品処理装置と管理サーバを接続し, 更新ソフトウェアを管理サーバから商品処理装置へダウンロードするシステムに定理証明技法を適用して検証を行った. その結果, 更新ソフトウェアの作成者が適切であることを, ダウンロードした商品処理装置が確認できることが確かめられた. また, ダウンロードした更新ソフトウェアが常に最新のものであることを確認できるよう, システムの改善案を提案した.

ここで, 定理証明技法による検証は, システムに与えられた条件 · 性質（公理）から, 安全性など常に満たして欲しい性質（定理）を証明することによって検証を行う方法である. つまり, この方法はモデル検査には一般には向かない正しさを目的とした検証に適している. さらに, モデル検査では行えない無限個の状態を持つモデルに対する検証も可能にし, 加えて特定のモデルだけでなく, 条件 · 性質を満たす考え得るすべてのモデルについてその性質が成立つことを帰結できる ([5, 4.4] 参照）. しかし, 不具合の発見についてはモデル検査において行なえるような反例の検出は, 定理証明の技法では一般には行なえない. ゆえに, 定理証明とモデル検査はお互い補い合う関係にある. 本研究では, 特に BAN logic と呼ばれる論理に基づいた定理証明による検証を行った. BAN logic は認証プロトコルの分析を行うために [1] において与えらた論理の形式体系であり, 広く用いられている [3, Section 4].

本稿では, 食品産業向け商品処理装置のために開発中であるソフトウエア更新システムのプロトコルに, BAN logic を適用した定理証明による検証の事例を報告する. まず, 2 節

[1]本研究は中小企業庁委託による中小企業支援型研究開発制度「安全なアップデートシステムの開発」により株式会社イシダと共同で行われた.

において BAN logic の概要を説明し, 3 節において対象となるシステムのプロトコル α の概要を説明する. 次に, 4 節においてプロトコル α の BAN logic における表現を与え, α が満たす安全性を示す. そして, 5 節において「商品処理装置は最新版の更新プログラムを受信するか, もしくはエラー通知を受信（送信）する」という性質を α が満たさない場合があるということ示し, この性質を常に満たすための案として, α を修正して得られた 2 つのプロトコル β と γ を示す.

2 BAN logic による検証

本節では BAN logic の概要を述べる. BAN logic は知識と信念を扱う論理であり, その論理式は以下の通りである：A, B をエージェント, X をメッセージ（BAN logic の論理式または後に与える述語のこと）とするとき,

A believes X	$\cdots$	A は X を信頼する,
A said X	$\cdots$	A は既に X を知っている（そして, A は X を送信した）,
A received X	$\cdots$	A は X を受信した,
A controls X	$\cdots$	A は X を正しく制御する.

表 1. BAN logic における論理式

を表す. そして, BAN logic は次の述語を含む. メッセージ X はプロトコルの実行前には現れていない新規のものであるとき,

$$\mathrm{fresh}(X)$$

と書く. 例えば, ナンス N(ランダムに生成された文字列) がそのような新規なものである場合, $\mathrm{fresh}(N)$ と書く. A と B が共通鍵 K_{AB} を持ち, かつ K_{AB} が A と B との間の秘密鍵として確実に機能しているとき, それを表す述語を BAN logic では次の様に表現する：

$$A \overset{K_{AB}}{\longleftrightarrow} B.$$

A が公開鍵 K_A を持ち, かつ K_A とその秘密鍵 K_A^{-1} が確実に機能しているとき, それを表す述語を

$$\mathrm{PK}(A, K_A)$$

と表す. これらの鍵によって暗号化されたメッセージ X は, それぞれ

$$\{X\}_{K_{AB}}, \quad \{X\}_{K_A}, \quad \{X\}_{K_A^{-1}}$$

と表す. 特に, $\{X\}_{K_A^{-1}}$ は公開鍵暗号系における秘密鍵を用いた A による署名付きメッセージ X を表している.

次に, BAN logic の推論規則を与える. 詳しくは, [1] や [4, Section 2] を参照.

Message Meaning

$$\frac{A \underline{\text{believes}}\ A \overset{K_{AB}}{\longleftrightarrow} B \quad A \underline{\text{received}}\ \{X\}_{K_{AB}}}{A \underline{\text{believes}}\ B \underline{\text{said}}\ X}\ (MM),$$

$$\frac{A \underline{\text{believes}}\ \text{PK}(B, K_B) \quad A \underline{\text{received}}\ \{X\}_{K_B^{-1}}}{A \underline{\text{believes}}\ B \underline{\text{said}}\ X}\ (MM)$$

1 つ目は, A が共通鍵 K_{AB} で暗号化された X を受信し, A が K_{AB} は B との秘密鍵として十分に機能していると信頼するならば, A は「B が X の正しさを信じて一度送信した (said)」ことを信じる, ということを表している. 2 つ目は, A が B によって署名された X (B の秘密鍵によって暗号化された X) を受信し, A は鍵 K_B と秘密鍵 K_B^{-1} が B の公開鍵暗号系における鍵として十分に機能していると信頼するならば, A は「B が X の正しさを信じて一度送信した (said)」ことを信じる, ということを表している.

Nonce Verification

$$\frac{A \underline{\text{believes}}\ \text{fresh}(X) \quad A \underline{\text{believes}}\ B \underline{\text{said}}\ X}{A \underline{\text{believes}}\ B \underline{\text{believes}}\ X}\ (NV)$$

これは, A は X がプロトコルの実行前には表れていない新規なメッセージであると信じていて, さらに A が「B が現時点までに X の正しさを信じて一度送信した (said)」ことを信じるならば, A は B が X を信頼していると信じる, ということを表している.

Jurisdiction

$$\frac{A \underline{\text{believes}}\ B \underline{\text{controls}}\ \text{X} \quad A \underline{\text{believes}}\ B \underline{\text{believes}}\ X}{A \underline{\text{believes}}\ X}\ (Ju)$$

これは, A は B が X を正しく制御していると信じていて, さらに A は「B が X を信じている」と信じているならば, A は X を信じる, ということを表している.

Belief Conjunction

$$\frac{A \underline{\text{believes}}\ X \quad A \underline{\text{believes}}\ Y}{A \underline{\text{believes}}\ (X, Y)}\ (BC)$$

$$\frac{A \underline{\text{believes}}\ B \underline{\text{believes}}\ (A, B)}{A \underline{\text{believes}}\ B \underline{\text{believes}}\ X}\ (BC), \quad \frac{A \underline{\text{believes}}\ B \underline{\text{said}}\ (X, Y)}{A \underline{\text{believes}}\ B \underline{\text{said}}\ X}\ (BC)$$

上の方について, A が X と Y それぞれを信頼するならば, それらを合わせた (X,Y) を信頼する, ということを表している. 下の方はいずれも, A が (X,Y) を信頼するならば, その部分である X と Y それぞれを信頼する, ということを表している.

Freshness Conjuncatenation

$$\frac{A \underline{\text{believes}}\ \text{fresh}(X)}{A \underline{\text{believes}}\ \text{fresh}(X, Y)}\ (FC)$$

これは, A が X がプロトコルの実行前には現れていない新規のものであると信じているならば, そのような新規な部分を含んでいる (X, Y) 全体をやはり新規なメッセージであると信じる, ということを表している.

Receiving Rules

$$\frac{A \underline{\text{believes}}\ A \overset{K_{AB}}{\longleftrightarrow} B \quad A \underline{\text{received}}\ \{X\}_{K_{AB}}}{A \underline{\text{received}}\ X}\ (RS), \quad \frac{A \underline{\text{received}}\ (X, Y)}{A \underline{\text{received}}\ X}\ (RS)$$

一つ目は, A が K_{AB} は B との秘密鍵として機能していると信頼しており, さらに A は K_{AB} によって暗号化された X を受信しているならば, A は X を受信している, ということを表す. 2 つ目は A が X と Y がペアになったメッセージ (X, Y) を受信したならば, X も受信していることを表す. Y についても同様である.

さて, 2 つの主体 A と B について, A から B へメッセージ X が送信された状況を考える. 我々はこのことを

$$A \to B :\ X$$

と表記する. このことを BAN logic の論理式で表すと

$$B \underline{\text{received}}\ \ X$$

となる. そして, A と B との間に共通鍵が存在するということは

$$A \underline{\text{believes}}\ A \overset{K_{AB}}{\longleftrightarrow} B, \quad B \underline{\text{believes}}\ A \overset{K_{AB}}{\longleftrightarrow} B,$$

という式で表され, また, B が公開鍵 K_B を持つことを A が知っているは

$$A \underline{\text{believes}}\ \text{PK}(B, K_B)$$

という式で表される. 上記のようなやりかたで, プロトコルを BAN logic の式へと翻訳することができる. これら前提となる式から, 成立つことを期待する性質が BAN logic の推論規則を用いて導出できる場合, その性質が証明されたという.

ただし, 現実のプロトコルをそのまま記述するのは煩雑な作業である. そこで, 検討したい性質に関連する部分を抽出し, BAN logic により定式化する. この関連部分の抽出は煩雑な作業を避けるためだけではなく, プロトコルのどの部分が注目する性質を導くのかを明らかにする効果もある.

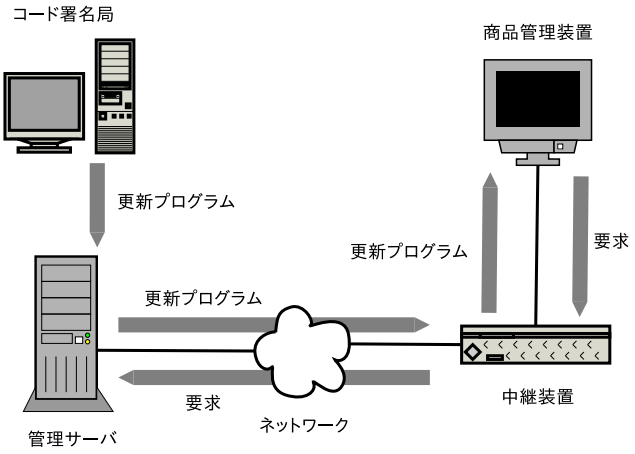

図 1: ソフトウエア更新システム

3　ソフトウエア更新プロトコルの概要

本節では検証の対象となったソフトウェア更新プロトコル α について述べる. プロトコル α は株式会社イシダが同社商品処理装置のソフトウェア更新のために開発しているものである. ソフトウェア更新システムはコード署名局, 管理サーバ, 中継装置, 商品処理装置の 4 つのエージェントより構成されている. これらエージェント間で交わされる通信プロトコルを簡略化して, BAN logic により定式化する. 図 1 に本システムの構成を, 図 2 にはエージェント間で交わされる通信のシーケンス図を示した. なお, 実際にはソフトウェアの更新が不要な場合と必要な場合とで異なったシーケンスとなるが, ここでは更新が必要な場合のシーケンスのみ示した.

この図において送受信されているファイルは次の 2 種類である

更新プログラム: 商品処理装置に改めてインストールされるプログラム.

INF ファイル: 商品処理装置に関する情報と, その商品処理装置にすでにインストールされているプログラムに関する情報が記載されている.

そして, このシステムは次の性質を持つ. まず, 商品処理装置と中継装置の間, 中継装置と管理サーバの間はそれぞれインターネットや LAN などのネットワークが介在しているが, コード署名局はそれらから隔離されており, データは人手により物理的に配送される. そして, 管理サーバは 1 つであるが, 商品処理装置や中継装置は複数存在し得る. しかし, 後の議論において登場する商品処理装置と中継装置は, 簡単のため 1 つずつにした.

商品処理装置と中継装置の間, 中継装置と管理サーバの間はそれぞれ SSL セッションが開かれており, SSL プロトコルによりあらかじめ交換された共通鍵による暗号化がなされる. コード署名局は INF ファイルと更新プログラムに対し, それぞれ正しいファイルであることを保証するため, 公開鍵暗号系における秘密鍵による署名を行う. INF ファイルや更新プログラムはコード署名局において署名された後, 管理サーバにネットワークを介さずに人手により置かれる. そして, 中継装置や商品処理装置は管理サーバから INF ファイル

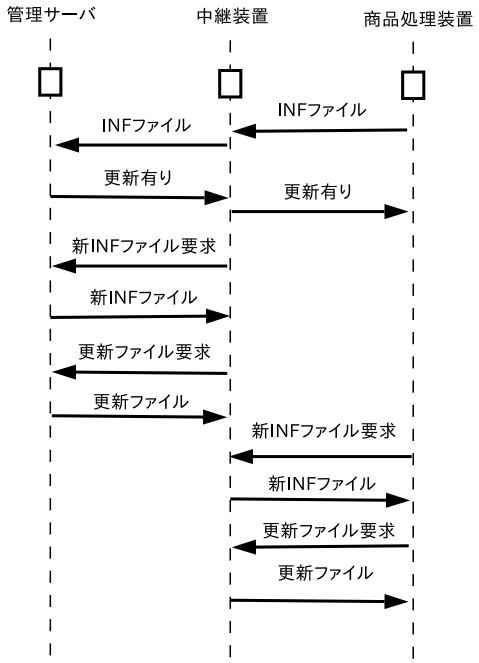

図 2: ソフトウエア更新プロトコル

や更新プログラムを受信したとき, コード署名局による署名がなされているかどうかを確認する. 同様に, 商品処理装置も現在の状況を表す INF ファイルを管理サーバに対して送信する際, 商品処理装置による署名を行う. その署名を中継装置や管理サーバは確認する.

ファイルの送受信や検証での失敗, またタイムアウトなどがあった場合, それぞれに対するエラーの種類を複数用意し, それぞれに対応して送信する.

さて, 本研究においてこのシステムに対して検討する性質は以下の通りである.

1. 管理サーバー側から見て, 商品処理装置の不正な要求にはエラーを返す.
2. 商品処理装置側から見て, 次のいずれかが成立すると信頼できる:
   (a) 商品処理装置が持つプログラムが, 管理サーバが持つ最新版に更新される,
   (b) エラーになる

これら性質について, 本研究では

- プロトコル α がこれらの性質を満たすか？
- または, α に対するどのような変更で上記の性質を示せるか？

ということを検討し, 次の結果を得た.

1. まず, プロトコル α の仕様から次が示される (4 節)：
   (a) 商品処理装置が受信したファイル（更新プログラム, 新 INF ファイル）はコード署名局を経て送信されたものであると, 商品処理装置側からも信頼できる.

(b) 管理サーバが受信した現行 INF ファイルは商品処理装置から送信されたものであると, 管理サーバ側からも信頼できる.

つまり, 公開鍵暗号系における秘密鍵による署名の効果がプロトコル α において現れていることを示すことができた.

2. 「商品処理装置側から見て, 更新したプログラムが実際に管理サーバが持つ最新版であると信頼できる」という性質について, プロトコル α の仕様からは示すことができない (5 節). 例えば, 中継装置が悪意を持っていて, 商品処理装置から送信された INF ファイルを途中にある中継装置が書き換えたり, また管理サーバから送信された更新プログラムを古い更新プログラムにすり替えたりした場合をプロトコル α は検出できない. そこで, どのような仕様の変更でその性質が導かれるかを検討し, 以下の変更によってその性質が導かれることを確認した (5 節).

   (a) 管理サーバが常に信頼できる場合 (5.1 節)：

      i. 商品処理装置から現行 INF ファイルを送信する際, 商品処理装置は新たにナンスを生成し, それも加えて送信する. そのとき, 商品処理装置はそれら 2 つを合わせて署名する.

      ii. 管理サーバは商品処理装置から送られたナンスと新 INF ファイルに対して合わせて署名し, それらを送信する. また, 更新プログラムについても同様に, その商品処理装置からのナンスと合わせて署名し送信する.

      iii. 商品処理装置は, 受信した INF ファイルや更新プログラムに添えられているナンスが, 自身が最近生成したものであるかどうかを確認し, 自身の最近生成したナンスであれば, それら INF ファイルや更新プログラムは最新のものであると商品処理装置は信頼できる.

   この変更を α に加えて得られたプロトコルを β と呼ぶ.

   (b) 管理サーバが信頼できない場合：コード署名局が商品処理装置ごとの要求に対応すると仮定 (5.2 節).

      i. 上記 (2(a)i) と同様に, 商品処理装置から現行 INF ファイルを送信する際, 商品処理装置は新たにナンスを生成し, それも加えて送信する. そのとき, 商品処理装置はそれら 2 つを合わせて署名する.

      ii. コード署名局は商品処理装置から送られたナンスと新 INF ファイルを合わせて署名する. また, 更新プログラムもそのナンスと合わせて署名する.

      iii. 上記 (2(a)iii) と同様に, 商品処理装置は受信した INF ファイルや更新プログラムに添えられているナンスが自身が最近生成したものかどうかを検証し, 自身が最近生成したナンスであれば, それら INF ファイルや更新プログラムは最新のものであると商品処理装置は信頼できる.

   ここで得られるプロトコルを γ と呼ぶ.

さて, プロトコルの仕様を BAN logic の論理式で表し, それらを始式として前述の推論規則を用いて BAN logic の論理式が導出されたとき, その BAN logic が表す性質がそのプロトコルの性質として認められる. 次節において, 図 2 において与えられているプロトコ

ル α の INF ファイルと更新プログラムのやり取りを BAN logic の記法で表し, そこから得られる帰結を示す.

4 プロトコル α の性質

記号を定める.

S:コード署名局, M:管理サーバ, P:中継装置, C:商品処理装置
f: 更新プログラム, **inf**, **inf'**: INF ファイル,
K_S, K_C: S と C それぞれの公開鍵, K_S^{-1}, K_C^{-1}: S と C それぞれの秘密鍵,
K_{PC}, K_{MP}: $P-C$ 間, $M-P$ 間それぞれの SSL コネクションで用いられている共通鍵.

さて, 前節で述べたように, ファイルの送信には主体間の秘密鍵によって暗号化がなされる. 例えば, 商品処理装置 C から中継装置 P への現行 INF ファイル **inf** の転送は

$$C \to P: \quad \{\mathbf{inf}\}_{K_{PC}}$$

と表現される. P と管理サーバ M の間の通信についても同様に必ずそれらの共通鍵によって暗号化されたものが送信される. また, 更新プログラムや新 INF ファイルにはコード署名局 S による署名がなされる. 例えば, M から P への新 INF ファイル $\mathbf{inf}'$ の転送は

$$M \to P: \quad \{\{\mathbf{inf}'\}_{K_S^{-1}}\}_{K_{MP}}$$

と表される. 以下は, 図 **2** で与えられている仕様を BAN logic の記法によって表現したものである.

4.1 プロトコル α の仕様

Message 1 $C \to P$: $\{\{\mathbf{inf}\}_{K_C^{-1}}\}_{K_{PC}}$
Message 2 $P \to M$: $\{\{\mathbf{inf}\}_{K_C^{-1}}\}_{K_{MP}}$
Message 3 $M \to P$: $\{\{\mathbf{inf}'\}_{K_S^{-1}}\}_{K_{MP}}$
Message 4 $M \to P$: $\{\{\mathbf{f}\}_{K_S^{-1}}\}_{K_{MP}}$
Message 5 $P \to C$: $\{\{\mathbf{inf}'\}_{K_S^{-1}}\}_{K_{PC}}$
Message 6 $P \to C$: $\{\{\mathbf{f}\}_{K_S^{-1}}\}_{K_{PC}}$

ここから, 次の公理 (ここでは、成立することが前提となる論理式) が得られる.

4.2 受信内容に関する公理

$A1$. P received $\{\{\mathbf{inf}\}_{K_C^{-1}}\}_{K_{PC}}$, $A2$. M received $\{\{\mathbf{inf}\}_{K_C^{-1}}\}_{K_{MP}}$,
$A3$. P received $\{\{\mathbf{inf}'\}_{K_S^{-1}}\}_{K_{MP}}$, $A4$. P received $\{\{\mathbf{f}\}_{K_S^{-1}}\}_{K_{MP}}$
$A5$. C received $\{\{\mathbf{inf}'\}_{K_S^{-1}}\}_{K_{PC}}$ $A6$. C received $\{\{\mathbf{f}\}_{K_S^{-1}}\}_{K_{PC}}$

4.3 鍵に関する公理

$A7.\quad C \ \underline{\text{believes}}\ \mathrm{PK}(S, K_S), \qquad A8.\quad C \ \underline{\text{believes}}\ P \overset{K_{PC}}{\longleftrightarrow} C,$

$A9.\quad M \ \underline{\text{believes}}\ \mathrm{PK}(C, K_C), \qquad A10.\quad M \ \underline{\text{believes}}\ M \overset{K_{MP}}{\longleftrightarrow} P,$

ここから, 次の BAN logic における証明図が得られる：

$$\dfrac{C \ \underline{\text{believes}}\ \mathrm{PK}(S, K_S) \qquad \dfrac{C \ \underline{\text{believes}}\ P \overset{K_{PC}}{\longleftrightarrow} C \quad C \ \underline{\text{received}}\ \{\{X\}_{K_S^{-1}}\}_{K_{PC}}}{C \ \underline{\text{received}}\ \{X\}_{K_S^{-1}}}\ (RS)}{C \ \underline{\text{believes}}\ S \ \underline{\text{said}}\ X}\ (MM) \qquad \ldots\ (PK)$$

ただし, $X = \mathbf{f}, \mathbf{inf}'$ である. つまり, C $\underline{\text{believes}}$ S $\underline{\text{said}}$ X が証明できる. また,

$$M \ \underline{\text{believes}}\ C \ \underline{\text{said}}\ \mathbf{inf}$$

に対しても (PK) と同様の証明図を与えることができる. このとき, 次のことが明らかになる：

1. 商品処理装置が受信したファイル（更新プログラム, 新 INF ファイル）はコード署名局を経て送信されたものであると, 商品処理装置側からも信頼できる.
2. 管理サーバが受信した現行 INF ファイルは商品処理装置から送信されたものであると, 管理サーバ側からも信頼できる.

つまり, 公開鍵暗号系における秘密鍵による署名の効果がプロトコル α の仕様において現れていることが示された.

5 商品処理装置側の安全性向上のための変更案

「商品処理装置側から見て, 更新したプログラムが実際に管理サーバが持つ最新版であると信頼できる」という性質について, 現行の仕様では示すことができない. 例えば, E_P が悪意を持った中継装置であり, 商品処理装置 C から送信された現行 INF ファイル $\mathbf{inf}$ を E_P が古い INF ファイル $\mathbf{inf}_{old}$ にすり替えて管理サーバ M に送信した場合, 現行の仕様では $\mathbf{inf}_{old}$ に対する新 INF ファイル $\mathbf{inf}'_{old}$ と更新プログラム $\mathbf{f}_{old}$ を C は受信することになり, 最新版を取得することができない. このことをこれまでに導入した記法を用いて表現すると, 以下のようになる.

5.0 悪意を持った中継装置 E_P が INF ファイル inf を古いファイル inf_{old} に差し替えるプロトコル

Message 1 $\quad C \to E_P$: $\quad \{\{\mathbf{inf}\}_{K_C^{-1}}\}_{K_{PC}}$

Message 2 $\quad E_P \to M$: $\quad \{\{\mathbf{inf}_{old}\}_{K_C^{-1}}\}_{K_{MP}}$

Message 3 $\quad M \to E_P$: $\quad \{\{\mathbf{inf}'_{old}\}_{K_S^{-1}}\}_{K_{MP}}$

Message 4 $\quad M \to E_P$: $\quad \{\{\mathbf{f}_{old}\}_{K_S^{-1}}\}_{K_{MP}}$

Message 5 $\quad E_P \to C$: $\quad \{\{\mathbf{inf}'_{old}\}_{K_S^{-1}}\}_{K_{PC}}$

Message 6 $\quad E_P \to C$: $\quad \{\{\mathbf{f}_{old}\}_{K_S^{-1}}\}_{K_{PC}}$

以下, このような場合を防ぐための仕様の変更案を, 商品処理装置側からみて管理サーバが信頼できる場合とそうでない場合とに分けて与える. いずれの場合も本質的な変更は, ファイルの送信の際, 商品処理装置が新規に与えたナンスとファイルを合わせて, 送信者が署名を行い送信することである. これは署名によりその作成者は明確になるが, それだけではその作成が現時点のものであるかどうかが明確にならないことから, 要求を発する商品処理装置が与えた新規ナンスとファイルを合わせて署名することによって, 現時点でのものであるかそうでないかを商品処理装置が判断できるようにするものである.

5.1 管理サーバーが常に信頼できる場合

ここで, 新たに述語を与える.

$\mathrm{uptodate}(x, C)$ $\cdots$ x は C に対して管理サーバが持つ最新版である

N をナンスとする.

5.1 プロトコル β

Message 1	$C \to P$:	$\{\{\mathbf{inf}, N\}_{K_C^{-1}}\}_{K_{PC}}$
Message 2	$P \to M$:	$\{\{\mathbf{inf}, N\}_{K_C^{-1}}\}_{K_{MP}}$
Message 3	$M \to P$:	$\{\{\{\mathbf{inf}'\}_{K_S^{-1}}, N, \mathrm{uptodate}(\mathbf{inf}', C)\}_{K_M^{-1}}\}_{K_{MP}}$
Message 4	$M \to P$:	$\{\{\{\mathbf{f}\}_{K_S^{-1}}, N, \mathrm{uptodate}(\mathbf{f}, C)\}_{K_M^{-1}}\}_{K_{MP}}$
Message 5	$P \to C$:	$\{\{\{\mathbf{inf}'\}_{K_S^{-1}}, N, \mathrm{uptodate}(\mathbf{inf}', C)\}_{K_M^{-1}}\}_{K_{PC}}$
Message 6	$P \to C$:	$\{\{\{\mathbf{f}\}_{K_S^{-1}}, N, \mathrm{uptodate}(\mathbf{f}, C)\}_{K_M^{-1}}\}_{K_{PC}}$

これをプロトコル β と呼ぶ. ここでナンス N は商品処理装置が毎回ランダムに生成するとする. 管理サーバが商品処理装置側から見て信頼できるという仮説がすでに与えられていることから, 以下の公理が得られる.

5.1 公理

1. C believes $\mathrm{fresh}(N)$.
2. C believes M controls $\mathrm{uptodate}(\mathbf{inf}, \mathrm{C})$.
3. C believes M controls $\mathrm{uptodate}(\mathbf{f}, \mathrm{C})$.

これらに加えて, 通信手続きから導かれる式を公理とする.

まず, p. 7 の証明図 (PK) と同様の推論を行い,

$$C \text{ believes } M \text{ said } (\mathbf{inf}, N, \mathrm{uptodate}(\mathbf{inf}, C))$$

が得られる. 次に, 上記の公理 5.1.(1) を得たとき, 次の証明図が与えられる.

$$\dfrac{\dfrac{C \text{ believes fresh}(N)}{C \text{ believes fresh}\,(\mathbf{inf}, N, \text{uptodate}(\mathbf{inf}, C))}\,(FC) \quad C \text{ believes } M \text{ said } (\mathbf{inf}, N, \text{uptodate}(\mathbf{inf}, C))}{\dfrac{C \text{ believes } M \text{ believes } (\mathbf{inf}, N, \text{uptodate}(\mathbf{inf}, C))}{C \text{ believes } M \text{ believes uptodate}(\mathbf{inf}, C)}\,(CC)}\,(NV)$$

ここで,

$$C \text{ believes } M \text{ believes uptodate}(\mathbf{inf}, C)$$

は管理サーバ M が INF ファイル **inf** が最新であると信じている, と商品処理装置 C が信じているという主張である. さらに, 公理 5.1.(2) を得たとき, 次の証明図が得られる.

$$\dfrac{C \text{ believes } M \text{ controls uptodate}(\mathbf{inf}, C) \quad C \text{ believes } M \text{ believes uptodate}(\mathbf{inf}, C)}{C \text{ believes uptodate}(\mathbf{inf}, C)}\,(Ju)$$

したがって, INF ファイル **inf** が商品処理装置 C にとっての最新版であることを C が信頼できることが示せた. 同様に,

$$C \text{ believes uptodate}(\mathbf{f}, C),$$

つまり, プログラム **f** が C にとっての最新版でことを C が信頼できることが示せる.

以上から, 商品処理装置が新規にナンスを生成して送信するというプロトコルの改良と, 管理サーバが正しい INF ファイルおよび更新ファイルを送信するとの仮定から, 例え中継装置に悪意があっても, 商品処理装置が受信できたファイルは改ざんされていない正しいものであることが商品処理装置から見て信頼できることが示せた.

なお 5.1 のプロトコル β では uptodate$(\mathbf{f}, C)$ というメッセージを送信しているが, **f** が最新版であることは管理サーバがそれを送信していることから明らかであるので, 実装上はこのメッセージは不要である.

5.2 コード署名局が個別の要求に対応する場合

ここでは, 管理サーバが信頼できない場合, 5.1 節において管理サーバが行っていたことをコード署名局が行うという形で仕様を変更するならば, 5.1 節と同様の結果を得ることができることを示す.

5.2 プロトコル γ

Message 3	$C \to P$:	$\{\{\mathbf{inf}_0, N\}_{K_C^{-1}}\}_{K_{PC}}$
Message 4	$P \to M$:	$\{\{\mathbf{inf}_0, N\}_{K_C^{-1}}\}_{K_{MP}}$
Message 5	$M \to P$:	$\{\{\mathbf{inf}', N, \text{uptodate}(\mathbf{inf}', C)\}_{K_S^{-1}}\}_{K_{MP}}$
Message 6	$M \to P$:	$\{\{\mathbf{f}, N, \text{uptodate}(\mathbf{f}, C)\}_{K_S^{-1}}\}_{K_{MP}}$
Message 7	$P \to C$:	$\{\{\{\mathbf{inf}', N, \text{uptodate}(\mathbf{inf}', C)\}_{K_S^{-1}}\}_{K_{PC}}$
Message 8	$P \to C$:	$\{\{\mathbf{f}, N, \text{uptodate}(\mathbf{f}, C)\}_{K_S^{-1}}\}_{K_{PC}}$

また, コード署名局の信頼性から次の公理が得られる.

5.2 公理

1. C believes fresh(N).
2. C believes S controls uptodate($\mathbf{inf}'$, C).
3. C believes S controls uptodate($\mathbf{f}$, C).

これに上のプロトコルから得られる公理を付け加える.

これらの公理から, 5.1 節と同様に

$$C \text{ believes uptodate}(\mathbf{inf}', C), \quad C \text{ believes uptodate}(\mathbf{f}, C)$$

が示される. そして, この場合, コード署名局に対する信頼性がこの結果の信頼性をもたらす. コード署名局はネットワークから隔離されているため, 管理サーバよりさらに信頼性は高い. よって, プロトコル γ はプロトコル β よりもより信頼性は高い. しかし, コード署名局と管理サーバーとがネットワークにより結ばれていないことから, プロトコル γ の実現は困難であると思われる.

6 結論

本研究では, プロトコル α の仕様から, 商品処理装置が受信した更新プログラムや新 INF ファイルはコード署名局を経て送信されたものであると, 商品処理装置側からも信頼できるということを示し, 公開鍵暗号系における秘密鍵による署名の効果がプロトコル α において現れていることを確認した.

次に, 商品処理装置側から見て, 「商品処理装置が持つプログラムが管理サーバが持つ最新版に更新されるか, もしくはエラーになるのいずれかである」とい性質がプロトコル α の仕様からは導かれないことを示し, 最新版であるかどうかを判定できるプロトコル β およびプロトコル γ を与えた.

なお, 本研究の成果は, 製品の改善の機会を与え, 安全性をより一層向上させるための示唆を与えたとイシダでは評価しているものの, 製品への検証は, テストなど通常の手法に従って別途行なわれており, 本研究での検証は, それに加えて製品の仕様書等について行なわれたものであること, 本研究の結果だけからは, 最終製品全体としての信頼性, 安全性について言及できないことを強調しておきたい.

本研究は株式会社イシダの協力のもとに行われた. ここに感謝したい.

参考文献

[1] Michael Burrows, Martin Abadi, Roger Needham, A logic of authentication, DEC SRC Research Preport 39, 1990.

[2] 暗号技術大全, Bruce Schneier 著, 山形浩生 監訳, ソフトバンク　パブリッシング, 2003.

[3] Etsuya Shibayama, Shigeki Hagihara, Naoki Kobayashi, Shin-ya Nishizaki, Kenjiro Taura and Takuo Watanabe, AnZenMail: A secure and certified e-mail system, Software Security - Theories and Systems, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2609, pp. 201-216, 2003.

[4] Paul Syverson and Iliano Cervesato, The logic of authentication protocols, Lecture Notes in Computer Science, 2171, 2001.

[5] 米田友洋・梶原誠司・土屋達弘, ディペンダブルシステム, 共立出版, 2005.

ソフトウェア更新システムプロトコルの BAN Logic による安全性検証 (Preliminary Version)
(算譜科学研究速報)
発行日 2007 年 6 月 1 日
編集・発行：独立行政法人産業技術総合研究所システム検証研究センター
同連絡先：〒563-8577 大阪府池田市緑丘 1-8-31
e-mail：informatics-inquiry@m.aist.go.jp

A case study on safety verification for the software update protocol by BAN logic (Preliminary Version)
June 1, 2007
Research Center for Verification and Semantics (CVS)
Ikeda Site
National Institute of Advanced Industrial Science and Technology (AIST)
1-8-31 Midorigaoka, Ikeda, Osaka, 563-8577, Japan
e-mail: informatics-inquiry@m.aist.go.jp

AIST01-J00022-69