

分散オブジェクトシステムにおける モバイルコード用セキュリティ機構の提案

鍛 忠 司[†] 洲 崎 誠 一[†] 梅 澤 克 之[†]
 近 藤 勝 彦^{††} 手 塚 悟[†]
 佐 々 木 良 一[†] 鬼 頭 昭^{†††}

モバイルコード (WWW サーバなどからダウンロードした際に自動的に実行されるという特徴を持ったプログラム) の利用は, ユーザ側の端末に特定のソフトウェアを導入する必要がないため, コンピュータシステムの保守やアップグレードが容易に行えるという利点を持っている. しかし, 分散オブジェクトシステムでモバイルコードを利用する場合には, 分散オブジェクトシステム, およびモバイルコードの各々のセキュリティ機能だけでは防止困難な新たなセキュリティ問題が発生する. 本論文では, 分散オブジェクトシステムにおいてモバイルコードを利用する場合に発生する, モバイルコードがユーザの権限を不正に利用できるというセキュリティ問題への対策の 1 つとして, SOA (Seamless Object Authentication) アプローチを提案する.

Seamless Object Authentication: A Security Mechanism for Mobile Codes on Distributed Object Systems

TADASHI KAJI,[†] SEIICHI SUSAKI,[†] KATSUYUKI UMEZAWA,[†]
 KATSUHIKO KONDO,^{††} SATORU TEZUKA,[†] RYOICHI SASAKI[†]
 and AKIRA KITO^{†††}

Mobile codes, which are programs that automatically run when they are downloaded from WWW server, has benefits for maintaining and upgrading the computer system, because it doesn't need to be installed in the user's computers. Although both distributed object systems and mobile codes have their own strong security functions. (ex. user authentication/authorization, communication protection, sandbox security, signed object and so on.) When those mobile codes are used in the distributed object system, new security problems arise. These security problems cannot be prevented by using security functions of distributed computing systems and mobile codes independently. In this paper, we propose SOA (Seamless Object Authentication) approach to protect above security problem such that the wicked mobile code could access the computer system by using the user's right.

1. はじめに

ネットワーク, 特に, インター/イントラネットの普及により, 多くのコンピュータがネットワークを介して相互に連携し, 処理を行う, 分散コンピューティングが一般的になってきた. そして, 分散コンピューティングのシステム形態は, 従来のクライアント・サーバ型のシステムから分散オブジェクトシステムへと変

化してきている. たとえば, CORBA¹⁾ や DCOM²⁾ は, そうした分散オブジェクトシステムの一例である.

また, ソフトウェア開発の面では, コンポーネント化技術が注目を集めている. これは, アプリケーションなどのソフトウェアやそれらが使用するデータなどを部品ごとにコンポーネント化する技術である. モバイルコードは, このコンポーネント化技術を利用したものであり, WWW サーバなどからダウンロードされると, 自動的に実行されるという特徴を持つ. モバイルコードの一例としては, Java アプレット³⁾ や ActiveX Control⁴⁾ などがある.

モバイルコードを利用したシステムは, ユーザ側のコンピュータにアプリケーションの基本コンポーネントだけをあらかじめインストールしておき, 基本部分

[†] 日立製作所システム開発研究所
Systems Development Laboratories, Hitachi, Ltd.

^{††} 日立システムアンドサービスプロダクトソリューション事業部
Product Solution Operations Division, Hitachi Systems
& Services, Ltd.

^{†††} 日立製作所ソフトウェア事業部
Software Division, Hitachi, Ltd.

以外の追加コンポーネントをアプリケーション起動時にサーバからダウンロードして動作する。その結果、サーバごとに異なるコンポーネントを用意することで、同じクライアントコンポーネントを様々な処理に使用できる。また、ユーザ側のコンピュータにインストールするソフトウェアは必要最小限のものでよいため、システムの保守やアップデートが容易になる。前述の分散オブジェクトシステムにもモバイルコードが用いられることが多くなってきている⁵⁾。

ところで、分散コンピューティングの発達により、ユーザ認証やアクセス制御、情報の暗号化などのセキュリティ技術⁶⁾に対する要求もこれまで以上に増している。分散オブジェクトシステムやモバイルコードも多くのセキュリティ機能を備えている^{7),8)}。たとえば、モバイルコードにデジタル署名を行った、“署名付きモバイルコード”はモバイルコードのセキュリティ機能の1つとしてよく知られている。

“署名付きモバイルコード”によって、機能豊富なモバイルコードがセキュリティを保ちながら利用可能になった。しかし、作成者やユーザの不注意などから新たなセキュリティ問題も発生している^{9)~11)}。

本論文で提案する SOA (Seamless Object Authentication) アプローチは、ユーザの不注意により発生するセキュリティ問題の1つで、分散オブジェクトシステムにおいて、不正な“署名付きモバイルコード”がユーザの権限を悪用し、不正な処理を行うという、セキュリティ問題に対するものである。

以下では、まず、2章で既存の分散オブジェクトシステム、モバイルコードのセキュリティ機能について述べ、3章で、2章で述べたセキュリティ機能だけでは防ぐことが困難な、ユーザの不注意によって引き起こされるセキュリティ問題について説明する。また、4章で3章のセキュリティ問題に対する対策として SOA アプローチを提案し、5章では、既存の分散オブジェクトシステムへの SOA の適用について説明する。さらに、6章で、既存の CORBA システムと、SOA を実装した CORBA システムの比較を行う。

2. 分散オブジェクトシステムとモバイルコードのセキュリティ機能

近年のコンピュータシステムでは、セキュリティは最も重要な機能の1つとなっている。分散オブジェクトシステムやモバイルコードも多くのセキュリティ機能を備えている。

2.1 分散オブジェクトシステムのセキュリティ機能

分散オブジェクトシステムは次のようなセキュリティ

機能を備えているものが多い⁷⁾。

- (1) ユーザやオブジェクトなどのプリンシパル(主体)の身元を確認する(認証機能)。
- (2) 身元を認証されたプリンシパルがオブジェクトなどの資源にアクセスする権限があるかどうかをチェックし、アクセスを制御する(アクセス制御機能)。
- (3) セキュリティに関する各種イベントを記録する(セキュリティ監査機能)。
- (4) クライアントとサーバとの間の通信を第三者から守る(通信保護機能)。
- (5) データの送受信を行ったことを後から否認できないよう、データの送受信の事実を証明する(否認不可機能)。
- (6) 管理者がセキュリティポリシーの設定などを行う(運用管理機能)。

2.2 モバイルコードのセキュリティ機能

モバイルコードもセキュリティを考慮して設計されているものが多い。中には、サンドボックスモデル¹²⁾と呼ばれる強力なセキュリティモデルを実装しているものもある。

サンドボックスモデルは、

- (1) モバイルコードがローカルな資源へアクセスすることは、基本的に認めない、
- (2) モバイルコードは、そのモバイルコードが保管されていたサーバとの間でしか通信できない、というものである。

しかし、上述のような強い制限はモバイルコードを利用するシステムの柔軟性を損なうものでもある。そのため、最近では、ネットワークを介してダウンロードしたモバイルコードが“署名付きモバイルコード”であり、かつ、その署名者が信用できるとユーザが認めた場合には、前述の制限は適用されないように機能拡張されてきている⁸⁾。

“署名付きモバイルコード”とは、モバイルコードにデジタル署名を行ったもので、プログラムの実行コードと、その実行コードのハッシュ値にデジタル署名したデータとがカプセル化されている。“署名付きモバイルコード”は、作成者を明らかにし、改ざんを防止するという利点を持っている。

3. 分散オブジェクトシステムでモバイルコードを用いる場合のセキュリティ問題

2章で述べたように、分散オブジェクトシステムやモバイルコードは様々なセキュリティ機能を持っている。しかし、作成者やユーザの不注意から、それらのセ

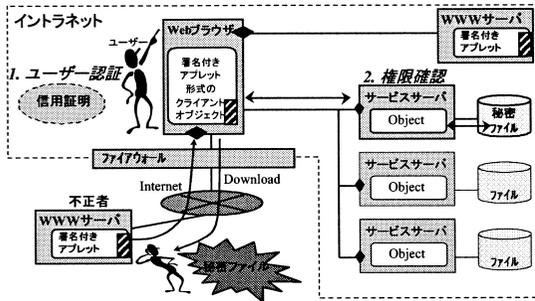


図 1 ユーザの不注意によって発生するセキュリティ問題の一例
Fig. 1 An example of a new security problem.

セキュリティ機能で防ぐことのできない新たなセキュリティ問題も発生している。

作成者の不注意によって発生するセキュリティ問題とは、プログラムの不備によって発生するセキュリティ問題である。たとえば、作成者の意図した以上の長さのデータを与えることによってバッファを溢れさせ、スタック上のデータを書き換える、などの問題が該当する。

一方、ユーザの不注意によって発生するセキュリティ問題とは、システム管理者が信用していない“署名付きモバイルコード”をユーザが誤まって信用し、実行してしまうことによって発生するセキュリティ問題である。これは、

- 署名付きモバイルコードには前述のモバイルコードのアクセス制限は適用されない、
- 分散オブジェクトシステムのセキュリティ機能は“ユーザ”によりアクセス制限を行う、

ことを利用し、権限を持っているユーザを踏み台にして、不正を行う手法である。

図 1 は、上記の問題が分散オブジェクトシステムに被害を与える一例である。

図 1 では、サービスサーバからユーザ端末にファイル転送を行うサービスが提供されている。ただし、転送が許可されるファイルは、ユーザの権限でアクセスが許可されたものに限定されている。

通常、ユーザは WWW サーバから署名付きモバイルコード形式のクライアントプログラムをダウンロードすることにより、このサービスを利用する。

一方、不正者は、自らの権限ではアクセスできないファイル(秘密ファイル)をサービスサーバからダウンロードし、コピーして不正者宛てに送信するような、不正な署名付きモバイルコードを生成し、不正者が管理する WWW サーバに保管している。

今、秘密ファイルにアクセスする権限を有するユー

ザが不正者の署名付きモバイルコードをダウンロードしたとする。ユーザが不正者が作成した署名付きモバイルコードを信用するように設定している場合、または、署名付きモバイルコードをダウンロードした時点で不正者を信頼すると設定した場合、署名付きモバイルコードは実行され、サービスサーバに秘密ファイルの転送を要求する。

このとき、署名付きモバイルコードは不正者の権限ではなく、ユーザの権限で動作するため、不正者は自分の権限ではアクセスできない秘密ファイルを不正に入手できる。

さらに、不正な動作を行う署名付きモバイルコードが、見かけ上、正常な署名付きモバイルコードと同様な動作をするようにしておけば、ユーザは、クライアントオブジェクトが不正な処理を行っていることにさえ気がつかないかもしれない。

4. SOA アプローチ

本章では、前章で述べたセキュリティ問題のうち、ユーザの不注意によって発生するセキュリティ問題への対策として提案する、SOA (Seamless Objectg Authentication) アプローチについて述べる。

前章で述べた、ユーザの不注意によって発生するセキュリティ問題は、サービス提供を要求してきたクライアントプログラム(モバイルコード)が不正な処理を行わないかどうかをユーザが検証するだけでなく、サーバ側でも検証することで防ぐことができる。

SOA アプローチは、クライアントプログラムに付加されている署名情報をユーザ端末からサーバに転送し、サーバ側でも署名者が信頼できるかどうかのチェックを行うことにより、サーバ側でもクライアントプログラムを検証する、というものである。

なお、SOA アプローチで“検証に成功する”とは、“サーバに転送されてきた署名情報が示す署名者がシステム管理者があらかじめ信頼できると設定した署名者一覧の中に含まれていること”である。

SOA アプローチを採用したシステム(図 2)では、ユーザ端末はサービスサーバにモバイルコードの署名情報を転送する機能を持ち、サービスサーバはユーザ端末から転送された署名情報を検証する機能を持っている。

これらによって、不正者の作成したモバイルコードがユーザの不注意によって実行され、サービスサーバにアクセスを試みた場合でも、サーバは不正な動作を行う可能性のあるクライアントプログラムがアクセスしてきたことを知り、サービスの提供を拒否する。

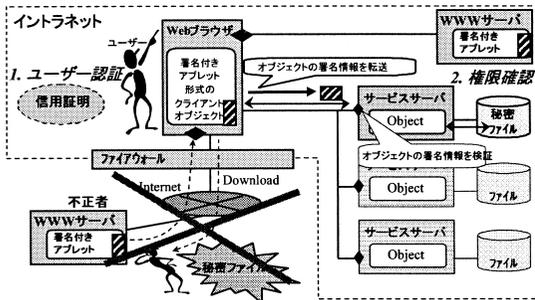


図 2 SOA アプローチ
Fig. 2 SOA approach.

5. CORBA システムへの適用

本章では、既存の分散オブジェクトシステム (Java ベースの CORBA システム) への SOA アプローチを適用する場合について説明する。以下、SOA アプローチを適用した CORBA システムを SOA 機構付き CORBA システムと呼ぶ。

5.1 Java ベースの CORBA システムの概要

まず、Java ベースの CORBA システムについて簡単に説明する。

5.1.1 Java ベースの CORBA システムの構成

Java ベースの CORBA システムは、図 3 に示すように、署名付きアプレット形式のクライアントオブジェクトが格納された“WWW サーバ”、ターゲットオブジェクトを実行する“サービスサーバ”、クライアントオブジェクトを実行する“ユーザ端末”、から構成される。

ターゲットオブジェクトは、クライアントオブジェクトに対してサービスを提供するオブジェクトである。

クライアントオブジェクトは、ターゲットオブジェクトにリクエストを送信し、サービスを楽しむオブジェクトである。

ORB は、ユーザ端末、サービスサーバに跨って動作し、次の 2 つの機能を持っている。

- 指定されたターゲットオブジェクトを探し、その位置情報をクライアントオブジェクトに通知する (バインド処理と呼ばれる)。
このとき、ターゲットオブジェクトが起動されていない場合には、ターゲットオブジェクトを起動する。
- クライアントオブジェクトとターゲットオブジェクトとの間の通信を媒介する。

5.1.2 Java ベースの CORBA システムの動作

次に、Java ベースの CORBA システムの動作について簡単に説明する。

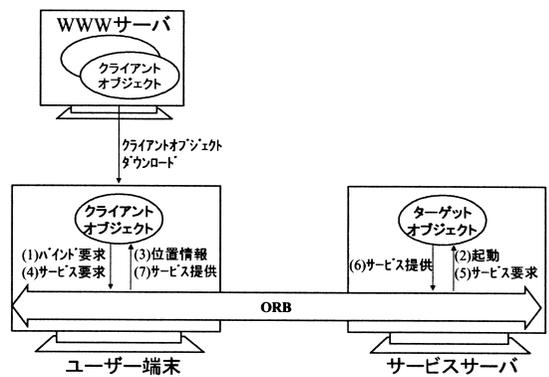


図 3 Java ベースの CORBA システム
Fig. 3 Java based CORBA system.

クライアントオブジェクトが WWW サーバからダウンロードされると、まず、ターゲットオブジェクトの位置を調べるため、バインド処理が行われる。Java ベースの CORBA システムでは、アプレットが、org.omg.CORBA.ORB クラスの bind() メソッドを呼び出すことにより、バインド処理を実行する。

バインド処理は、以下の手順で行われる (図 3 (1) ~ (3))。

- (1) クライアントオブジェクトが ORB に対して、ターゲットオブジェクトの位置探索を依頼するリクエストメッセージを発行。
- (2) ORB は、ターゲットオブジェクトを探し、ターゲットオブジェクトが起動されていない場合には、ターゲットオブジェクトを起動。
- (3) ORB は、ターゲットオブジェクトの位置情報をクライアントオブジェクトに返す。

次に、クライアントオブジェクトは、バインド処理によって得たターゲットオブジェクトの位置情報を利用し、ターゲットオブジェクトからサービスを楽しむ。一般には、この処理も ORB を介して行われる (図 3 (4) ~ (7))。すなわち、

- (4) クライアントオブジェクトが ORB にターゲットオブジェクトへのサービス要求メッセージを送信。
- (5) ORB はターゲットオブジェクトにサービス要求メッセージを配送。
- (6) ターゲットオブジェクトはクライアントオブジェクトへの応答メッセージを ORB に送信。
- (7) ORB は応答メッセージをクライアントオブジェクトに配送。

5.1.3 インタセプタ

多くの CORBA システムは、上述の基本的な構成に加え、様々な機能を付加できる、インタセプタとい

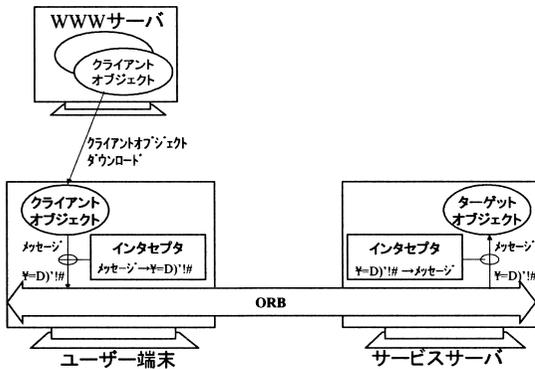


図4 インタセプタを備えた CORBA システム

Fig. 4 CORBA system with Interceptor.

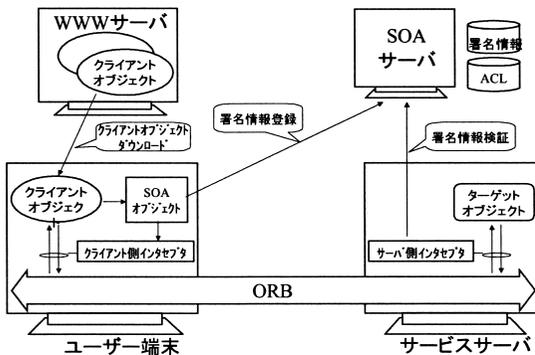


図5 SOA 機構付き CORBA システムの構成

Fig. 5 CORBA system with SOA mechanism.

う仕組みを持っている(図4)。

インタセプタは、クライアントオブジェクトと ORB との間、またはターゲットオブジェクトと ORB との間で動作し、メッセージを加工するなどの所定の操作を行うことにより、様々な付加機能を提供する。

なお、インタセプタは、バインド処理時に呼び出される“バインドインタセプタ”、クライアントオブジェクトがターゲットオブジェクトへのリクエストを ORB に送信する際に呼び出される“クライアントインタセプタ”、ターゲットオブジェクトがクライアントオブジェクトのリクエストを ORB から受信する際に呼び出される“サーバインタセプタ”、の3種類に分類される。

5.2 SOA 機構付き CORBA システム

次に、SOA 機構付き CORBA システムについて説明する。

5.2.1 SOA 機構付き CORBA システムの構成

SOA 機構付き CORBA システムは、前述の WWW サーバ、サービスサーバ、ユーザ端末に加え、SOA サーバから構成される(図5)。

SOA サーバはクライアントオブジェクトの署名情報を登録・管理するサーバである。SOA サーバの管理者は、サービス提供を許可する署名情報のリスト(ACL, Access Control List)をターゲットオブジェクトごとに作成し、SOA サーバにあらかじめ登録しておく。SOA サーバは、クライアントオブジェクトの署名情報と ACL とを比較し、サービスを提供するか否かを判定する。

なお、本論文では SOA サーバで集中的に署名を検証する方法を採用したが、各サービスサーバ上でサービスを提供するか否かの判定を行うことも可能である。SOA サーバを用いる方式はサービスサーバで判定を行う方式に比べ、SOA サーバが外部からの攻撃を受けないように厳重に管理しなければならないが、多くのサービスサーバが存在するシステムでも ACL を変更する場合の変更・配布の手間を小さくできる。

ユーザ端末では、クライアントオブジェクト、ORB、SOA オブジェクト、SOA 機構のためのバインドインタセプタ、およびクライアントインタセプタが動作している。SOA オブジェクトは、クライアントオブジェクトの署名情報を SOA サーバに登録し、SOA サーバから発行されたオブジェクト識別子(ID)をクライアントインタセプタに提供する。クライアントインタセプタは、ID をサービス要求メッセージに付加し、サービスサーバに転送する。

サービスサーバでは、サービスオブジェクト、ORB、SOA 機構のためのサーバインタセプタが動作している。サーバインタセプタは、サービス要求メッセージから ID を取り出し、SOA サーバにサービスを提供してよいかどうかを確認する。

なお、SOA 機構付き CORBA システムで用いられるクライアントオブジェクトは、SOA 機構を使用するように特別にプログラミングされた、署名付きアプレットでなければならない。これは、後述するように、バインド処理時に署名情報を偽りなく登録するためである。SOA 機構を使用するようにプログラミングされていない、署名付きアプレットや、署名されていないアプレットは、SOA 機構付き CORBA システムでは動作しない。

5.2.2 SOA 機構付き CORBA システムの動作

SOA 機構付き CORBA システムで、クライアントオブジェクトが WWW サーバからユーザ端末にダウンロードされると、まず、クライアントオブジェクトはバインド処理を行う。

図6に示すように、SOA 機構付き CORBA システムでのバインド処理は次の手順で行われる。

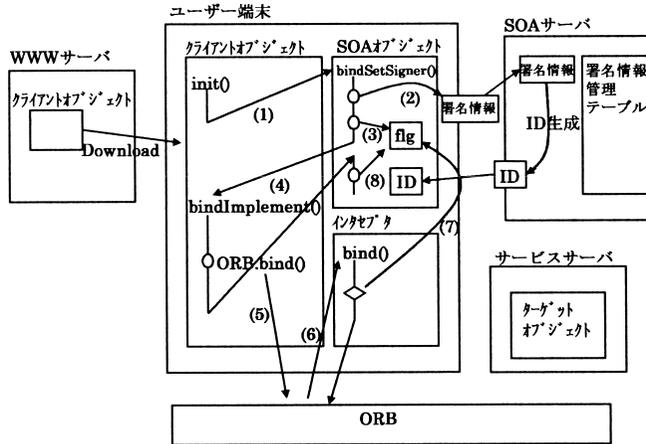


図 6 SOA 機構でのバインド処理
Fig. 6 Bind process on SOA mechanism.

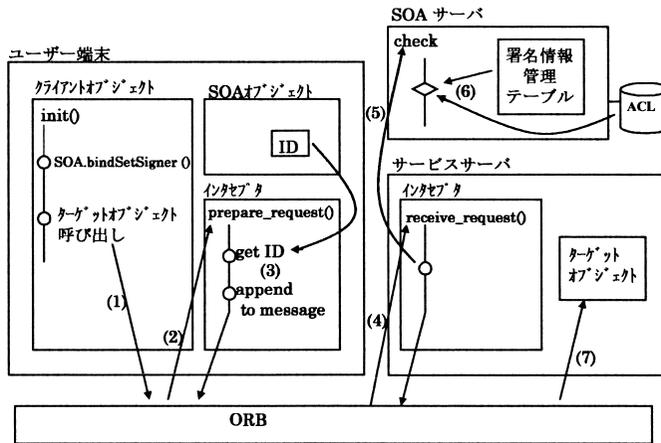


図 7 SOA 機構でのサービス要求処理
Fig. 7 Service Request process on SOA mechanism.

- (1) たとえば、init() メソッドからバインド処理を行う場合、クライアントオブジェクト自身への参照 (this) を引数として、SOA オブジェクトの bindSetSinger() メソッドを呼び出す。
 - (2) SOA オブジェクトは this から署名情報を取り出して、SOA サーバに登録する。このとき、SOA サーバは、署名情報と ID の対応表を作成し、ID を SOA オブジェクトに返す。
 - (3) SOA オブジェクトは ID を保存し、正規の SOA 機構の使用法に基づいてバインド処理が行われていることを確認するためのフラグを立てる。
 - (4) this のバインド用メソッド (bindImplement()) を呼び出す。このようにすることにより、(1) で他のオブジェクトを引数に渡し、成りすましを行うことを防止する。
 - (5) ORB に対してバインド処理を行う。
 - (6) ORB からインタセプタの bind() メソッドが呼び出される。
 - (7) 正規の SOA 機構の使用法に基づいてバインド処理が行われているかをフラグでチェックする。
 - (8) バインドの処理が終了したら、フラグを OFF にする (他のクライアントオブジェクトのバインド操作に利用されないようにするため)。
- 次に、クライアントオブジェクトは、バインド処理で得たターゲットオブジェクトの位置情報を利用し、ターゲットオブジェクトにサービスを要求する。
- 図 7 に示すように、SOA 機構付き CORBA システムでのサービス要求は次の手順で行われる。
- (1) クライアントオブジェクトがターゲットオブジェクトへのサービス要求メッセージを ORB に発行
 - (2) メッセージをサーバへ転送する前にインタセプトする。

表 1 サービス提供の可否比較

Table 1 Comparison between traditional CORBA system and CORBA system with SOA.

形式	クライアントオブジェクト		既存の CORBA システム	SOA 機構を持つ CORBA システム
	実行許可の有無			
	ユーザ	管理者		
SOA 用 アプレット	有	有		
	有	無		×
	無	無	×	×
署名付き アプレット	有	-		×
	無	-	×	×
署名なし アプレット	-	-	×	×

(...サービスを提供, ×...サービス提供を拒否)

- (3) SOA オブジェクトから ID を得て, メッセージに ID を付加する.
- (4) サービスサーバ側では, ターゲットオブジェクトがメッセージを受け取る前にインタセプトする.
- (5) メッセージから ID とサービス名を取得し, SOA サーバへ問い合わせる.
- (6) 署名情報管理テーブルから署名者を取得し, あらかじめ SOA サーバの管理者が設定した ACL で当該署名情報を持ったクライアントオブジェクトがサービスを受けられるかをチェックする.
- (7) チェック結果が OK なら, オブジェクトが呼び出される.

6. 既存の CORBA システムと SOA 機構付き CORBA システムの比較

表 1 に, 既存の CORBA 環境と, SOA 機構付きの CORBA 環境において, クライアントオブジェクトの種類ごとにターゲットオブジェクトがサービスを提供するか否かを比較した結果を示す.

既存の CORBA システムでも, クライアントオブジェクトが署名されていないアプレットや, 署名付きアプレットであってもユーザが実行を拒否した場合には, サービスは提供されない.

しかし, クライアントオブジェクトが署名付きアプレットで, かつユーザが実行を許可した場合には, ターゲットオブジェクトへのアクセスが許可される.

一方, SOA 機構を持った CORBA システムでは, クライアントオブジェクトが SOA 用アプレット(すなわち, SOA 機構を使用するようにプログラミングされた署名付きアプレット)でなければ, ユーザが実行を許可しても, ターゲットオブジェクトへアクセスすることはできない. また, クライアントオブジェクトが SOA 用アプレットであっても, ユーザが実行を許可し, さらに, クライアントオブジェクトの署名者が管理者があらかじめターゲットオブジェクトにサービ

ス提供を許可した署名者である場合にのみ, ターゲットオブジェクトにアクセスできる.

このように, SOA 機構を持った CORBA システムでは, 管理者の知らないところで作成された不正なクライアントオブジェクトがユーザの過失や不注意によって実行された場合でも, CORBA サービスを不正に利用することはできない.

7. ま と め

分散オブジェクトシステムにおいてモバイルコードを利用する場合には, プログラム作成者やユーザの不注意によって, 従来のセキュリティ機能では防ぐことが困難な新たなセキュリティ問題が発生する.

本論文では, そのようなセキュリティ問題の 1 つで, 署名付きモバイルコードがユーザの権限を悪用し, 不正を行うというセキュリティ問題について説明した.

また, 上記のセキュリティ問題への対策として, モバイルコードに付加された署名情報をサーバ側でも評価するという SOA アプローチを提案した.

SOA 機構を持った分散オブジェクトシステムでは, 管理者があらかじめ信頼できると設定した人物が作成したモバイルコードにのみサービスを提供し, 管理者が信頼していない人物が作成したモバイルコードにはサービスを提供しない. このため, 分散オブジェクトシステムの管理者の知らないところで作成された不正なモバイルコードがユーザの過失や不注意によって実行され, 分散オブジェクトシステムに危害が加えられるということを防止できる.

参 考 文 献

- 1) Mobray, T.J. and Malveau R.C.: *CORBA Design Pattern*, Jhon Wiley & Son's (1997).
- 2) 古山: DCOM ガイドブック, オーム社 (1997).
- 3) 井田: はやわかり Java, 共立出版 (1996).
- 4) Neou, V. and Pank, M.: *ActiveX Controls to*

Go: The Instant Toolkit for Web Site Developers, Prentice Hall (1997).

- 5) Orfali, R. and Harkey D.: *Client/Java Programming with Java and CORBA*, Jhon Wiley & Son's (1997).
- 6) 佐々木, 宝木, 櫻庭, 寺田, 浜田: インターネットセキュリティ, オーム社 (1996).
- 7) OMG: Security Service Specification, *CORBAservices: Common Object Services Specification*, OMG, Chapter 15 (1998).
- 8) Oaks S.: *JAVA Security*, O'Reilly (1998).
- 9) 勝村: モバイル・コードのセキュリティ対策, 日経インターネットテクノロジー, No.14, pp.96-103, 日経 BP (1998).
- 10) 兒島, 丸山: オブジェクトサイニングについての考察, 第 56 回情報処理学会全国大会論文集, Vol.1, pp.364-365 (1998).
- 11) 兒島, 丸山: Java パッケージシステムのセキュリティ, コンピュータセキュリティシンポジウム'98, pp.171-176 (1998).
- 12) Gong, L.: New Security Architectural Directions for Java (Extended Abstract), *Proceedings of IEEE COMPCON* (1997).

商標等に関する表示

- Java およびすべての Java 関連の商標は, 米国およびその他の国における米国 Sun Microsystems, Inc. の商標または登録商標です。
- JDK は, 米国およびその他の国における米国 Sun Microsystems, Inc. の商標または登録商標です。
- ActiveX は, 米国およびその他の国における米国 Microsoft Corp. の商標です。
- CORBA は, Object Management Group が提唱する分散処理環境アーキテクチャの名称です。
(平成 12 年 5 月 26 日受付)
(平成 13 年 1 月 11 日採録)



鍛 忠司

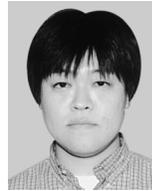
1996 年大阪大学大学院基礎工学研究科情報工学分野博士前期課程修了。同年(株)日立製作所勤務。企業情報システム, 分散オブジェクトシステムのセキュリティに関する研究開発に従事。

研究開発に従事。



洲崎 誠一(正会員)

1991 年横浜国立大学電子情報工学科卒業。同年(株)日立製作所システム開発研究所に入所。以来, 情報セキュリティ技術の研究開発に従事。現在, 同研究所セキュリティシステム研究センター研究員。1996 年情報処理学会第 52 回全国大会優秀賞, 平成 12 年度山下記念研究賞受賞。



梅澤 克之(正会員)

1996 年早稲田大学大学院理工学研究科機械工学専攻修士課程修了。同年(株)日立製作所システム開発研究所入所。以来, 企業情報システム, 分散オブジェクトシステムのセキュリティ技術等の研究・開発に従事。



近藤 勝彦

1986 年愛知県立一宮工業高等学校卒業。同年日立中部ソフトウェア(株)(現(株)日立システムアンドサービス)入社。主に通信システム系の開発業務に従事。1998 年 5 月(株)日立製作所システム開発研究所派遣。以来, 分散オブジェクトシステムのセキュリティ技術等の研究・開発に従事。



手塚 悟(正会員)

1984 年慶応義塾大学工学部数理工学科卒業。同年(株)日立製作所マイクロエレクトロニクス機器開発研究所を経て, 現在, システム開発研究所セキュリティシステム研究センター勤務。パーソナルコンピュータのオペレーティング・システム, デバイス・ドライバ, LAN システムの研究を経て, 現在, セキュリティシステム, 特に電子認証の研究開発に従事。工学博士。



佐々木良一(正会員)

1971年3月東京大学卒業。同年4月(株)日立製作所入所。システム開発研究所にてシステム高信頼化技術,セキュリティ技術,ネットワーク管理システム等の研究開発に従事。

同研究所セキュリティシステム研究センター長等を経て現在同研究所主管研究長。工学博士(東京大学)。1983年電気学会論文賞受賞。平成1998年電気学会著作賞受賞。著書に「インターネットセキュリティ入門」(岩波新書,1999年)、「インターネットコマース新動向と技術」(共編著,共立出版,2000年)等。IEEE,電子情報通信学会,電気学会等会員。情報処理学会コンピュータセキュリティ研究会主査。



鬼頭 昭(正会員)

1985年名古屋大学大学院工学部情報工学科卒業。同年(株)日立製作所勤務。企業情報システム,分散オブジェクトシステムの研究開発に従事。