システムソフトウェアと
オペレーティング・システム 76－4
（１９９７．８．２１）

オブジェクト指向分散環境 OZ の暗号化通信プロトコル

濱崎 陽一[1] 樋口 忠幸[2] 西岡 利博[3] 塚本 享治[1]
hamazaki@etl.go.jp thiguchi@fsi.co.jp nishioka@mri.co.jp tukamoto@etl.go.jp

1: 電子技術総合研究所 2: 富士ソフト ABC 3: 三菱総合研究所
〒305 つくば市梅園 1-1-4 〒247 鎌倉市岡本 2-13-18 〒100 千代田区大手町 2-3-6

ネットワーク上で情報のみならず、サービスやプログラムも共有できる、ワールドワイドプログラミング環境の実現を目指して開発を進めているオブジェクト指向分散環境 OZ では、セキュリティ機能が重要である。ここではネットワーク上を行き交う情報が窃取されることなくリモートメソッド起動を行うための暗号化通信について述べる。

OZ の暗号化通信では、アプリケーションレベルで暗号化器／復号器を実装することにより既存のソフトウェアパッケージ等の利用を容易にし、またネットワークを介してリモートメソッド起動を行える対象であるセルごとにそのセルが内蔵する情報の重要性に合わせた暗号化方式の設定をすることにより、細やかなセキュリティの設定を可能にしている。

また、プロトコルを工夫することで暗号化したリモートメソッド起動を最大二往復の通信で暗号化方式の交渉をも含めて完了することができる。

The Protocol of Enciphered Communication for OZ: An Object-Oriented Distributed Systems Environment

Yoichi Hamazaki [1] Tadayuki Higuchi [2] Toshihiro Nishioka[3] Michiharu Tukamoto [1]

1:Electrotechnical Laboratory 2:Fuji Soft ABC, Inc.
1-1-4, Umezono, Tsukuba, 301 Japan 2-13-18 Okamoto, Kamakura, 247 Japan
3:Mitsubishi Research Institute, Inc.
2-3-6 Otemachi, Chiyoda-ku, 100 Japan

This paper presents the design and the implementation of the enciphered communication for remote method invocation on OZ: an object-oriented distributed systems environment. To realize worldwide programming environment in which not only data but services and programmes also are shared on the networks is the aim of OZ. And in such environment, it is important to keep security. Enciphered communication described here is a OZ's security mechanism against the eavesdropping.

In OZ, encryptors and decryptors are implemented in the application level, so software package could be linked easily. The owner of cells, which are objects accessible over networks, can assign a set of ciphers to each cell according to the importance of data in it, so security setup could be fine-grained and flexible.

By introducing combined packet which contains data for negotiation and encypted data according to the cached information about cipher, a remote method invocation could be completed with two turn-around communication between caller and callee including negotiation of cipher in the worst case.

1 はじめに

OZは、ワールドワイドプログラミングをめざして開発を進めているオブジェクト指向分散環境である[1]。ワールドワイドプログラミングとは、ネットワーク上で情報のみならず、サービスやプログラムも共有し、ネットワーク上にプログラムやサービスを提供、蓄積、再利用することにより分散アプリケーションが容易に構築、実行できるような環境である。

ワールドワイドプログラミングでは、ネットワーク上を重要な情報が飛び交い、また他人の作ったプログラムが配送されて、それが実行される。そのために、セキュリティ機能が重要となる。OZは、OZ++ [2]システムの開発で得られた知見をもとに、よりセキュリティ機能を重視したシステムを目標としている。

OZでは認証に基づくセキュリティ[3]とオブジェクトレベルのセキュリティ[4]によって不法侵入や直接的なデータの破壊／窃取に対応し、本稿で述べる暗号化通信によって、ネットワーク上を行き交う情報を窃取されないようにする。

OZの暗号化通信は暗号化をアプリケーションレベルで行うことにより他の暗号化パッケージなどの利用を容易にしている。また、ユーザがオブジェクトの保持する情報の重要性に基づいて、オブジェクト毎に暗号化方式を設定する方式をとることにより細やかなセキュリティの設定が可能である。実装においては、暗号化通信のプロトコルを工夫し、暗号化方式の交渉も含めて最大二往復の通信で暗号化されたリモートメソッド起動ができる。

2 OZシステムの概要

OZシステムについて、暗号化通信に関係が深いセルとローカルオブジェクトおよびオブジェクトの名前づけを中心に概説する。

2.1 セルとローカルオブジェクト

OZのオブジェクト(インスタンス)には、ネットワーク上からのメソッド起動を受け付けることのできるセルと、セルを構成する部品であるローカルオブジェクトの2種類があり、これらのクラスを区別するモデルを用いている。

セルには広域ネットワーク上で識別できる固有の名前が付与されるが、それ自身がネットワーク上を移動することはできない。また、セルはその部品であるローカルオブジェクトと共に永続化することが可能である。一方、ローカルオブジェクトはメソッド起動の引数や返り値としてネットワークを介してコピーされた実体が転送される。数値などの基本型のデータもローカルオブジェクトと同じく転送されるので、以下では基本型のデータも含めた意味でローカルオブジェクトの語を用いる。

セルは、そのセルを生成した利用者の所有に属する。

2.2 セルの識別子

OZシステムの想定している計算機環境は以下のようなものである。

分散環境の利用者は何らかの組織(サイト)に属し、その組織内のネットワークで接続された計算機を利用できる。複数の計算機を同一のアカウントで利用できる場合には、それらの計算機の間では同じファイルシステムを利用者が利用できるようにNFSなどのファイル共有システムが稼働している。

そこで、利用者のアカウントおよびそのファイルシステムを抽象したOZホームを導入した。サイトはその識別子としてDNSドメイン名を持つ。OZホームはサイト内で識別されるためのOZホーム識別子を持つ。OZホームのイメージを図1に示す。

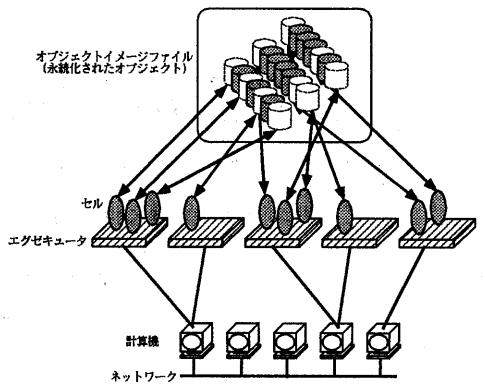

図1: OZホームの構成

セルが永続化された際には、オブジェクトをシリアライズしたものを格納したファイル(オブジェクトイメージファイル)という形態でOZホームに存在する。セルに付属してその状態を管理するためのファイルも必要なために、一つのセルに対して一つのディレクトリを割り当て、OZホーム内ではそのディレクトリ名でセルを識別することとした。このディレクト

リ名をドット区切り形式で表したものをGOD(Global Object Directory) と呼ぶ。

永続化されているセルはオブジェクトイメージファイルからエグゼキュータ上に読み込まれて活性化されることにより他のセルからのリモートメソッド起動を受けられる状態になる。活性化されているセルが同時に他のエグゼキュータ上で活性化されることはない。またセルはエグゼキュータ上から非活性化されてファイル上に永続化される。

OZ ではセルを利用者が利用可能な任意の計算機上のエグゼキュータで実行可能とした。これは、セルに対してそれを実行するエグゼキュータあるいは計算機を固定すると、廃棄された計算機と関係づけられていたセルが起動できなくなる、また新規導入された計算機を既存のセルで利用する事もできないなどの問題を生じ、昨今の変化のはげしい計算機環境にはそぐわないからである。

セルはその所有者である利用者のOZ ホームに属し、OZ ホームはサイトに属す。よって、セルのネットワーク上での識別子 GOL(Global Object Locator) はサイトの識別子、OZ ホーム識別子、セルのGOD の三つをコロンで区切って表すものとした。サイト、OZ ホーム、セルとそれらの識別子などの関係を表すクラス図を図2に、GOL の例を図3に示す。セルが稼働する計算機は活性化ごとに変わる可能性があるので、URL をセルの識別に用いることは不適切である。

セルはその GOL によって識別され、その所有者は GOL に含まれる DNS ドメイン名および OZ ホーム識別子からアクセスする前に明らかとなる。

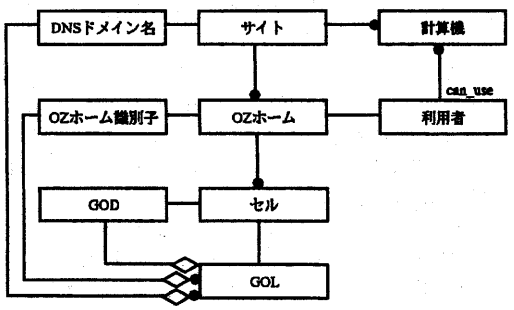

図 2: サイト、OZ ホーム、セルのクラス図

リモートメソッド起動は、異るエグゼキュータ上で稼働しているセル間のメソッド起動である。引数あるいは返り値としてオブジェクトを渡す際には、セルは

```
etl.go.jp:hamazaki:server.directory.name
```

図 3: Global Object Locator(GOL) の例

その参照である GOL を渡すことによって、ローカルオブジェクトはそれが直接あるいは間接に参照しているローカルオブジェクトも含めてコピーが渡される。

リモートメソッド起動は、呼び出し側から Call パケットが送られ、メソッドが実行され、そのメソッド実行の結果として Result パケットあるいは Exception パケットが返されることにより実現される。エグゼキュータは GOL から相手のエグゼキュータの通信アドレスを解決する [5]。Call パケットには、個々のリモートメソッド起動を識別するセッション ID(SID), 双方の GOL、メソッド起動の連鎖の識別子 (PID) と、呼び出すべきメソッドのセレクタおよび必要な引数を含んだプロクシオブジェクトが含まれる。Result パケット、Exception パケットにも先頭に SID が含まれる。リモートメソッド起動に用いるパケットを図4に示す。

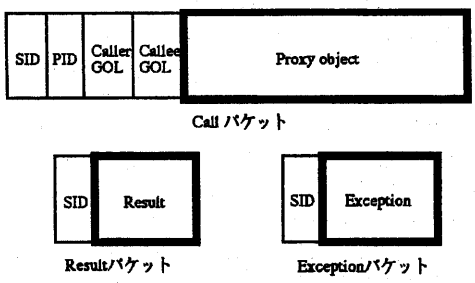

図 4: リモートメソッド起動のパケット

3 暗号化通信

OZ ではリモートメソッド起動の際には、引数あるいは返り値としてローカルオブジェクトがネットワーク上を転送される。そうしたローカルオブジェクトには、他人に窃取されたり、改竄されてはならない重要なものがあり、それらを保護するのが本稿の暗号化通信の目的である。

最初に暗号化方式設定の単位と通信に用いる暗号化方式を決定する交渉について考察を述べ、その実装を述べる。実装にあたってはプロトコルを工夫し暗号化方式の交渉も含めて最大でも二往復の通信で暗号化されたリモートメソッド起動ができる。

3.1 暗号化方式の設定

OZ ではシステムの柔軟性の観点から、標準として用いる暗号化方式を固定するのではなく、利用者が自由に選択できるようにした。そのために、暗号化はネットワーク階層の中で行うのではなく、アプリケーションレベルで行うとの方針をとった。OZ では暗号化器／復号器はアプリケーションレベルで実現され、実装言語である Java のクラスとして提供されるので、既存の暗号化ライブラリなどの利用が容易である。暗号化方式はクラスの識別子により識別される。

暗号化方式は、その暗号強度と暗号化によるオーバヘッドの観点から選択されるが、その設定の単位としては次の 3 つが考えられる。

1. 利用者ごと
2. セルごと
3. セルのメソッドごと

1. 利用者ごとでは、一つ機密を要する情報を保持したとたんに、機密を要しない他の情報までも同じレベルで保護しなければならなくなり、設定の単位が粗すぎる。また 3. メソッドごとではそのセルのクラスを設計したプログラマはセルの内蔵する情報の重要性を事前に予測することはできず、また利用者はプログラムの詳細を知る事が困難であるので、適切にセキュリティを設定する事は事実上不可能である。そこで、2. セルごとに設定することとした。同一のクラスのセルであっても、そのセルが内蔵する情報 (ローカルオブジェクト) によって重要性は異ってくるが、セル単位の設定であれば問題が生じない。

利用者はセルごとにそのセルが内蔵する情報の重要性に鑑み、受容できる強度の暗号化方式のリストを登録する。このリストはこのセルを出入りする情報に要求される暗号強度を規定するので、「要求暗号化方式リスト」と呼ぶ。空リストの場合は、そのセルを出入りする情報に対する保護を必要としない事を意味している。

一方、暗号化を用いて利用者が相互に認証するためには、利用者の認証サーバへの登録が必要となる。ここでいう認証サーバとは、秘密鍵暗号系で用いられる KDS(Kerberos の Key 配布センター) や公開鍵暗号系で用いられる公開鍵サーバなどを指す。本稿では認証サーバについてはふれず、暗号化通信のための会話鍵を得る方法があるものとして話を進める。

利用者が利用できる暗号化方式、すなわち暗号化器／復号器を所持しており、かつその方式の認証サーバへの登録が終了している暗号化方式は利用者 (OZ ホーム) ごとに異る。また暗号化／復号に特殊なハードウェアを必要とする暗号化方式ではそれが利用できる計算機が限られる場合がある。そこで、利用者が使用可能な暗号化方式のリストは計算機毎に設定することとした。このリストを「利用可能暗号化方式リスト」と呼ぶ。

OZ ホーム、セルと暗号化方式の対応についてクラス図にまとめたものを図 5に示す。

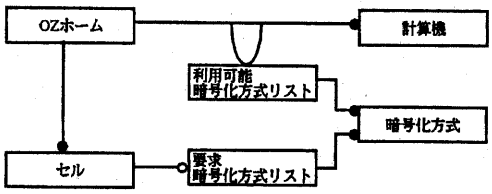

図 5: OZ ホーム、セルと暗号化方式のクラス図

3.2 暗号化方式の交渉

リモートメソッド起動には、呼び出し側と呼び出される側の二つのセルが関係し、両方向の情報伝達 (ローカルオブジェクトの授受) が行われる。そうした情報を暗号化して通信するためには、双方のセルが許容できる共通の暗号化方式が必要であり、暗号化に先だって交渉が必要になる。

セルの持つ「要求暗号化方式リスト」は、そのセルが保持する情報を保護するためにセルを出入りする情報に対してユーザが必要と考える暗号強度を満たす暗号化方式のリストである。

保護を必要としていないセル inSecure から、高い暗号強度を求めているセル verySecure へアクセスする場合を考える。verySecure から inSecure に返される情報 (返り値もしくは例外) について高い暗号強度が必要なのは明らかである。

inSecure から verySecure に渡される情報 (引数) についてはどうであろうか。verySecure が内蔵する重要な情報も、元をただせば他のセルに源がある。inSecure のように保護を必要としないセルからだといって保護することなく情報を受け取るとしたら、改竄された情報を受け入れ事による情報の劣化／破壊や、盗聴によって得られた部分情報を蓄積、統合することによる機密情報の漏洩などのセキュリティ上の問題が発生する。

つまり、「要求暗号化方式リスト」はセルの内蔵し

ている情報の重要性に鑑み設定されるが、それはセルから出て行く情報のみならず、セルに入って来る情報をも同様に保護すべきであるとの要求と考えられる。

よって、情報の伝播方向にかかわらず同一の暗号化方式が使われるべきであり、それは二つのセルの「要求暗号化方式リスト」(それが空の場合は「利用可能暗号化方式リスト」)の積集合から選択する。

暗号化方式の決定は、双方の「要求暗号化方式リスト」の積集合を作り、その中から選択することにより行われるので、その交渉の手順は以下のようになる。暗号化方式が決定されれば、それに従って情報を暗号化してリモートメソッド起動の通信をおこなう。

1. 呼び出し側がそのセルの「要求暗号化方式リスト」を送る。
2. 呼び出された側で自身の「要求暗号化方式リスト」と受信したリストとの積を取り、そのなかから適切なものを選択する。
3. 選択した暗号化方式を呼び出し側に知らせる、あるいは双方の要求を満たす暗号方式が見つからなかった事を知らせる。

3.3 暗号化通信の実装

暗号化通信では、先に述べたように暗号化方式を交渉し、交渉で決まった暗号化方式によって暗号化した情報をやりとりする。暗号化通信で保護するのはネットワークを転送されるローカルオブジェクトである。つまり、図4に示したパケットの太線で囲んだ部分、引数あるいは返り値がそれである。また、太線の外の部分はパケットを転送するのに必要な情報を含んでおり、それらを暗号化してしまうとアプリケーションゲートウェイなどの中継器の実現が困難になる。

あるセルから別のセルに繰り返しリモートメソッド起動する場合、その暗号化方式が毎回変更される事は考えにくい。とすれば前回利用した暗号化方式を記録しておき、それを利用して暗号化したものを送れば成功する確率が高く、交渉を省略できる可能性がある。しかし、暗号化方式の設定に変更があり、受信を拒絶された場合には暗号化方式の交渉から始めねばならないためにペナルティが大きい。そこで記録しておいた前回の方式を用いる方法を修正して、交渉に必要な情報と前回成功した方式で暗号化した情報の両方を混合パケットで送る混合方式を考案した。3方式の比較を表1に示す。考案した方式は暗号化したデータと共に交渉に用いる暗号化方式リストをも転送するのでデータ転送のオーバヘッドが余分にかかるが、最悪でも2往復の通信で済み、多くの場合は1往復の通信でリモートメソッド起動ができる点で優れている。特に広域に分散した環境では一往復の通信オーバヘッドが大きいので提案の方式が有利である。

表 1: 暗号化交渉方式の比較

	最善の場合	最悪の場合
毎回交渉	2往復 (交渉、データ)	2往復 (交渉、データ)
前回の記録	1往復 (データ)	3往復 (失敗、交渉、データ)
混合方式	1往復 (データ)	2往復 (失敗と交渉、データ)

暗号化通信に使われるパケットのうち主なものを図6に示し、暗号化通信の代表的な手順を図7に示す。交渉リスト付き暗号文パケットおよび交渉リスト付き平文パケットが先に述べた混合パケットにあたる。

始めてリモートメソッド起動をする際(図7の(a))には、呼び出し側のセルの要求暗号化方式リストの内容を交渉リストパケットによって送る。呼び出された側では、セルの要求暗号化方式リストとそのOZホームの利用可能暗号化方式リストの積集合と、受信したリストを比較し、双方で一致した中で一番オーバヘッドの少ない(多くの場合、一番暗号強度の弱い)方式を選択して、それを交渉結果パケットによって返送する。もしも選択できる暗号化方式がなければ、交渉は失敗である。

交渉された暗号化方式を用いて暗号化しデータ転送を行う。呼び出し側では会話鍵と暗号文を暗号文(往)パケットで送り、結果を暗号文(復)パケットで受け取る。リモートメソッド起動が終了したら、暗号化方式を記録しておく。

二度目以降のリモートメソッド起動では記録された暗号化方式を用いて暗号化し、それに交渉リストを付加した交渉リスト付き暗号文パケットを送る。もしも、キャッシュされていたその暗号化方式が受け入れられるものなら結果が暗号文(復)パケットによって返される(図7の(c))。この場合、交渉リスト付き暗号文パケットの交渉リストの部分は利用されない。

キャッシュが無効の場合(図7の(b))には、交渉リスト付き暗号文パケットの暗号文の部分が捨てられて、交渉リストの部分を用いて暗号化方式の選定を行い、その結果を交渉結果パケットによって返送する。

あとは一度目のリモートメソッド起動の場合と同様に、暗号文(往)、暗号文(復)によってデータが送られる。

セルが要求暗号化方式リストが空である場合は、デフォールト値としてそのOZホームの利用可能暗号化方式リストを用いる。利用可能暗号化方式リストには、暗号化せず平文のまま送信する非暗号化が方式の一つとして必ず含まれており、双方のセルの要求暗号化方式リストが空の場合には、平文による通信が行われる。

また、始めてリモートメソッド起動をする際であっても、呼び出し側のセルの要求暗号化方式リストが空であれば、交渉リスト付き平文パケットを用いて、相手の要求暗号化方式リストが空であることを期待する。この場合、平文で情報がネットワークをながれるが、その機会は各セルの対において高々1回であり、そのセルの要求暗号化方式リストが空であることから、セキュリティ上の危険性は低いものと判断した。

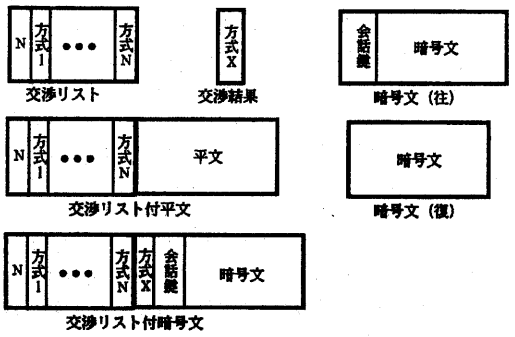

図 6: 暗号化通信のパケット

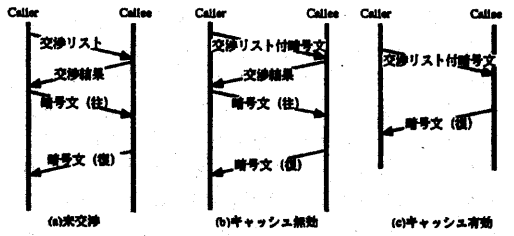

図 7: 暗号化通信の手順

4 まとめ

オブジェクト指向分散環境OZのセキュリティ機能の一つである暗号化通信について述べた。

暗号化器/復号器をアプリケーションレベルで提供するので既存の暗号化パッケージなどとの連係が容易であり、利用者がセルごとにそれが内蔵する情報の重要性に基づいて暗号化方式を設定できるために、細やかにセキュリティの設定をできるところに特長がある。またOZではクラスをもネットワーク上で共有できるが、暗号化方式の設定はインスタンス(セル)ごとであるのでクラスの共有になんら問題がない。

また、その実装にあたってはプロトコルを工夫し最大でも二往復の通信で暗号化方式の交渉も含めた暗号化通信によるリモートメソッド起動が可能である。

OZはJavaを用いて実装されているが、Javaでは認証などのフレームワークが現在までに暗号化通信の機構の実装を行い、今後認証サーバとの連結、暗号化器/復号器の試作を計画している。

OZシステムは本年度末を目標に開発が進められており、エグゼキュータなどの基本部分が実装されている。現在、OZシステムの最初の版(1.0 α)がインターネットで公開されており、順次、機能拡張した版を公開する予定である[6]。

本研究は、情報処理振興事業協会「創造的ソフトウェア育成事業」の一環として行われたものである。

参考文献

[1] 濱崎、西岡、塚本:「ワールドワイドな分散オブジェクト指向環境の構想」、情報処理学会 研究報告 96-DPS-76-7、May.1996

[2] 塚本、濱崎、音川、西岡:「クラスの共有と配送にもとづくオブジェクト指向分散システムの設計と実現」、情報処理学会 論文誌、May.1996

[3] 西岡、濱崎、塚本:「オブジェクト指向分散環境OZ のセキュリティモデル」、情報処理学会 研究報告(PRO)、SWoPP'97、Aug.1997

[4] 濱崎、西岡、塚本:「オブジェクト指向分散環境OZ のプロセス内セキュリティ」、情報処理学会第54回全国大会、Mar.1997

[5] 杉野、西岡、中川、塚本:「オブジェクト指向分散環境OZ のアドレス解決機構」、情報処理学会第54回全国大会、Mar.1997

[6] OZ プロジェクトのホームページ:
http://www.etl.go.jp/etl/bunsan/OZ_Proj/