IONA Technologies

Making Software Work Together™

# OrbixSecurity 3.0 White Paper

IONA Technologies
September 1999


Making Software Work Together™



m2371

# Summary

This white paper introduces OrbixSecurity 3.0, from IONA Technologies. OrbixSecurity is an open standards-based technology for developing and deploying security services for CORBA objects that also includes a sophisticated and scalable management interface.

This white paper begins by explaining how the CORBA-based Orbix object architecture is a solution to the crisis facing traditional mainframe and client-server. Unfortunately, every distributed solution incurs a trade-off between the degree of security and the degree of integration.

This paper then describes how OrbixSecurity successfully resolves this conflict, simplifying the administration and scalability of enterprise networks while reducing security risks.

# Audience

This paper addresses information services managers — particularly those now moving to distributed environments and those new to the issues of information security.

The **Introduction** details the necessity for a distributed system approach and briefly describes the market-leading features that OrbixSecurity brings to this environment.

**Part One** discusses distributed-system security in a business context.

**Part Two** explores new ways of thinking about information security in distributed environments—focusing on how to develop practical, comprehensive approaches using new and existing technologies.

**Part Three** describes in detail the enterprise solution available through IONA Technologies 'OrbixSecurity.

# Table of Contents

# Introduction

Technological and economic imperatives are making it increasingly beneficial for enterprises to decentralize their information systems. The traditional mainframe and homogenous client-server enterprise models have become ineffective for world-class companies seeking to compete in today's flexible and rapidly changing business environment.

## Orbix

To facilitate the integration of heterogeneous computing systems, the Object Management Group (OMG) defined the Common Object Request Broker Architecture (CORBA). The core component, the Object Request Broker (ORB), is the foundation for delivering platform-neutral interoperability across business and enterprise systems. Its language-neutral behavior distinguishes it from comparative Java solutions. Orbix is IONA Technologies 'ORB implementation, available on a wide variety of platforms. For further information, see the Orbix and OrbixWeb white papers.

## OrbixOTM

The CORBA specification extends the basic ORB infrastructure by specifying a wide range of CORBAservices and CORBAfacilities. OMG-defined interfaces here allow CORBA developers to provide event management, naming, messaging, object persistence, transaction, and security management services standardized to interoperate with ORBs from different vendors. OrbixOTM (Object Transaction Monitor) is IONA's implementation of an enterprise-level CORBA-standard application server integrated development environment. For further information, please see the OrbixOTM white paper.

## OrbixSecurity

CORBA provides a powerful paradigm for developing distributed, object-oriented applications. Using a CORBA-compliant ORB, a client application can easily communicate with resources and services distributed throughout a network; these resources and services are modeled as objects that are located transparently by the ORB on behalf of the client.

This distributed model facilitates flexibility and delivers competitive advantage. However, distributed environments by design may introduce potential security holes across the enterprise. Security protection makes systems more rigid and, as

a result, distributed systems have a required trade-off between the degree of security and the desired level of integration.

OrbixSecurity is IONA ŝ implementation of an enterprise security solution that meets and exceeds the rigorous demands of the OMG ŝ CORBA Security Level 2 standard. With extensive security policy management features that allow security administrators to construct, analyze and test their policy implementations, OrbixSecurity also features detailed auditing features that allow extensive user monitoring and enforce rigid accountability and non-repudiation throughout the secured system.

0rbixSecurity simplifies and eases the user experience with unitary login features that ensure that a single authenticating user login can be propagated to specific systems, thus avoiding irritating multiple login requirements. User privileges can be delegated to necessary principals, while the security administrator ŝ fine-grained access control and authorization allows precise user control and monitoring.

OrbixSecurity ŝ additional login facilities, such as the token-based SecureID or certificate-based systems, decrease system vulnerabilities, while mutual authentication, confidentiality, and data integrity is assured through the use throughout of low-level Secure Sockets Layer (SSL) encryption using RSA protocols.

For developers, OrbixSecurity delivers an open, interoperable security solution that allows secure communication and interaction with other ORB implementations within a distributed framework. OrbixSecurity provides an advanced security development environment for new applications, and also facilitates the efficient integration of legacy applications and systems within the enterprise security framework.

# Part 1: Security as an Enabler for Network Applications

## Network Applications Drive Competitiveness

Corporations are discovering the power of online services to increase customer loyalty, support sales efforts and manage internal information. The common thread in these diverse efforts is the need to present end-users with a unified view of information stored in multiple systems, particularly as organizations move from static websites to the transactional capabilities of electronic commerce.

To satisfy this need, legacy systems are being integrated with powerful new Network-based applications that provide connectivity across a multitude of back-end systems. These unified applications bring direct bottom-line benefits, for example:

### On the Internet

- A bank cements relationships with commercial customers by offering increased efficiency with online currency trading. This service requires real-time updates and links to back-office transactional and profitability analysis systems.

### Via Extranets

- A bank and an airline both increase their customer bases with a joint venture: a credit card that offers frequent flyer points sponsored by the bank. This service requires joint data-sharing, such as purchase payment and charge-back information, as well as decision support applications to retrieve, manipulate, and store information across enterprise boundaries. Additionally, employees from both companies must be able to access information.

### Using an Intranet

- A global manufacturer accelerates the organizational learning curve by creating a global knowledge-sharing system for manufacturing research and development. Plant engineers on one continent can instantly share process breakthroughs with colleagues thousands of miles away.

## Network Applications Increase Risks

These new network applications may sometimes have a downside. They can open a direct pipeline to the enterprise's most valuable information assets, presenting a tempting target for fraud, malicious hackers, and industrial espionage.

Appropriate protections are a prerequisite for doing business, both for an organization's credibility with its stakeholders and its financial viability. For example:

### Integrity

- The bank offering currency trading needs to protect the integrity of its core systems from unauthorized transfers or tampering.

### Access

- The bank and airline involved in a joint venture might compete in other areas or through other partnerships. A secure barrier, permitting only authorized transactions, must be erected between the two enterprise computing environments.

### Authorization

- The manufacturer posting proprietary discoveries needs to ensure that competitors or their contractors cannot tap into the system. Attacks and unauthorized access from both the outside and inside must be blocked.

## Information Security Goals: Enable Use, Prevent Intrusion

To secure information assets, organizations must provide availability to legitimate users while barring unauthorized access. In general, there are four key goals for security:

### Confidentiality

- Safeguard user privacy and prevent the theft of information both stored and in transit.

### Integrity

- Ensure that electronic transactions are not modified anywhere along the way, either accidentally or maliciously.

### Accountability

- Detect attacks in progress; trace any damage caused by successful attacks. Prevent system users from later denying completed transactions.

### Availability

- Ensure uninterrupted service to authorized users. Service interruptions can either be accidental or maliciously caused by denial-of-service attacks.

Secure systems must provide support for at least these four key areas. To achieve this, information security must be an integral part of system design. Needles to say, OrbixSecurity has been designed from the ground up to excel in these critical areas.

## Risk Management Holds the Key

A large middle ground exists between the extremes of avoiding network applications altogether, fatalistically launching unprotected systems, or burdening every application with prohibitively costly and user-unfriendly security measures.

This middle ground is the area of risk management. The risk-management approach aims not to eliminate risk but to control it. Risk management is a rigorous balancing process of determining how much and what kind of security to incorporate in light of business needs and acceptable levels of risk. It unlocks the profit potential of expanded network connectivity by enabling legitimate use while blocking unauthorized access. The goal is to protect adequately to meet business needs without undue risk, making the right trade-offs between security and cost, performance and functionality.

Consider four different network users: an Internet Service Provider (ISP), a hospital administrator, a banker, and a military officer. Each has a different security concern.

### The ISP

- The ISP is concerned primarily with availability: making services available to its customers.

### The Hospital Administrator

- The hospital administrator wants to ensure data integrity: to ensure that only authorized staff can update patient records.

### The Banker

- The banker is most concerned about accountability: that the person who authorizes a financial transaction is identified and tracked.

### The Military Officer

- The military officer wants confidentiality: to keep military secrets out of the hands of potential enemies.

The challenge is to implement security in a way that meets business needs cost-effectively, both in the short term and as enterprise needs expand. This requires a collaborative effort between corporate strategists and information technology managers. Understanding the business drivers for information security helps clarify where to focus security measures. Understanding the underlying application architecture — how components work together — clarifies the most practical approach for building system security. Distributed applications in particular require new ways of thinking. OrbixSecurity provides an enabling platform that both facilitates these new ways of thinking and provides practical development environment to bring them to market.

# Part 2: Distributed Systems, Distributed Security, Enterprise Control

## Distributed Objects Mean Rapid Development

Object technology, which closely groups data and the business logic that makes use of the data, is having a dramatic impact on the business computing landscape. Developments in the field of distributed object computing allow cooperating objects to reside on different machines, networks or even enterprises. This enables businesses to enhance and reuse installed applications rapidly, representing new power to tap the immense value of legacy resources. As a result, many organizations are migrating from traditional single layer client-server applications to multi-tiered application architectures.

Distributed object technology provides the foundation for next-generation network applications because it offers so much versatility. Distributed object components that encapsulate code and data can reside anywhere on the network. Client software need only know about the object's interface and how the object is implemented; the location where it is running is transparent to the invoking application. Transparency and reusability give distributed object computing environments great power, but they present new challenges for information security. These challenges require new ways of thinking and new tools.

## Security Challenges In Distributed Object Environments

Traditionally, computer security practice has emphasized the centralization (and to some degree, isolation) of sensitive data. In a distributed computing environment, this approach must be modified because security functionality is distributed through the application architecture.

Because distributed object environments are frequently heterogeneous, security may be enforced by the application objects themselves, by middleware, by operating systems, by hardware or by some combination of these. Because many new distributed objects must interact with existing hardware and application logic "legacy systems", security management and facilities must also interoperate with these systems.

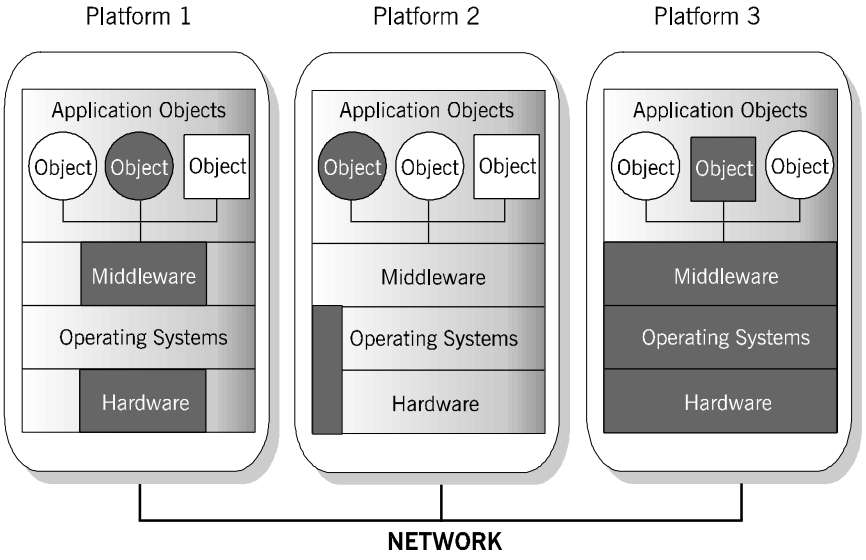



**Figure 1 - Distributed Environment, Distributed Security**

In contrast to more rigid but easily analyzed centralized security designs, distributed object systems have the security architecture shown in Figure 1. Security functionality (the darkest, shaded areas of the diagram) in object systems is distributed throughout the system.

Some platforms and applications may contain a great deal of code that can be trusted to enforce an organization's security policy, while others may have very little.

Distributing security in this manner means that a particular distributed application may be secure, but that fact is hard to confirm. As the number of platforms and diversity of access increase, the difficulty of analyzing security expands exponentially. That is a concern, because if a system cannot be readily analyzed, it is not at all certain that valuable data is being protected.

Furthermore, all the flexibility and openness of distributed object systems make security administration a real challenge. Systems managers with experience administering security in UNIX or Windows NT environments know how difficult it is to get it right. Many security attacks on these systems are not due to obscure security vulnerabilities but to inadvertent administrative errors, or ‘leaving the back door open. ”

Several other characteristics of distributed object systems also complicate security enforcement. The systems are:

### Layered

- Systems consist of many security layers (applications, middleware, operating system, hardware, and network) that must fit together.

### Exposed

- Many distributed object systems are designed to work over the Internet or large intranets. Data going over networks is subject to packet-sniffing interception.

### Dynamic

- Object systems are designed to be dynamic, allowing new application objects to be created on the fly. Objects can play both client and server roles, and can interact in multiple and unpredictable ways. This means that security policies must also be dynamic, adding complexity.

### Multi-enterprise

- Distributed object computing allows the sharing of information among enterprises. Enterprise security policies will be different (say, between a hospital and a bank), which means that data sharing requires translation between enterprise policies.

Configuring and administering security for distributed object systems is potentially far more complex than for a traditional system. Without special tools, security has to be administered manually for each layer independently, leaving room for mistakes and inconsistencies. For instance, an application may correctly confirm that a loan officer is authorized to access a record before allowing changes. However if supporting operating system calls have not been set up with complementary file permissions, access protection is not complete.

The challenge is to create an environment in which the complexity is minimized, ensuring that security administration is enforced automatically and consistently.

## The Solution: Enterprise Services

Software that implements common standards within a manageable structure is the key. This paper focuses on CORBA security because of CORBA ŝ broad

acceptance and inherent scalability and because OrbixSecurity implements the strictest CORBA security standards. However, the security issues raised are pertinent regardless of the standards implemented.

CORBA is a set of standards defined by the Object Management Group (OMG), an industry-wide consortium leading object technology vendors and solution providers. CORBA supports the management of distributed objects; commercially available Object Request Brokers that conform to OMG standards are available for all computing platforms.

The OMG has also defined different levels of CORBA security services that interface with ORBs, simplifying the creation and administration of flexible, enterprise-coherent security policies. CORBA security services leverage the fundamental role played by the ORB to provide a central clearing point for security services. By making security pervasive in the architecture rather than needing to be enforced in each layer separately, CORBA security services dramatically reduce the need for developers to focus on low-level details to make applications secure.

The basic ORB security add-on applies the Secure Sockets Layer (SSL) protocol to secure communications between the client and the server. SSL applies fixed security standards for confidentiality and integrity to network communications. However, SSL alone is like a secure pipe. That pipe can be plastic so no one can see in (confidentiality), or steel so no one can see or break in (confidentiality plus integrity), but both ends are open. It requires custom programming to extend control over the security-sensitive operational environment in which communication takes place.

Higher-level OMG CORBA security specifications, when implemented in CORBA-compliant security service software, make systems less error-prone and also address other security challenges:

- Dynamically managing access to resources based on an enterprise's security policy, the type and content of a transaction, and the credentials of the user.
- Preventing authorized users of the system from gaining access to information that should be hidden from them.
- Stopping internal or external hackers from masquerading as someone else to obtain access to whatever that user was authorized to do or see. (In this scenario, security measures also must prevent actions and damage from being attributed to the wrong person.)
- Preventing internal or external hackers from bypassing security controls altogether.
- Tracing security breaches. This requires, among other capabilities, adequate

identification of users.

Specifically, OMG defines two levels of ORB security, Level 1 (security-unaware applications) and Level 2 (security-aware applications).

## CORBA Level 1 — Security-Unaware Applications

At Level 1, the ORB enforces basic audit and access control functions in security-unaware applications, which are applications that possess no knowledge of the security functions applied to them.

Here, the system administrator defines the security context of network communications. Take for example the operation "get balance," invoked on the object "account." With SSL alone, a fixed protocol for securing confidentiality and integrity is applied to the communication itself. With ORB-level security, the system administrator can evaluate the operation being invoked and place constraints on how "get balance" is invoked, by whom, under what circumstances, and so on. The administrator sets the standards for what type of security, such as confidentiality, will apply to the communication. The application itself is security-unaware; it has zero knowledge of the security functions applied to it, and little interest in managing its own security-enabled conversations.

Security-unaware applications are most appropriate when the security responses are not dependent on application-specific data.

## CORBA Level 2 — Security-Aware Applications

Security-aware applications are more powerful but also more complex. These are endowed with the intelligence and authority to manage the security functions that apply to them.

CORBA Level 2 includes applications programming interfaces (APIs) that enable security-aware applications to manage their own security. This level is particularly important in enterprise environments where fine-grained, scalable approaches are needed.

Within the structure of an overall enterprise policy, security-aware applications can possess intelligence about the communication and about the reason communication is being initiated. The applications can independently manage the user authentication, access control, confidentiality and integrity protection, and auditing of requests, for the services and data they provide. At the same time, the application remains easy to analyze and security administrators retain all the control they need to manage a secure environment.

Security-aware applications are useful when value judgments must be made about the security context of an invocation. For example, the application might need to recognize the difference in a medical record between a patient's blood type and AIDS status, and apply content-appropriate access controls.

## CORBA Security Model

At a high level, the figure below illustrates the CORBA security model describing how and where a secure system enforces security policies for both security-aware and security-unaware applications. In this model, all object invocations are mediated by appropriate security functions to enforce policies such as access controls. The goal is for policies to be tamper-proof, always invoked when required by security policy and to function correctly.

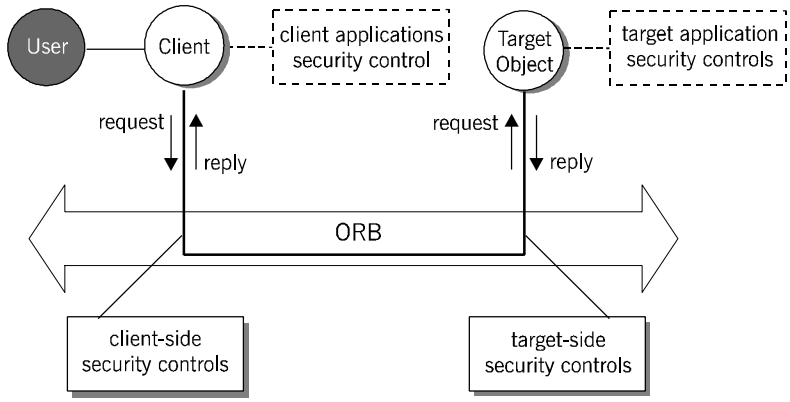



**Figure 2 - Basic CORBA Security Model**

How well a given implementation of the model ensures information security depends on how the following capabilities are implemented:

- Identification and authentication
- Authorization and access control
- Security of communication
- Security auditing
- Administration of security information

Of critical interest to the managers of enterprise-scale systems are features supporting scalability and ease of use for the key stakeholders.

## Scalability and Scalability Goals

Security is needed for a range of systems, from small, local systems to large intra- and inter-enterprise ones. Often, an initial small-scale installation, for instance a corporate marketing site on the Internet, will be the pioneering step for larger-scale systems, such as the addition of interactive services across multiple lines of business to the same site. Concurrently, a net service may be launched to a small audience, with rapid expansion planned if it is well received. Scalability of the security services ensures that systems such as these can grow without either opening security vulnerabilities or needing extensive re-administration of security. To facilitate scalability, systems should incorporate the following:

### Privilege Attribute Roles

- Base access controls on the privilege attributes of users by roles or groups rather than by individual identities. This reduces administrative costs.

### Security Domains

- Have a number of security domains, that enforce different security policy details but support internetworking between them, subject to policy.

### Key Management

- Manage the distribution of cryptographic keys across large networks securely and without undue administrative overheads.

## Diversity and Diversity Goals

Security services must also be usable by a diverse audience. The OMG goals for ensuring this usability include the following:

### Simplified Administration

- Administrators should be able to understand and manage complex security systems using a simple, clear-cut interface. They should not have to specify controls for individual objects or individual users of an object (except where needed by security policy). The system should provide flexibility and granularity.

### Unitary Login

- End users should only need to log on to the distributed system once to access applications and other IT services. Security availability should be transparent.

### Facilitate Development

- Developers should not need to be aware of security for their applications to be protected. However, a developer who understands security should be able to protect application-specific actions within the scope of enterprise security policy.

## Scalability and Scalable Solutions — Using Roles to Group Users and Principal Access

An important ability when managing enterprise systems with thousands, tens of thousands, or in the case of Internet applications, potentially millions of users, is the setting of access rights based on roles rather than individual identity. For instance, users identified as Internet customers might be granted one level of access, users identified as administrative staff another, and senior management yet another. Preferred customers might have more rights than standard customers, or administrative staff in Human Resources might have access to salary files across the division, while middle managers in Marketing might not.

Role-based access is implemented via the attributes assigned to principals. A principal is defined as a human user or system entity that is registered in and authentic to the system. Initiating principals are those that initiate activities. An initiating principal may be authenticated in a number of ways; for human principals the most common authentication method is a password. For system entities, authentication information such as a long-term key must be associated with the object.

### Principal Attributes

An initiating principal has at least one and possibly several identities. These are represented in the system by attributes, which may be used to:

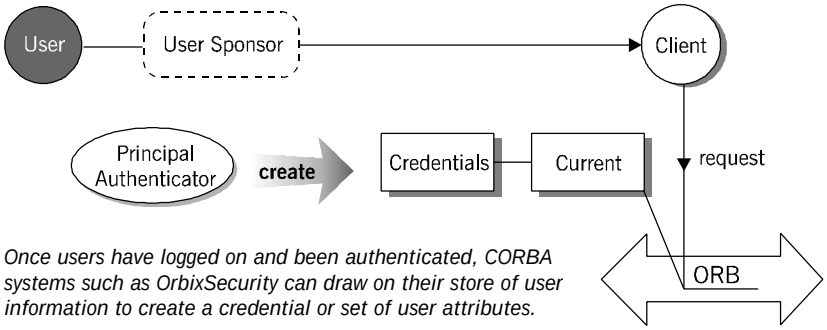
- Make the principal accountable for its actions.
- Obtain access to protected objects (though other privilege attributes of a principal may also be required for access control).
- Identify the originator of a message.
- Identify whom to charge for use of the system.

When a user or other principal is authenticated, it normally supplies:

- Its security name.
- The authentication information needed by the particular authentication method used.
- Requested privilege attributes (though the principal may change these later).

The principal may have privilege attributes that can be used to decide what it can access. A variety of privilege attributes might be available depending on access policies. The privilege attributes that a principal is permitted to take are known to the system. At any one time, the principal may be using only a subset of these permitted attributes, either chosen by the principal (or an application running on its behalf) or by using a default set specified for the principal. There may be limits on the duration for which these privilege attributes are valid and controls on where and when they can be used.

In large object systems, using individual identities for access control is often impractical, as many sets of control attributes need to be changed when a user joins or leaves the organization or changes jobs. Fortunately, individuals can be assigned to groups, and attributes of the group used for security controls. Roles, or job function, are a common means of grouping privilege attributes. Examples of roles and other grouping constructs might include job titles, membership in a project team, management level or placement in a geographic location.




*Once users have logged on and been authenticated, CORBA systems such as OrbixSecurity can draw on their store of user information to create a credential or set of user attributes.*

**Figure 3 - Authentication, Credentials, and Attributes**

## Grouping Objects: Security Policy Domains

The OMG security policy defines a domain as a set of objects that a security authority administers. Within the domain, a set of security activities applies to all domain members (the objects). Domains can be large, with thousands of objects, or small, with just a handful.

The policy represents the rules and criteria that constrain activities of the objects to make the domain secure. The policy concerns access control, authentication, secure object invocation, delegation, and accountability. Access to the policy control is itself controlled, limiting who may administer security-relevant policy information.

In an object system, the cost of using the security mechanism at the individual object level in all environments would often be prohibitive and unnecessary. Preventing objects from interfering with each other might require them to execute in separate systems processes or virtual machines, but in most object systems, this would be considered an unacceptable overhead, if applied to each object.

## Authentication Overhead

Authenticating every object individually could also impose too large an overhead where:

- There is a large object population.
- There is high connectivity and therefore a large number of secure associations.
- The object population is volatile, requiring objects to be frequently introduced to the security services.

Security policy domains permit application of security policies to security-unaware objects without requiring changes to their interfaces, by associating the security policy management interfaces with the domain rather than with the objects to which policy is applied.

Domains also provide a mechanism for delimiting the scope of administrators ' authorities.

Examples of domains might include the set of objects related to Eastern operations, the set of objects containing sensitive personnel data, or the set of objects containing proprietary manufacturing data.

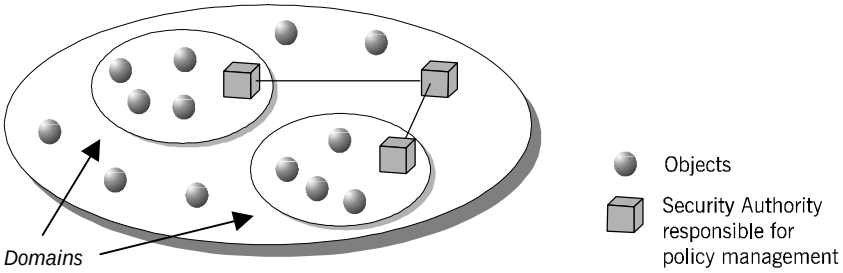



**Figure 4 - Security Policy Domains**

## Grouping Operations — Required Rights

An object system can have many objects, each of which might have many operations, so it may not be practical to associate control attributes with each operation on each object. Doing so would impose too large an overhead on the administration of the system.

Analogous to role-based attributes for principals, required rights provide a means of grouping operations. Rights correspond to an action that a client is allowed to perform on a target. Required rights are a way of collecting common actions together. Defining rights in such a way that the administrator only needs to know what rights are required, rather than the semantics of particular operations, simplifies administration without compromising security.

A common case of grouping types of actions together is the implementation of rights in the UNIX operating system, with its familiar categories of "read," "write," and "execute." Unlike an operating system, CORBA, deals with a broad category of rapidly changing objects; therefore the standard rights in CORBA are different. CORBA rights map onto standard business practices and operations.

| Right | Meaning |
|---|---|
| get | Used for any operation on the object that does not change its status. |
| set | Used for operations on an object that change its state. |
| manage | Used for operations on the attributes of the object, not its state. |
| use | Used for operations on an object that may change the overall state of the system, but not the state of the object itself. |

## CORBA Rights

For example, consider an object that contains a customer record. The following table illustrates how the required rights for this object's associated operations might be assigned.

| Operation | Required rights |
|---|---|
| View transaction histories | get |
| View customer information (such as address, phone number, and so on.) | get |
| Update account status | manage |
| Update the record with a new transaction | set |
| Make a copy of the record | use |

## Enterprise CORBA Security Model

CORBA security services enable administrators of large and complex systems to manage more effectively by grouping the security policies of the crucial variables that control how a client may interact with objects in a distributed environment.

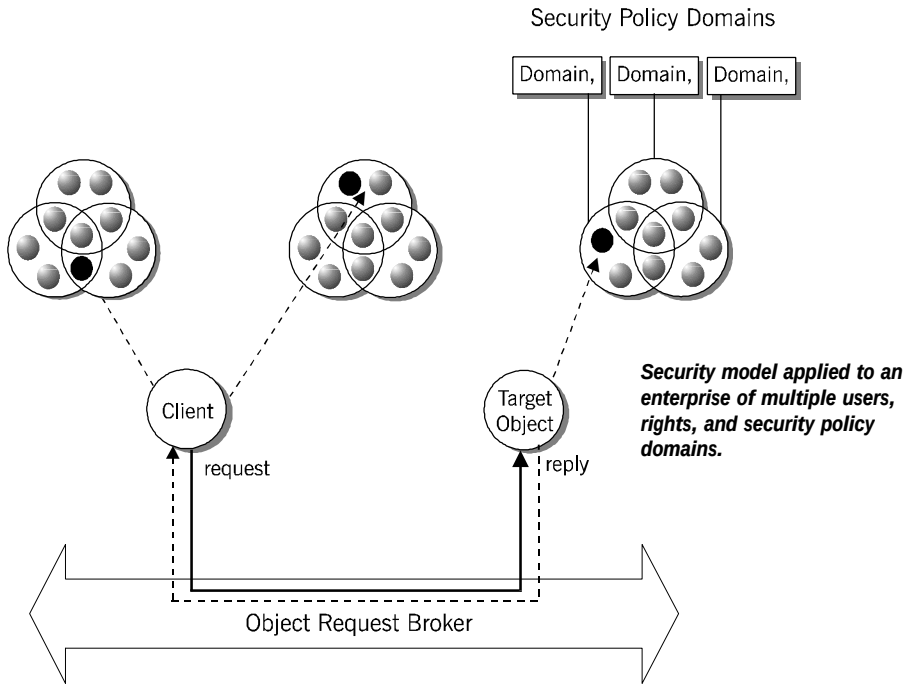



**Figure 5 - Security Policy Domains**

Each client corresponds to a principal (human or active system entity), and is mapped to a set of privilege attributes (for example, user id, group, or clearance). The client's request contains an interface or operation that maps to a set of required rights (for example, get, set, manage or use). Target objects are grouped into security policy domains, which define invocation access, invocation security, audit, delegation, and application-specific security policies.

An invocation access policy, for example, maps principals 'privilege attributes to the rights that principals may use to invoke operations on objects within the domain. For instance, accounts payable administrators (privilege) might be the only ones allowed to change account status (a manage right) within a corporate division (a domain).

Privilege attributes, required rights, and domains are all combined to fine-tune and streamline security administration. Adding new capabilities to distributed

systems quickly and securely becomes dramatically easier. When new users, new options for object actions, or even completely new objects are added, existing security measures can be automatically enforced.

The nearly boundless flexibility of distributed object computing poses challenges for the security designer. However, CORBA security makes these challenges manageable. Moreover, the speed with which distributed object systems can be developed, adapted and enhanced for new market demands is the key to significant competitive advantages.

# Part 3: OrbixSecurity — An Enterprise Focus

IONA Technologies supplies OrbixSecurity, a CORBA security services software product that exceeds the OMG Security Services Specification for Secure Functionality, Level 2 Conformance. These services include:

- Comprehensive, underlying security services for all CORBA applications.
- Association of privileges with users through the use of credentials.
- Delegation of credentials to allow objects to act on behalf of users.
- Access control to hosts and objects based on credentials.
- Integrity and confidentiality protection for communications between entities.
- Auditing of authentication, communication establishment, and object invocations .

For security-aware applications, IONA ś OrbixSecurity provides:

- Application-level access control.
- Auditing of application-related events.
- Ability to constrain delegation and restrict privileges.
- A standard security administration interface.
- OMG-compliant interfaces allowing ORB vendors to develop security administration graphical user interfaces consistent with their existing product lines.

OrbixSecurity also includes additional capabilities above and beyond the CORBA Level 2 security service specification in the areas of hierarchical domains, unitary login, audit and cryptographic protection.

OrbixSecurity is designed to support scalability and to simplify security management in enterprise-level systems.

## Hierarchical Domain Trees

The key to this scalability and simplified management is the unique way in which OrbixSecurity manages domains. In addition to supporting the multiple domains required for CORBA compliance, it is possible to implement a unique hierarchy of domains that allows the implementation of even the most complex policies for large sets of objects.

Domains can be linked in a conceptual tree of branching policies, where security policy information can be assigned to critical nodes. Operators can be assigned to each of the nodes detailing how to compose the security policies of the objects below. The lower-level objects 'policies are inherited from those objects above them. This avoids the arduous and error-prone task of repeating all the policy information for every object in the system.

Rules of composition guide how to traverse the tree. When an object is part of a domain, the administration then navigates the tree to its root, composing the domain information along the way. Managing a large-scale transactional system becomes simplified, for instance, as tens of thousands of objects can be put into a few dozen domains, which then can be arranged into a reasonable hierarchy with policy information at strategic points in the tree.

## Policy Authority: Rules to Streamline Enterprise System Management

Conceptually, the administrative services in OrbixSecurity serve as a policy authority that coordinates the policies within a large enterprise. The policy authority has three rule sets that define enterprise-wide policy:

- *Principal authentication rules* define a standard way to map principals to security attributes (for example, identifying attributes that are within certificates from a recognized certification authority).
- *Object interface rules* provide a consistent approach for defining families of access rights across different application interfaces.
- *Domain composition rules* define a standard way to compose security policy domains, so that domain hierarchies can be supported.

Finally, federated policy rules are planned for future releases. Federated policy rules will govern communication among enterprises. They will define an approach for mapping privilege attributes from one policy authority to another, so that data may be shared between enterprises in a controlled fashion. The federated policy rules will provide a common means for enterprises to communicate the trustworthiness of their principals. They will translate the credentials of a foreign principal into privilege attributes of a local principal; local security policies then use these translated attributes to enforce local policy. A user from one enterprise, for example, could be treated as a special guest of a federated enterprise and allowed access only to selected object resources.

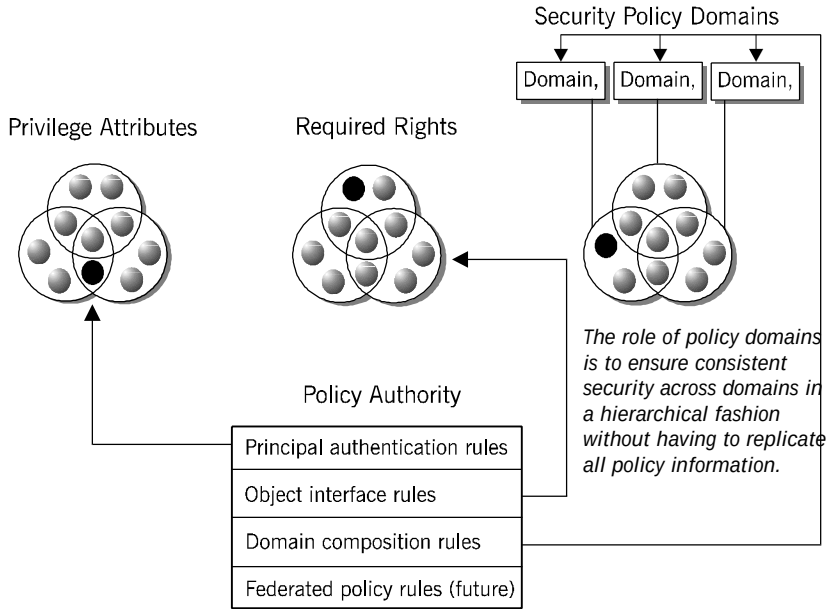



**Figure 6 – The Policy Authority**

## Services Overview

This section provides an overview of the core functions of OrbixSecurity. The below diagram illustrates how these components operate within the OrbixSecurity toolkit architecture available for sophisticated leading-edge secure enterprise application integration.

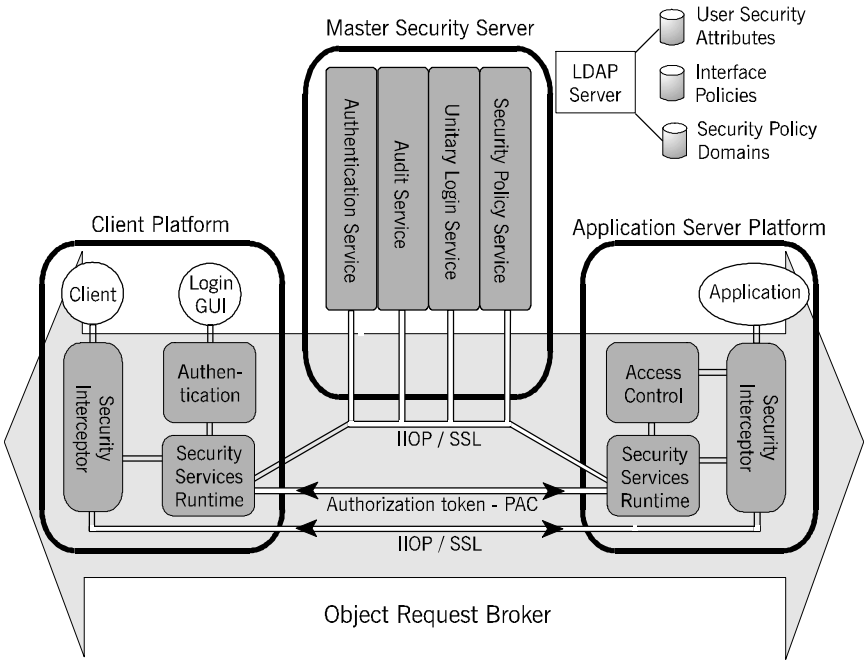



**Figure 7 - OrbixSecurity Toolkit Architecture**

## Identification and Authentication

To protect critical information, an information system must know who is attempting to access its information. Users identify themselves to information systems by providing a login identity, for example, an identity known to the information system, and some data that proves their authenticity.

Authentication data could be a static password, a one-time password or a digitally signed challenge. The strength of the authentication needed for an environment is based on such factors as the creditability of the information being protected and the threats within the environment. Anonymous users may be allowed to access data or resources not requiring protection. However, some form of identification and authentication must be performed before decisions can be made regarding access to protected information.

OrbixSecurity provides selectable authentication mechanisms for users to log in to CORBA environments. Three different authentication mechanisms are supported:

- A user ID and password login.
- A login using Security Dynamics 'SecurID.

- A public key-based authentication scheme.

## User ID and Password

User ID and password-based authentication schemes are currently the most pervasive authentication schemes in use and are supported by OrbixSecurity for three reasons:

- They are easy to administer.
- Their use does not require licensing of particular products.
- They do not require the purchase of additional hardware systems or tokens.

User ID and password authentication is particularly suited for closed environments, such as Intranets, where a known group of relatively trustworthy users have access to the information systems.

## Security Dynamics' SecurID

Security Dynamics' SecurID generates a new, unpredictable access code every 60 seconds, providing two-factor, token-based authentication. The SecurID token works in conjunction with Security Dynamics 'hardware or access control modules (ACMs), including ACE/Server®. SecurID technology offers strong authentication for a wide range of platforms in one easy-to-use package. OrbixSecurity integrates the ACE/Server® into distributed CORBA environments and provides an interface that allows users to enter the access code displayed on their SecurID token.

## Public Key-Based Authentication

Public key-based authentication schemes are based on the concept of challenge-response. The authenticating entity (for example, the client) is sent a challenge by the party with whom it wishes to communicate (for example, the target). The client signs the challenge using its private key and returns the signature. The target validates the signature using the client's public key, which the client sends to the target or the target obtains from a central directory. OrbixSecurity provides two-way public-key based authentication by:

- Authenticating clients to targets so that the target can be assured it knows from whom a request is coming.
- Authenticating targets to clients so that clients can be assured they are communicating with their intended target.

## Authorization and Access Control

Within a secure computer system, there must be a reliable way to check whether a user, or object acting on behalf of a user, is permitted to access resources within the system. Authorization and access controls provide the mechanisms necessary to restrict access to resources to prevent their unauthorized use. These mechanisms validate that a user, or object acting on behalf of a user, has the appropriate privileges before permitting that user or object to access protected resources.

After users log in to the environment, OrbixSecurity associates privileges with a user through credentials. Credentials are objects associated with a user's current thread of execution that specify, in a secure manner, the privileges associated with that user. These privileges are checked before object invocations are allowed.

OrbixSecurity provides access control at two levels of granularity:

- At the host level, a coarse level of granularity.

  Controlling access at the host level means an administrator can grant or deny a user (or group of users) access to a host, or a set of hosts, within the OrbixSecurity environment. The user is then allowed or denied the ability to invoke operations on that host, subject to any restrictions imposed at the operating system or application levels.

- At the object-interface level, a much finer granularity of control.

  For a finer grained, more flexible authorization scheme, OrbixSecurity supports access at the object-interface level. Determining whether to allow a user to invoke on the object's interface is based on the user's privileges, the control attributes associated with the object or resource, and the policies governing that object.

Regardless of the level of granularity, the access policy is enforced for all applications, whether they are aware of security or not. In addition, a security-aware application can make use of OrbixSecurity's access control mechanisms and perform access control within the application based on its own set of rules and policies. Application access control polices are enforced by the client or target application, as opposed to being enforced by the ORB.

## Delegation of Privileges

Within an object environment, a client may call on an object to perform a particular operation. However, instead of completing the operation itself, the object will call on another object to complete the task. To allow intermediate objects to perform tasks on behalf of initiating entities, the privileges of the

initiators must be delegated to intermediate objects or to chains of intermediate objects.

OrbixSecurity supports delegation of privileges to allow an object to act on behalf of a client. Administrators can also prohibit delegation from occurring. This latter feature is useful for preventing objects with wide-reaching privileges from delegating privileges to less-than-trustworthy objects.

OrbixSecurity supports simple delegation. In this delegation scheme, the client permits the intermediate to assume its privileges, both using them for access control decisions and delegating them to others. The target object receives only the client's privileges, and does not know who the intermediate is.

Applications do not need to be aware of the delegation policy in use within the environment to benefit. However, a security-aware application can override the default delegation policy and specify its own delegation rules.

## Distributed Access Control

IONA Technologies is developing Privileged Attribute Certificates (PACs) using X.500 standards to solve the problem of communicating user credentials from one point to another, particularly when the communications may be between ORBs from different vendors. In the PAC paradigm, identity and authorization are separated. Identity is passed around via an identity certification at the Secure Sockets Layer (SSL) level, and authorization is handled a level above that.

OrbixSecurity maintains the mapping between the authorization and the identity. This approach offers greater flexibility for administrators needing to integrate with systems that do not handle SSL, for instance a transaction processor or third-party product. With the PAC support on all affected ORBs, identity can be reconfigured without having to redo all attributes associated with authorization.

## Unitary Login

A critical requirement in distributed systems is the need for a unitary login capability that allows users to log in to an environment only once and be granted access to all systems and applications for which they are authorized. This includes legacy applications and legacy systems that cannot be fully integrated into a CORBA environment.

Without a unitary login capability, users often need to remember a multitude of user IDs and passwords and to enter them continuously whenever they move from application to application or across systems.

OrbixSecurity provides a unitary login capability based on interpreting security-aware wrappers that developers can create for every enterprise application. These wrappers obtain the login information needed to log users into commercial applications and legacy systems. This login information is encrypted to protect it from unauthorized disclosure or misuse.

## Security of Communication

In distributed environments, object interaction may span systems or networks. Before critical information can be exchanged and protected functions invoked, trust must be established between communicating clients and target objects. Authentication is used to establish that trust. Clients authenticate themselves to targets so targets know from whom a request initiated, and targets authenticate themselves to clients so that clients can be assured they are communicating with their intended targets.

Security of communication also encompasses the protection of data against unauthorized modification and disclosure as it travels between objects or across networks. Integrity protection ensures that data is not modified between the time it leaves one object and the time it arrives at another. Integrity mechanisms protect requests initiated by the client as well as responses generated by the target.

Confidentiality protection provides security of communication through encryption. Confidential communications are established using encryption keys known only to the communicating entities. Confidentiality mechanisms protect information, both requests and responses, from unauthorized disclosure as well as from unauthorized modification.

OrbixSecurity provides extensive features for establishing security of communication. The underlying protocol supported by the product SSL.This provides:

- Mutual authentication between communicating entities.
- Integrity protection of data in transit.
- Message data encryption.

## Security Auditing

The security functions just described are designed to protect critical information in computer systems from unauthorized disclosure or modification. Security auditing logs access attempts to protected or sensitive resources.

Auditing makes it possible for authorized personnel to monitor two key characteristics of a functioning information system:

- The actions of users attempting to access the system and its resources.
- The reactions of the information system to those attempted accesses.

Auditing ensures that users and the implemented security features of the system are behaving appropriately.

OrbixSecurity supports auditing of security-relevant events that occur at the ORB-level and within OrbixSecurity itself. An audit decision object is used to determine which of a set of events is to be audited within the OrbixSecurity environment or domain. This is enabled via the system administration interfaces provided with OrbixSecurity.

### Audit Logs

To facilitate correlation of events across distributed systems, OrbixSecurity can be configured to record all audit logs in a single persistent data store. Audit logs are recorded in a standard, specified format so they can be easily imported into a database for querying and report generation. Audit logs are integrity protected to prevent removal, insertion, and modification of audit data. The integrity protection mechanism renders all changes to audit logs detectable. OrbixSecurity provides an alarm feature to alert appropriate personnel if tampering occurs or if suspicious activity is detected.

Security-aware applications can make use of OrbixSecurity's auditing mechanisms to audit application-level events. Applications can define their own policies and record security-relevant application-level events in the central, or in their own, integrity-protected audit log.

## Administration of Security Information

Authorized users of information systems and the privileges associated with those users must be established using administration functions or interfaces provided by the system. Security information must be carefully administered so that:

- Users are granted privileges for only those functions necessary to perform their jobs.
- Access rights associated with objects map appropriately to assigned privileges.

Within any CORBA environment, privileges and access policies based on those privileges can become very complex. Mechanisms are needed to simplify administration of security policies across the distributed environment.

The administration section of the OMG Security Service Specification includes a framework for administering security policies and provides details for administering particular types of policies. These include three main points:

- Default protection of information traversing between communicating entities, at enterprise-defined quality level thresholds.
- Delegating credentials.
- Events to be audited.

OrbixSecurity implements the security administration interfaces required by the OMG Security Services Specification for Security Functionality, Level 2 Conformance. These interfaces are used by graphical user interface (GUI) developers to build GUIs with the same look and feel as other user interfaces provided with their ORB product. This allows ORB vendors to provide a consistent interface to all of their packaged components.

Additionally, OrbixSecurity contains a number of administration interfaces for creating, modifying, and deleting security-relevant persistent information. These interfaces are used to administer user account information, including authentication information and privilege data.

## Using OrbixSecurity

The security technology that implements OrbixSecurity has also been integrated with other enterprise ORBs, including Hitachiÿs TPBroker and Inpriseÿs VisiBroker. This security technology offers a consistent interface and is fully integrated with the respective ORB services, simplifying the learning curve for users already familiar with a particular ORB interface. The resulting secure ORB product can be used by:

- System integrators can extend the basic security services by integrating a new encryption algorithm or specifying new events to be audited.
- System integrators and application developers can make calls to security services from within security-aware applications to add security at the application layer or to enhance the quality of protection provided at the default level.
- A security-unaware application can sit on top of the security services provided by the secure ORB product, knowing its data is being protected by the underlying security services completely transparent to the application.

- System administrators can use the system administration interfaces to configure the system in accordance with its enterprise information security policy. The system administration interfaces are accessible through GUIs provided by ORB vendors.
- End users can interface with the security services through client applications that make use of, or simply benefit from, the underlying security services.

Orbix makes software work together. OrbixSecurity makes software work together securely.

# Glossary

**Asynchronous Communication**
Asynchronous data communication describes a process where an exchange of communication continues without the requirement for any indication of whether an individual communication was successful or not. Asynchronous communication is most commonly used in large systems where speed and reduction of bandwidth congestion are more important than guarantee semantics.

**Authentication**
Used to establish that users are who they claim to be. Identities must be verified using one or more of passwords, security tokens, and/or biometric or physical devices.

**Authorization Control**
This process must be invoked to check user privileges granting them access to a given resource. This is also referred to as access control.

**Access control**
This is also referred to as authorization control.

**Biometric Device**
A device that authenticates users based on some unique physical, biological, or behavioural characteristic or set of characteristics. Examples include voiceprint, fingerprint, typing rhythm pattern, or retinal scan.

**Boolean**
A search method using the Boolean logical parameters of "and", "or" or "not" as in, one looks for blue and red; blue or red; blue not red.

**Client**

In CORBA, a program that requests services from a CORBA object.

**Confidentiality**
This is the safeguarding of a system's applications and/or data against unauthorized disclosure.

**CORBA**

Common Object Request Broker Architecture. A standard architecture, defined by the OMG, for communications between distributed objects. An ORB is the core element of the wider OMG framework for developing and deploying distributed components.

**Countermeasure**
This is a device, process, or software object that inhibits a specific vulnerability or class of vulnerabilities from compromising the integrity of a system.

**Decoupled Communication**
Clients and objects communicate through a proxy, thereby removing the need for one to be aware of the specific instance of another. This communication is generally found in Publish-Subscribe technology where the object publishing data is indifferent to whatever object is subscribing to it.

**Digital Signature**
A method of authenticating the source and content of digital data or applications. Digital signatures identify any forging or unauthorized alteration.

**Encryption**
This processes data into a jumbled format unreadable without properly authorized decryption information.

**Exploit**
Unauthorized internal or external parties may exploit system vulnerabilities to gain unauthorized access.

**Extranet**

An intranet that is extended to strategic users outside a company, such as partners or suppliers.

**Firewall**
A device or collection of devices that regulate the exchange of information between two networks according to a specified security policy.

**Integrity**
This is the process of making certain that data is received unchanged from that originally transmitted.

**Internet**

A global, decentralized computer network. Unlike an intranet, access to the Internet is unlimited.

**Intranet**

A network used to share information in an organization. An intranet is usually accessible only by the organization's members or employees.

**IP Multicast**

IP Multicast is a type of addressing that allows for one to multiple host communication over IP. A good example of its use is in OrbixTalk, the IONA messaging product, which uses UDP over IP Multicast.

**Keys**
These are mutually defined parameter specifications for the encryption and decryption of data.

**Legacy Application**

An existing application that uses languages, platforms, or techniques from an earlier IT era than the present.

**Middleware**

Any software that connects two or more otherwise separate pieces of software.

**Multicast**
Multicast is communication between a single sender and multiple receivers on a network. In the Notification Service, this definition should not be confused with IP Multicast or UDP Multicast, both of which are specific implementations of Multicast technology.

**Non-Repudiation**
Legitimate users must be prevented from later denying responsibility for satisfactory, fulfilled transactions.

**Object**

In object-oriented programming, a single software entity that consists of both data and procedures that manipulate that data. In CORBA, objects can be located anywhere in a network. The functionality of a CORBA object is accessed through interfaces defined in IDL.

**OMG**

Object Management Group. A consortium that aims to define a standard framework for distributed, object-oriented programming. The OMG is responsible for the CORBA specification.

**Operation**

The IDL equivalent of a function or method. Operations are defined in IDL interfaces and can be called on CORBA objects.

**ORB**

Object Request Broker. An ORB is a middleware component that acts as an intermediary between a client and a distributed object. The ORB is responsible for delivering messages between the client and object across a network. The ORB hides the underlying complexity of the distributed system, such as differences in hardware, operating systems, and programming languages, from the application programmer.

**Point-to-Point Communication**
A specific client sends data to a specific object, the existence and location of which is known to the client.

**Secure Fallover**
Enterprise security systems must ensure that, during a system crash or excessive demand or load periods, system security remains intact.

**Security Auditing**
This keeps a record or log of relevant security events and accesses of data and applications within a secure system.

**Security of Communications**
Data should be secure from interception and modification by unauthorized users while in transit across a network. Any exceptions should be noted and flagged as such.

**Server**

A program that provides services to clients. CORBA servers act as containers for CORBA objects, allowing clients to access those objects using IDL interfaces.

**Smartcard**
A token-based device, often used in security systems to enhance password authentication. Usually credit-card sized.

**Synchronous Communication**
Synchronous data communication requires that each end of an exchange of communication responds in turn, without initiating a new communication. As each transmission is received, a response is returned indicating success or the need to resend. Each successive transmission of data requires a response to the previous transmission before a new one can be initiated.

**Token**
Provides identity or authorization confirmation, usually in the form of a dynamically produced alphanumeric string that provides greater assurance than a fixed password string.

**Trojan horse**

This is an application that initially conceals security-breaching functionality beneath an unrelated, apparently innocuous functional exterior.

**Unitary login**
Following login, the user authentication data is disseminated automatically. This minimizes the requirement for subsequent user logins.

**Virus**
A self-propagating piece of code that attaches itself to applications within a system. Without adequate protection, viruses can spread unchecked among and between systems, compromising data integrity and reliability.

**Vulnerability**
This is a weak point or oversight within an secure system that can be exploited to facilitate unauthorized access.

# Further Reading

1. IONA Technologies. *Orbix 3.0 Release Notes*. February 1999[1].

2. Object Management Group (OMG). *The Common Object Request Broker: Architecture and Specification, Revision 2.1*. OMG document number 97-09-01. August 1997[2].

3. Object Management Group (OMG). *CORBAservices: Common Object Services Specification.* OMG document number 98-12-09. March 1995.

4. Baker, Seán. *CORBA Distributed Objects: Using Orbix*. Addison-Wesley, November 1997.

5. Henning, Michi, and Vinoski, Steve. *Advanced CORBA Programming with C++*. Addison-Wesley, February 1999.

6. Geraghty, Ronan, and others. *COM/CORBA Interoperability*. Prentice Hall, January 1999.

7. Slama, Dirk, and others. *Enterprise CORBA*. Prentice Hall, March 1999.

---

[1] Orbix release notes are available from the following location:
`http://www.iona.com/online/support/update/index.html`

[2] OMG documents are available from the following location:
`http://www.omg.org`

# Contact Details

IONA Technologies PLC
The IONA Building
Shelbourne Road
Dublin 4
Ireland
Phone: ..... +353 1 637 2000
Fax: ..... +353 1 637 2888

IONA Technologies Inc.
200 West St
Waltham, MA 02451
USA
Phone: ..... +1 781-902-8000
Fax: ..... +1 781-902-8001

IONA Technologies Japan Ltd.
Aoyama KK Bldg 7/F
2-26-35 Minami Aoyama
Minato-ku, Tokyo
Japan 107-0062
Phone: ..... +813 5771 2161
Fax: ..... +813 5771 2162

Support: ..... support@iona.com
Training: ..... training@iona.com
Orbix Sales: ..... sales@iona.com
IONA's FTP site ..... ftp.iona.com

**World Wide Web:** http://www.iona.com/