Entegrity® DCE

Secure Co-existence with other Middleware Platforms

Today the emphasis is on providing users and partners access to applications via web technology – but levering existing applications built on a variety of middleware platforms. This white paper describes the various techniques open to DCE systems to securely co-exist and complement other middleware platforms.

Entegrity Solutions® Whitepaper

The information contained in this document is subject to change without notice.

ENTEGRITY Solutions®

Table of Contents

Executive Summary

Many Enterprises use a number of different middleware platforms to support their business applications, many based on DCE. Today the emphasis is on providing users and partners access to applications via web technology, but levering existing applications built on a variety of middleware platforms. Traditionally many of an Enterprise's applications have been disjoint, providing a service to a particular department only. With the on-set of total value-chain processing where disparate systems need to be connected, different applications and middleware platforms need to be either securely connected, and/or co-exist. This white paper is intended to illustrate to DCE users the options open to them using Entegrity products and services that permits their DCE systems to either securely connect to other middleware platforms or for their DCE system to be part of an overall authentication and authorization framework.

Introduction

The white paper is split into 4 main sections, as follows:

- ***Middleware Landscape:*** Provides an overview of the primary middleware technologies in use, and explains some of the trends associated with them. This section also explains some of the business drivers facing Enterprises using DCE.

- ***AssureAccess:*** Entegrity's AssureAccess access management product is briefly introduced, in particular those functions that are important in providing portions of the solutions described later on in the white paper.

- ***Common Authentication and Authorization Platform:*** This section describes how a unified authentication and authorization platform can be supported when DCE applications and 3-Tier Web systems are required to form a homogenous system.

- ***Bridging:*** Describes how DCE can securely interoperate with a variety of different middleware platforms.

The Middleware Landscape

Trends

There are two primary trends in the IT industry right now:

- Many organisations are opening up their applications and systems to Web users. These users are either internal or external users. The common theme is that the user uses a Web Browser to access the application or data. Many traditional back-end products – such as **SAP R/3** or **PeopleSoft** – all now have the ability to gain access to them from Browsers. Suppliers such as these and many so called "pure play" vendors have developed products called Portals that enable such connections to be established. Portals are specialised web servers that support many different types of "Portlets" that communicate with the appropriate back-end system, so for instance most Portals support a SAP R/3 Portlet. Most Portal vendors support toolkits so that it's quite easy to produce a custom Portlet.

- There is tremendous activity to develop and deploy Service-Oriented architectures (SOA) – characterised by "Web Services". The services approach focuses on building the business logic of the application as services. These services can be deployed on the SOA without the service needing to know about the presentation and flow of the application, or how the application will make requests to the service. Web Services provides a platform for building services with the following characteristics:
  - loosely-coupled
  - location transparent

Islands Of Distributed Computing

There are 5 primary types of distributed computing architectures present in the industry today. They are:

3-Tier Web:

This is the "Web model". A Web browser interacts with a web server (or Portal). The web server contains presentational logic and interacts with a backend application server for the business logic. The Application Server is normally in the form of a J2EE Enterprise Java Bean (EJB) container. The web server and application server interact using the Remote Method Interface (RMI) which underpins EJB technology. RMI is a form of RPC.

Distributed Component Object Model (DCOM):

Microsoft's distributed technology is based on DCE RPC. DCOM clients use RPC to communicate with DCOM components. A typical use scenario is for a user on a workstation to interact with a local server, the server then uses DCOM to obtain services from distributed DCOM components. DCOM can be used in either Web or traditional environments.

DCE:

Distributed Computing Environment.

CORBA

Common Object Request Broker Architecture (CORBA) is an architecture and specification for creating, distributing, and managing distributed program objects in a network. It allows programs at different locations, and developed by different vendors to communicate in a network through an "Object Request Broker." (ORB)

Web Services:

Web services are typically made available from a business's Web Server or Portal for Web-connected applications. The underlying protocol used by Web services is Simple Object Access Protocol (SOAP) and means for a program running on one system to communicate with a program in another system by using the World Wide Web's Hypertext Transfer Protocol (HTTP) and XML as the mechanisms for information exchange. SOAP specifies exactly how to encode an HTTP header and an XML file so that a program in one computer can request a service on another computer and pass it information. It also specifies how the called service can return a response. SOAP is a form of RPC – but using HTTP & XMLFigure 1 shows the Islands, except for CORBA – which can be envisaged as an object-oriented version of DCE.

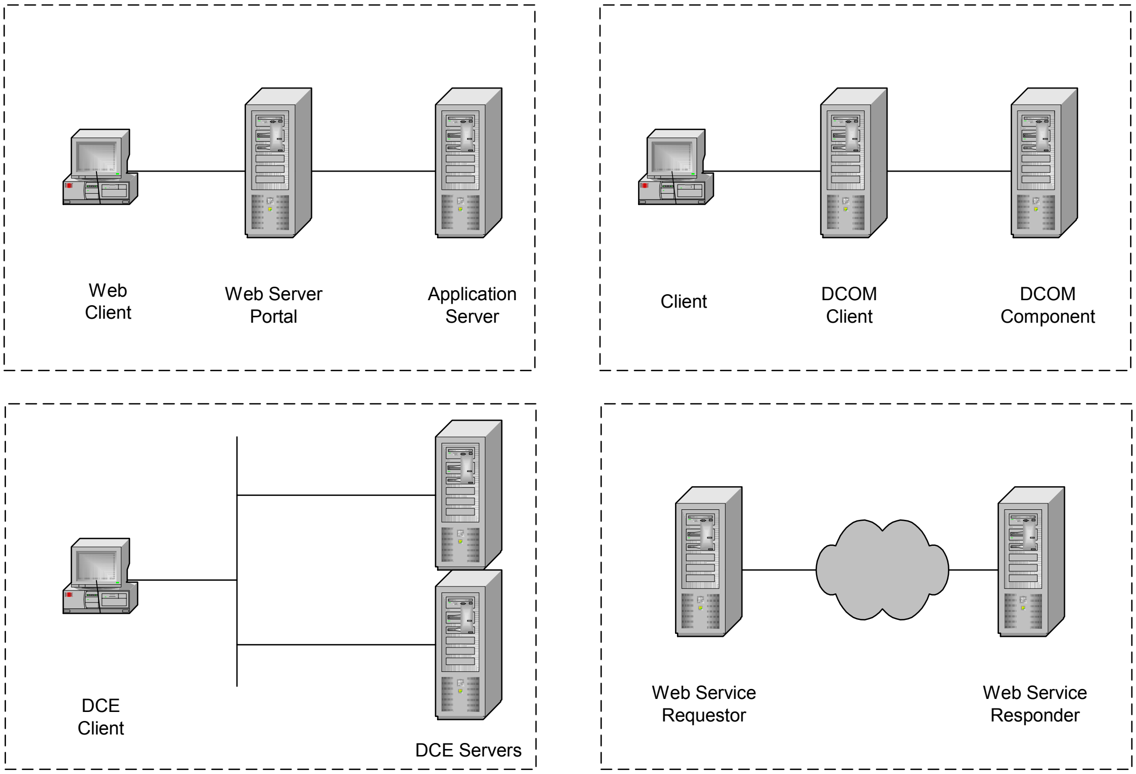


Figure 1: Islands of Distributed Computing

Business Drivers and Requirements

What are the business drivers that "encourage" organizations to look at connecting their DCE applications to other applications:

- Increased Return on Investment (ROI) on DCE. Many organizations have a large investment in the DCE infrastructure, DCE applications and trained administrators. To port or rewrite applications for other middleware technologies will require a large new investment. Many organizations can leverage their existing DCE investment.

- The existing DCE applications have proven value and provide a service to a current set of users. Other users and applications may require those services

In addition to these business drivers it will be usually the case that the set of users using a DCE application and those using other associated non-DCE applications will, to a large degree overlap. This will usually result in administration staff having to maintain two distinct sets of user information and authorization/permission rights. Obviously this significantly increases the load and will result in data entry errors. Saving administration costs so that only one set if user information is maintained and administered is also a significant benefit.

Given that there are 4 other distributed computing environments, other than DCE, there are other possible business requirements for a DCE system:

- A DCE application may want to call upon a web service, or a web service may wish to call a DCE application to obtain some data
- A DCE application may require a Web front end, perhaps directly from a Portal.
- Microsoft DCOM applications may wish to exchange data with DCE applications.
- A DCE application may require a access to a CORBA based application, and vice versa
- All exchanges are secure and appropriate identity/Principal information is exchanged.
- A common user administration system is in place

The following section describes how these business requirements can be satisfied using products and services from Entegrity Solutions.

AssureAccess

At this point, let us introduce the Entegrity AssureAccess product. AssureAccess provides Authentication, Single-Sign On and Authorization facilities within a 3-Tier web environment – primary geared around J2EE environments.

Integration Points

AssureAccess provides a number of ***integration points*** into a 3-Tier architecture, in particular it enables you to enforce security in the following components:

- Web Servers
- Portals
- Servlets
- J2EE application servers
- Java applications
- .NET and COM applications – such as ASP pages

Its important to introduce it now – as it's a fundamental part of our vision on how to have 3-Tier architectures and DCE applications within a single security framework.

For more information on the AssureAccess Integration Points refer to the ***Application Integration with AssureAccess*** white paper.

User Repositories

One of the powerful facilities of AssureAccess is the ability to support a wide variety of "user repositories". User repositories are where the definition of users and their logon credentials are maintained. In DCE terms this would be the Security Service. In many 3-Tier environments user information are maintained in LDAP directories or databases.

AssureAccess supports the notion of User Repository Connectors (URCs) which the product uses to ascertain when a user identity and their supplied authentication credentials are valid during an authentication process. In addition the URCs are responsible for obtaining the relevant user session attributes, for instance group and organization membership. AssureAccess provides a number of standard URCs, together with the ability to produce custom URCs.

Policy Framework

AssureAccess provides a platform-independent policy management, deployment, decision, and enforcement framework. In addition to providing policy-based access control for protecting resources, the framework also provides policy for audit, authentication, profiling, and administrative subsystems. In fact, the framework is arbitrarily extensible and may be leveraged for any policy requirements. The AssureAccess policy platform provides flexibility to support all levels of access control including historical models such as Access Control Lists (ACLs) and Role Based Access Control (RBAC). The policy framework also supports the hierarchical use of policies within other policies, and the plugging-in of custom rules and policies. The AssureAccess policy model is both extensive and extensible, so there is no policy requirement that cannot be satisfied by this comprehensive framework.

For more information on the AssureAccess Policy Framework refer to the ***AssureAccess Policy Model and Security Framework*** white paper.

Common Authentication and Authorization Platform

Requirement

As described early, a primary requirement when an Enterprise supports a number of middleware platforms is the ability to manage identity and authorization/permissioning information from a single administration point. This reduces the amount of administration effort required and also reduces the potential for mis-configuration. The following sections describe how this requirement can be satisfied.

Unified Identity and Authentication

In the DCE environment users are registered and their attributes are managed by the DCE Security Service. Many organizations wishing to use other middleware platforms would want to lever this, for instance user authentication to a Portal would be determined by the DCE Security Service. To achieve this, an AssureAccess DCE User Repository Connector (URC) is used. AssureAccess integrated into a Portal, Web Server or Application Server would make use of the DCE URC to perform user authentication and gather up the user's session attributes. The AssureAccess DCE URC is provided with the user's username and password and then passed into the DCE Security Service for validation. If valid, then the session attributes would be then provided back to AssureAccess, such as "group membership".
This facility gives you two very powerful features:

- The 3-Tier web environment **and** the DCE environment would have a common user repository – that is the DCE Security Service. Therefore you only require a single user administration facility across both environments

- The DCE user session attributes can be used to make authorization decisions on whether a user can access resources within the 3 tier web environment.

A Custom URC for DCE is available as a separate product. Please contact Entegrity for more details. Figure 2 illustrates the position of the AssureAccess DCE URC in an Architecture.

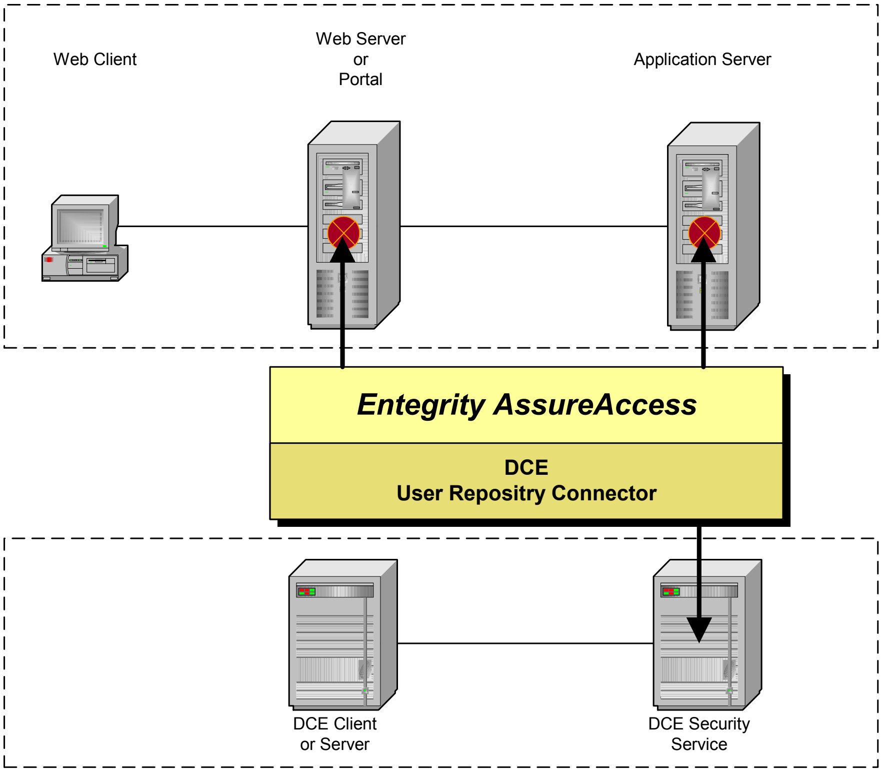


Figure 2: DCE User Repository Connector

Authorization

We have just described how to have a common identity and authentication platform, but what about the authorization and permissioning system? First it is useful to briefly describe the current DCE authorization system. DCE authorization is based on the POSIX Access Control List (ACL) model. Resources such as servers, directories, files and other types of resources can have an associated ACL that specifies which operations a principal (e.g. a user) can be performed. Typically this is implemented by a component called the ACL Manager. Access decisions are made according to the ACL entries and the principal's session attributes, such as username and group membership. ACL Managers are embedded with a number of DCE services, including CDS and DFS. DCE is designed so that the ACL Manager can be replaced or even that you can have multiple ACL Managers controlling access to the same resource. In this case the ACL Managers are "chained" together with one performing access control decisions according to one model and another ACL Manager according to a different model. There are a number of different access points to ACL Managers, and can be achieved using either a management tool (acl_edit), APIs or interfaces.

The approach in this situation is to use the AssureAccess authorization engine to make access control decisions. This is achieved in either of two ways:

- *Replacement ACL Manager:* The standard ACL Manager is replaced, with all APIs and Interface maintained. However the actual access decisions are made by AssureAccess, the replacement ACL manager called an AssureAccess adapter. Figure 3 illustrates the design.

- ***sec_acl() wrapper:*** this is a lightweight solution and does not require the installation of a full ACL Manager. It is intended for those situations where new or existing applications just want to make use of the sec_acl() API. The implementation of this can be imagined as a wrapper around the AssureAccess authorization calls– using the sec_acl() API.

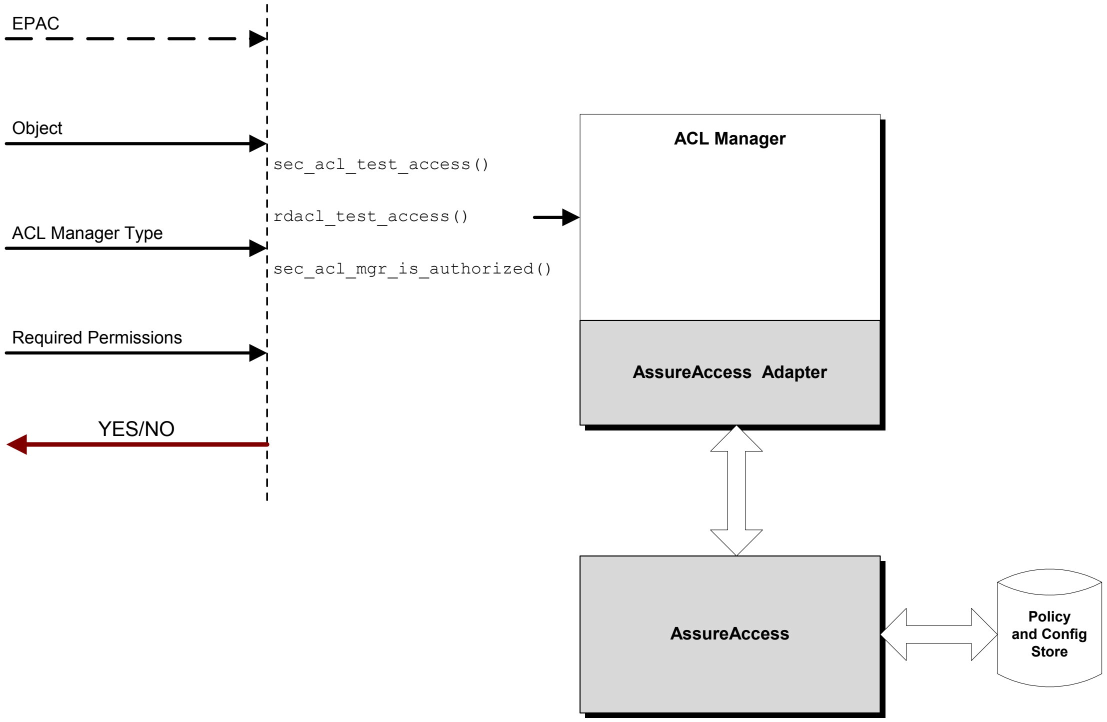


Figure 3: Authorization with AssureAccess

So what can you do when using either the enhanced ACL Manager or sec_acl() wrapper – that DCE does not already provide?:

- It provides a common authorization platform so that the same authorization policies can be enforced across a number of middleware platforms – and of course from **a single administration point**

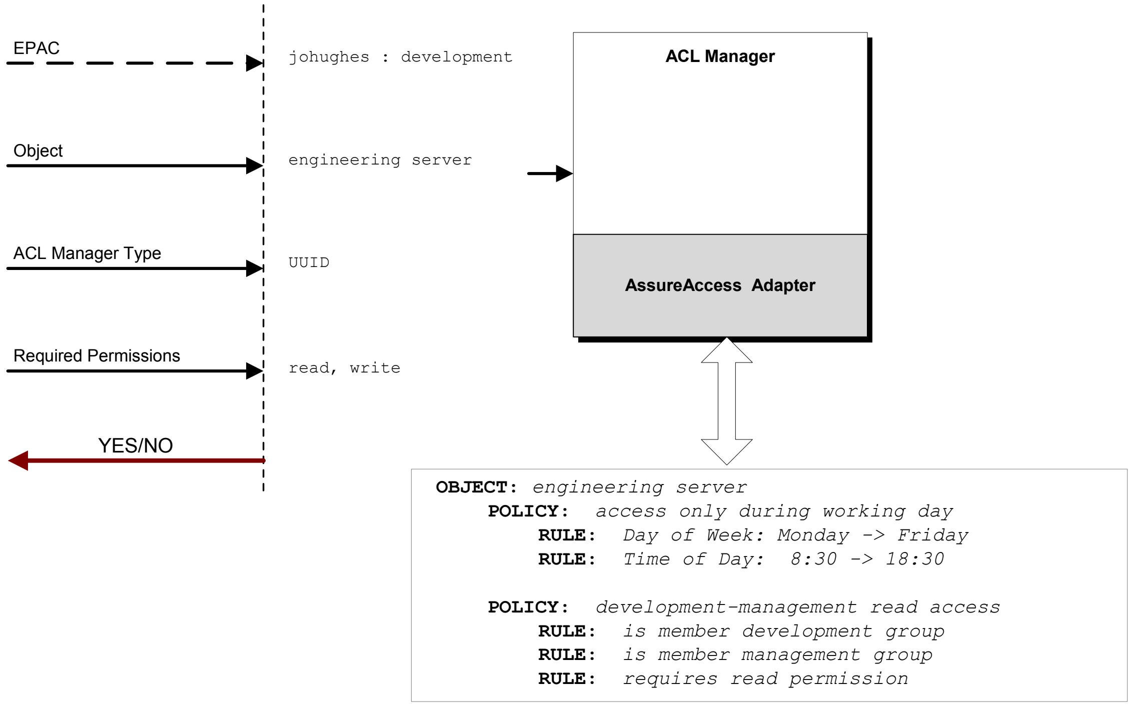
- It will enable you to enforce a Role based Access Control (RBAC) model, as well as the traditional POSIX ACL model supported by DCE

- In addition it will also allow you to enforce a Policy based authorization model

Figure 4 illustrates a simple example of where policy-based enforcement is combined with a traditional ACL style of enforcement. In this case AssureAccess has been configured so that the object "engineering server" has associated with it two policies, both of which has to be satisfied in order for user johughes (who is a member of the "development" group) to gain access to the server. The first policy ("access only during working day") is an example of policy-based enforcement and consist of two rules that dictate the day and time by which the user can gain access to the server. The second policy ("development-management read access") is an example of how a DCE ACL would be expressed within AssureAccess. A given policy can consist of any number of rules and an object can have associated with it any number of polices.

A powerful aspect of AssureAccess is that having defined a set of polices, the same policies could be assigned to a number of different objects – throughout the distributed system. Therefore the policy ("access only during working day") could be applied to each server within a system, and potentially to each file and directory.

To gain a more complete understanding of the power of the AssureAccess authorization model you are encouraged to read the ***AssureAccess Policy Model and Security Framework*** white paper.



```
OBJECT: engineering server
    POLICY:  access only during working day
        RULE:  Day of Week: Monday -> Friday
        RULE:  Time of Day:  8:30 -> 18:30

    POLICY:  development-management read access
        RULE:  is member development group
        RULE:  is member management group
        RULE:  requires read permission
```

Figure 4: Policy Enforcement on an Object

Bridging and Wrappers

There are two generic techniques available should you wish for applications on different middleware platforms to communicate with each other. They are:

- ***Wrappers:*** These have one of more characteristics. They enable a program written in one language to call a service that has been implemented in another programming language – and/or – they provide a higher level programming interface.

- ***Bridges:*** There are two types of Bridges, dynamic and static. Static bridges involved source level code changes whilst dynamic bridges are analogous to "gateways" (e.g. performing on the fly protocol conversion). A dynamic bridge does not require source code modifications

The following sections describe our offerings in this area, however note:

- We only offer Bridging into DCE servers – that is we have no support for Bridging into or from DCE clients. If you would like such a facility we would be more than happy to discuss the requirement with you.

- We only provide dynamic bridges – not static bridges. As use of static bridges always requires source code modifications then we advise that one of the wrappers is used.

Figure 5 illustrates that the Bridging and the Wrappers provide the ability for middleware platforms and components to securely interoperate with each other. Throughout the following sections the role of AssureAccess will be presented in how it provides the "security glue".

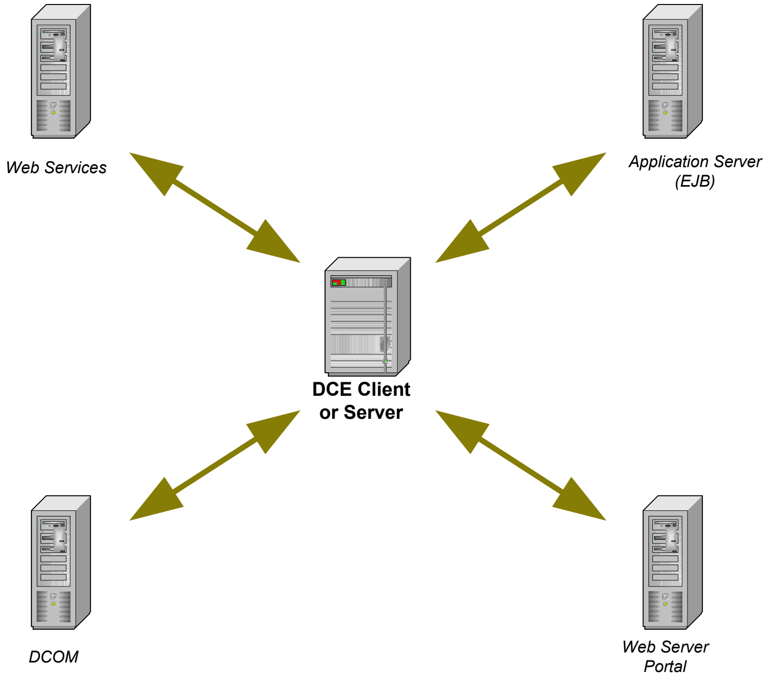


Figure 5: Bridging and wrappers

Wrappers

Java DCE Wrapper

The Java DCE Wrapper is a Java Framework that provides a high level API to DCE interfaces and API calls.

An example of its use is provided in Figure 6. In this case we have a Portal and users accessing the Portal wishing to execute DCE based applications hosted on a remote DCE Server. There are a number of important points to be made regarding the architecture in figure 6.

- This is an example of where the AssureAccess DCE URC can be used to implement a common authentication platform. Users challenged by the Portal and the normal DCE users are both authenticated using the DCE Security Service.

- The DCE Portlet can contain authorization logic by calling the appropriate AssureAccess API – thereby permitting enforcement of ACL, RBAC and policy-based access decisions.

- Portlets are usually written in Java, with most vendors providing a Portlet development kit. The Java DCE Wrapper permits easy interaction with DCE applications.

For more information on how AssureAccess can secure Portals please refer to the ***AssureAccess Portal Blueprint***.

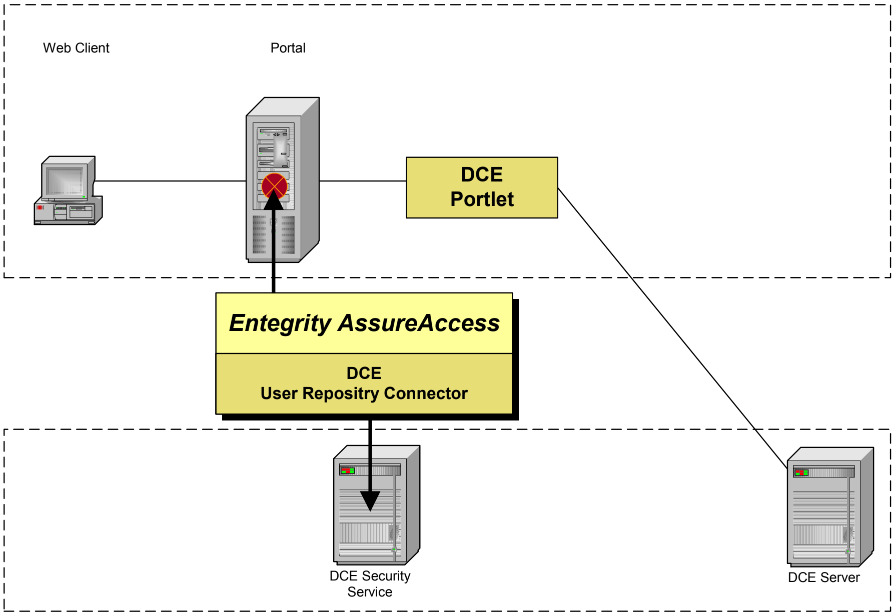


Figure 6: DCE Portlet using Java DCE Wrapper

COM DCE Wrapper

The COM DCE Wrapper is very similar to the Java DCE Wrapper, but using the Microsoft COM interface.

Bridges

Java and CORBA Bridge Server

Overview

The Java and CORBA Bridge Server permits either Java programs, Java Beans , EJB's or CORBA applications to call DCE servers. As this is a dynamic bridge it does not any source code modification in the DCE application. Interface information, in the form of the DCE IDL files are provided to the Bridge Server. Figure 7 illustrates the placement of the Bridge Server within an architecture. Note the use of AssureAccess. AssureAccess is used to provide authentication, single sign-on and authorizations services to the Bridge Server. These are explained in more detail in the following section.

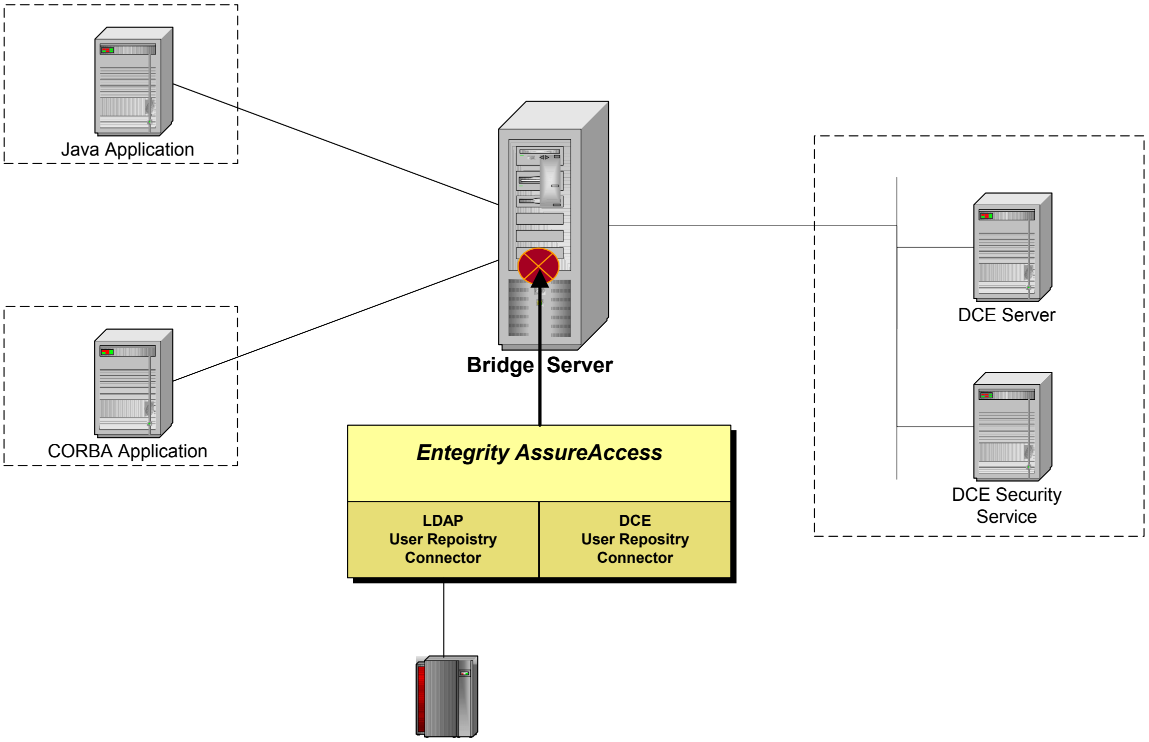


Figure 7: Bridge Server with AssureAccess

The Bridge Server exposes the follow interfaces to an application:

- RMI service – over IIOP or IIOP/SSL
- CORBA objects
- EJB beans – either session or entity.

The Bridge Server maps the calling application RPCs to the appropriate DCE interface. Any attributes transported by the calling application (for instance using CSIv2) is mapped into a DCE EPAC.

Security

If channel security between the Java and the CORBA applications to the Bridge Server is necessary then this is provided by using Secure Sockets Layer (SSL). The connection between the Bridge Server and the DCE components is provided by the usual DCE security services supported by DCE RPC

Identity propagation from a Java or CORBA client into the DCE environment can be achieved by a number of different mechanisms and depends on the application environment being supported. Mechanisms supported include:

- CSIv2 using username/password authentication over SSL. In addition attributes can be transported using the SAS protocol within CSIv2
- Using username/password when using RMI/IIOP/SSL
- Client Authentication using SSL

The Bridge Server uses AssureAccess to provide a number of important security functions. These are:

- Authenticates the supplied credentials (e.g. username and password). AssureAccess could be configured to authentication against the DCE Security Service, a LDAP directory, Windows NT or a database.
- Perform authorization checks to determine whether access to the DCE interface can be achieved according to the defined policies. For example taking into account the authenticated identity, environmental attributes (e.g. time of day) and connection attributes (if a SSL session and what key size)

AssureAccess can also play a role in propagating identity information between the two environments. Within a 3-Tier Web environment AssureAccess propagates identity and attribute information around the environment ensuring that suitable authentication and authorization is enforced. When used with the Bridge Server AssureAccess also propagates this information from the 3-Tier Web environment into the Bridge Server, enabling authentication and authorization to be performed and the user's attributes transferred into a DCE EPAC for use by the DCE application.

Figure 8 illustrates the role of AssureAccess in providing security features/

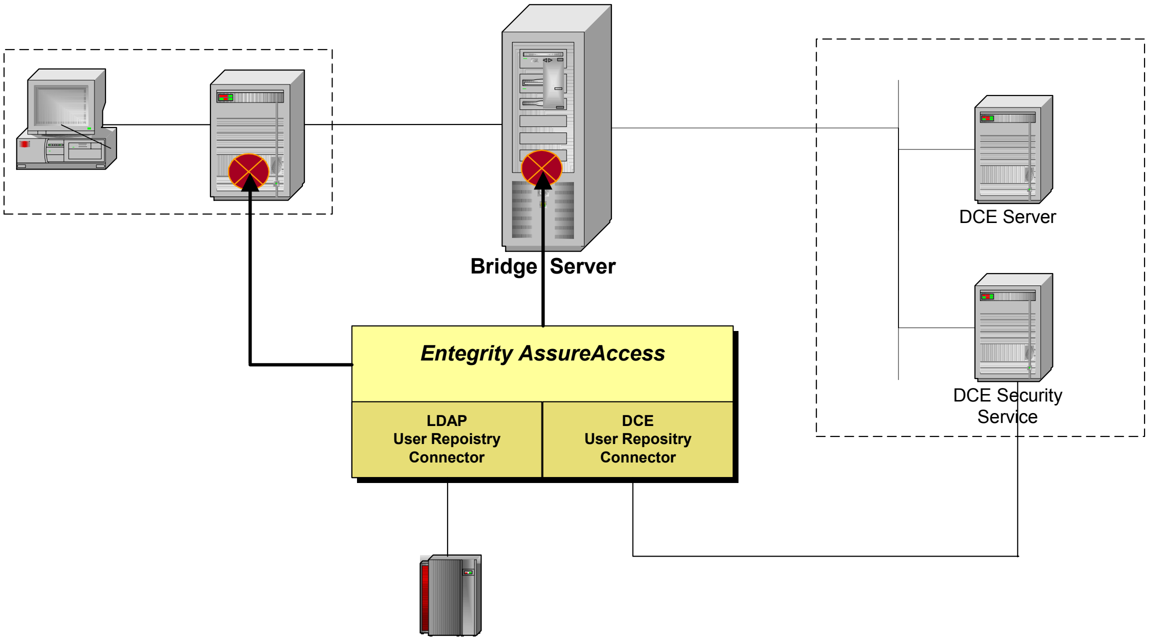


Figure 8: Bridge Server with AssureAccess

Web Services Bridge Server

Overview

The fundamental building block of Web Services is the Simple Object Access Protocol (SOAP). SOAP is a protocol that uses HTTP to provide a simple request/response facility. SOAP is defined using Extensible Markup Language (XML). XML is a flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere. It is in essence a type of RPC mechanism.
DCE applications could interact with web services in two manners:

- The DCE application could call a web service to obtain information. In this case the DCE application would become the requester.

- A web service based application could call a DCE server to obtain information. The DCE server therefore becoming the web services responder.

But what about security and identity and attributes propagation. Web Services security is being defined by the OASIS standards body – of which Entegrity are members. There are two work items that are of interest:

- ***SAML (Security Assertion Markup Language)*** is a XML standard that allows a user to log on once for affiliated but separate Web sites. SAML is designed for business-to-business and business-to-consumer transactions. SAML specifies three components: assertions, protocol, and binding. There are three assertions: authentication, attribute, and authorization. Authentication assertion validates the user's identity. Attribute assertion contains specific information about the user. An authorization assertion identifies what the user is authorized to do.   In this context it permits authentication between sites as well as transporting an entities credentials. SAML 1.0 is a published OASIS standard. Entegrity's AssureAccess product was one of the first products to support SAML in a released standard version.

- ***WS-Security (Web Services Security)*** is a OASIS Technical Committee developing standards that addresses security when data is exchanged as part of a Web service.. WS-Security specifies enhancements to SOAP messages and is primarily aimed at protecting the integrity and confidentiality of a message and authenticating the sender. It is based on digital signatures and encryption techniques. WS-Security also specifies how to associate a security token with a message, without specifying what kind of token is to be used.   Tokens could be SAML assertions, Kerberos Tickets or DCE PACs.

Therefore there are two choices for exchanging credential/attribute information with a remote system:

- Map the contents of a DCE PAC into a SAML assertion (which is not constrained on what it carries as far as attributes is concerned). The remote system then needs to understand the SAML assertion

- Carry the DCE PAC in the WSS message. This of course means the remote system has to be able to process the DCE PAC.

Support for WSS or SAML with Web Services is not in the first release of the Web Services Bridge Server, however the general approach is described in the following sections.

DCE Client to Web Service

In this situation a DCE Client issues RPC calls to what it thinks is a DCE server application. In fact it's a Bridge Server that support the defined DCE Interface, The DCE Interface calls are then mapped into the appropriate SOAP message and sent to the Web Services Responder. The resulting response is then fed back to the DCE Client. Figure 9 shows the overall architecture.

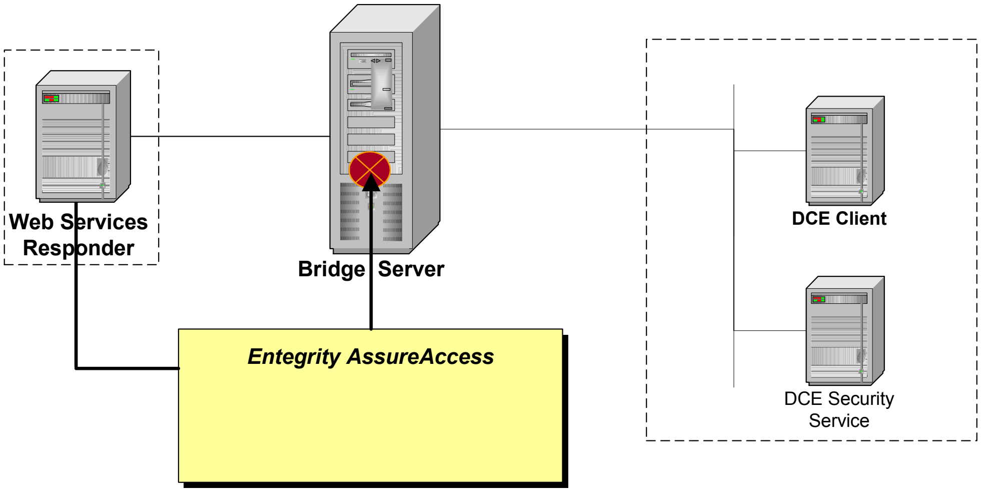


Figure 9: DCE Client to Web Service Responder

Security between the DCE client and the Web Services Bridge Server is provided by the usual DCE security mechanisms.

Policy-based authorization decisions can also be performed by AssureAccess, for instance can this DCE client issues this call to the Web Service.

When the use of SAML within a WSS object has been fully defined by OASIS then AssureAccess can be used to create an appropriate SAML assertion for inclusion in the SOAP message transmitted to the Web Services Responder. The Attributes and identity information in the DCE EPAC would be converted and placed in the SAML assertion. AssureAccess can also be used in the Web Services Responder to validate the SAML assertion. However if you require a more immediate solution then please contact us and we can discuss possible solutions to satisfy you requirements more quickly.

Web Service to DCE Server

The case were a Web Service Requestor make a call to the DCE Server is shown in Figure 10. Its just the reverse of the DCE Client to Web Services Responder.

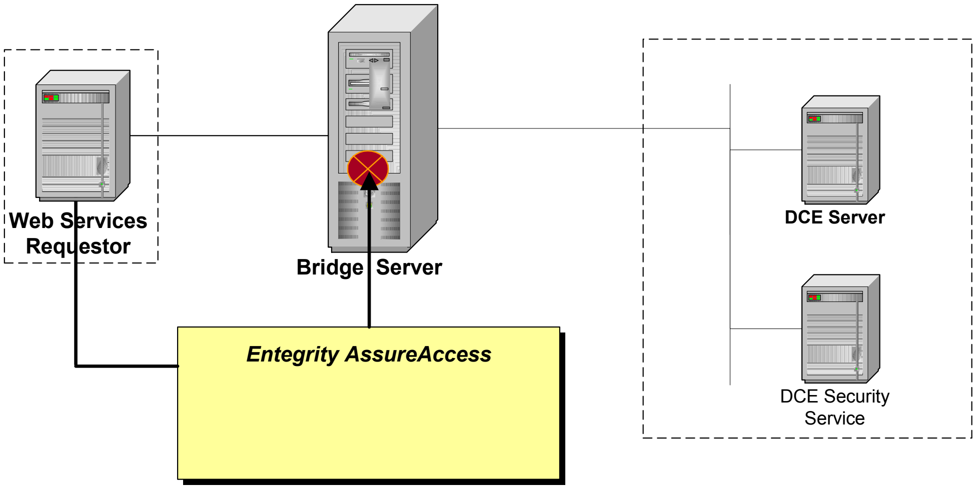


Figure 10: Web Service Requestor to DCE Server

DCOM Bridge Server

Microsoft DCOM/COM+ is based on DCE RPC, with a Object oriented overlay. Hence interworking between a DCE server and DCOM/COM+ is relatively straightforward. As Figure 11 illustrates a Bridge Server is positioned between a DCOM component and the DCE Server. The DCOM Bridge Server will not be available in the first tranche of the Bridge Server release.

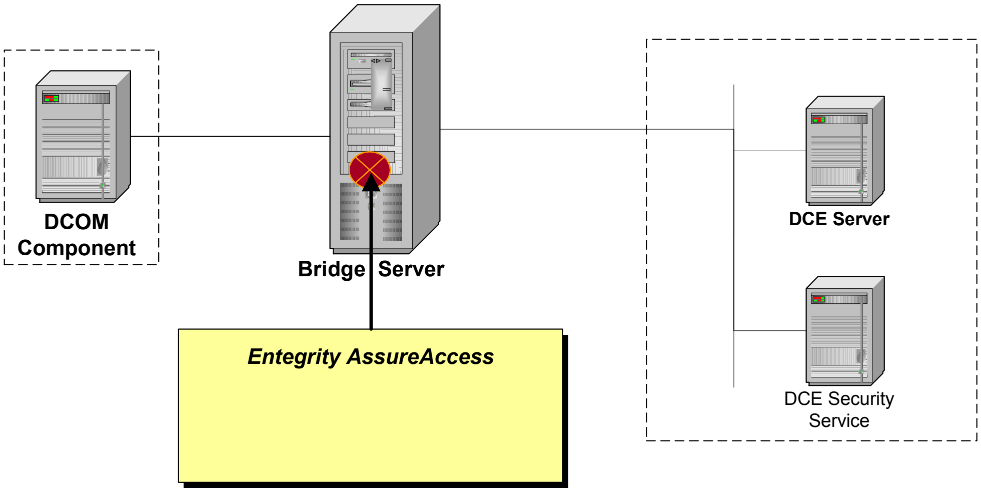


Figure 11: DCOM and the Bridge Server

Conclusion

In this White paper we have described the Middleware landscape and demonstrated that they need not be distinct "Islands" with no inter-connectivity. Indeed we have shown that Entegrity Solutions have products and services that allow:

- A Common access management framework that provides authentication and authorization services across both DCE and 3-Tier Web environments

- Different middleware applications can securely communicate with each other

About Entegrity Solutions

Entegrity AssureAccess access management software provides authentication, single sign-on, authorization, audit, and security policy administration for Web, J2EE, and Microsoft applications. Entegrity also offers secure file transfer and legacy security software products.

For more information about Entegrity Solutions, please contact us at info@entegrity.com or visit www.entegrity.com.

CONTACT

Entegrity Solutions
410 Amherst Street, Suite 150
Nashua, NH 03063
United States

Phone: +1-603-882-1306 x2700
Toll Free (US): 1-800-525-4343 x2700
Fax: +1-603-882-6092

Entegrity Solutions makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Entegrity Solutions shall not be liable for errors contained herein, or for any direct or indirect, incidental, special or consequential damages in connection with the furnishing, performance, or use of this material. Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (i) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Entegrity®, Entegrity Solutions® and AssureAccess® are trademarks or registered trademarks of Entegrity Solutions Corporation or its subsidiaries in the United States and other countries. All other brand and product names are trademarks or registered trademarks of their respective holders.

Sun, Sun Microsystems, the Sun logo, iForce, Java, Netra, Solaris, Sun Cobalt, Sun Fire, Sun Ray, SunSpectrum, Sun StorEdge, SunTone, The Network is the Computer, all trademarks and logos that contain Sun, Solaris, or Java, and certain other trademarks and logos appearing in this document, are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Microsoft COM, DMO etc

Version 2.