Entegrity® AssureAccess®

Technical Overview

This white paper presents an overview of the architecture and technical features of AssureAccess®. It covers Authentication, Single Sign-On, Authorization, Audit, and Administration in the product, and discusses specific features.

Entegrity Solutions® Whitepaper

ENTEGRITY Solutions®

Table of Contents

Executive Summary

Entegrity® AssureAccess® is an access management solution that provides comprehensive single sign-on, authentication, authorization and audit services for e-Business applications. For enterprises deploying business systems to the Internet, AssureAccess provides essential user access management capabilities, greatly accelerating the deployment of e-Business applications.

AssureAccess …

- authenticates users to existing user stores eliminating the need to migrate users into any specific location or repository.
- maintains user session through single sign-on across J2EE and Web applications.
- secures existing applications using bolt-on security that does not require application modifications.
- integrates tightly with applications for fine-grained access control while abstracting security logic.
- provides 15 standard rules for use in policies and allows custom business rules to be added.
- provides policy based domain administration enabling delegated administration.
- provides complete policy based auditing.
- implemented in Java and fully J2EE compliant.
- evaluates and enforces policy decisions at the application and web servers.

Introduction

Access management is a policy-based process for controlling user access to resources within internal and external application environments.

To enable collaborative e-Business processes with employees, customers, partners and suppliers, organizations are rapidly extending applications and portals to the Internet. As companies make mission critical applications and highly sensitive data available over the Internet, the need for automated processes and tools to effectively control user access to business resources becomes a critical requirement. Access management solutions like AssureAccess provide application security and portal security by providing the following key services; authentication, single sign-on, authorization, audit and administration.

This paper gives a product overview, and then covers each of the following areas in turn:

- Authentication – verifying and validating a users identity.
- Single Sign-On – maintaining a user's identity across applications, domains and portals.
- Authorization – determining what applications, application functions and data a user can access.
- Audit – monitoring, recording and logging system events and user access events.
- Administration – creating policies, system configuration and delegated administration.

AssureAccess Overview

There are four key processes to access management. Authentication is the process of determining who a user or system is. Authorization is the process of determining who can access which resources. Resources can be Web pages, Web objects (buttons on a page), or business logic as implemented in an application server. The third component, often overlooked, is audit trail generation. In order to track policy compliance, detect anomalous events and deter insider abuse a central, robust, audit system is required. The fourth component in an access management system is the administration tool that is used to easily manage policies and application security.

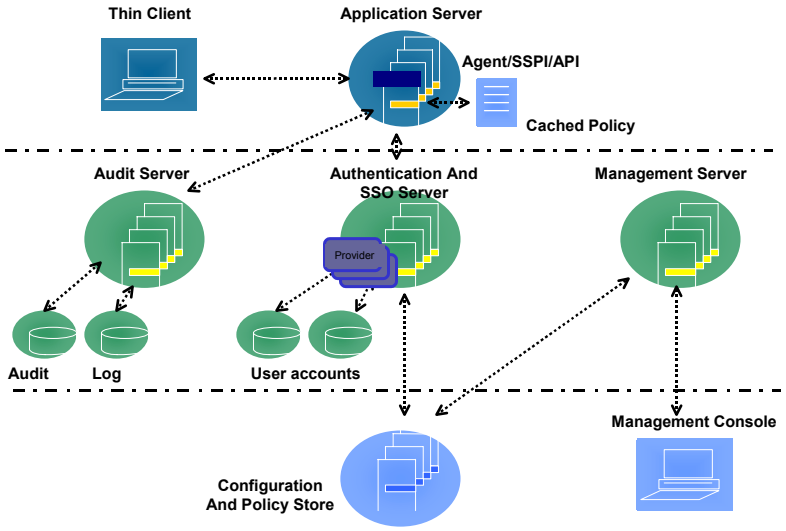
AssureAccess …

- provides complete authentication services, ensuring that only legitimate users gain access to e-Business applications. AssureAccess leverages an enterprise's current technology infrastructure by authenticating users against existing user repositories (LDAP, NT Domain, Relational Database, PKI etc). User information can be collected from multiple sources and made available for authorization and audit.
- provides comprehensive single sign-on as users move from one e-Business application to another – even if these e-Business applications belong to partners, affiliates, Application Service Providers or reside in different DNS domains. Single sign-on creates a superior user experience, increased security and reduced helpdesk costs.
- provides comprehensive fine-grained authorization services by allowing administrators to restrict access to application resources like files, directories, pages, buttons, content,

data, Enterprise Java Beans, even down to a specific method, parameter or arbitrary section of code.

- eliminates the need to code security logic in an application, thereby significantly reducing the cost and time required to develop, deploy and maintain the application.
- enables the creation of flexible access policies, based on real world business logic, which reflect an organization's complex and changing business relationship with their customers, suppliers, partners and employees. AssureAccess makes this easy by providing 15 pre-built access control rules that allow user access based upon time of day, day of week, client or server IP address, DNS domain, encryption strength, authentication provider and more. Policies can be extended by implementing custom rules in Java to include any arbitrary business logic. Furthermore, an administrator can change policies while the application is running without having to take the application offline, or cycle the server.
- provides policy based domain administration, allowing access management administration to be easily delegated, even within a complex hierarchy. AssureAccess allows organizations to segregate access services to optimize application scalability and to accommodate applications with unique security needs.
- provides complete policy based auditing, with audited events being written to a file or database. Organizations have complete freedom to configure this information, and determine what fields will be presented.
- is 100% Java and fully J2EE compliant; allowing AssureAccess to easily integrate with Java based applications and the leading J2EE application servers such as BEA WebLogic Server and IBM WebSphere.
- evaluates policies locally at the application and web server, eliminating the need for a central policy server and enabling superior performance. The AssureAccess architecture allows linear performance increases as additional web servers are added and can support millions of users. AssureAccess servers can be replicated and clustered to support increasingly heavy traffic while maintaining high levels of reliability, and AssureAccess supports automatic fail-over in the event of a server failure.

Architecture



The AssureAccess architecture is one of the keys to its scalability and performance. The following section describes the major components.

Client

No AssureAccess software is installed on the client. Typically the client would be a standard Web browser, but it could also be an Enterprise JavaBeans client. AssureAccess can support clients which use a variety of authentication methods, from user name and password to PKI or even smartcards or biometrics.

Application Server Adapter

AssureAccess adapters are provided for popular Web and J2EE servers. The adapter facilitates the authentication process, including Single Sign-on, evaluates policies, and provides enforcement. Adapters provide both bolt-on solutions that do not require modifying applications, and integrated solutions that allow more flexibility and control when securing applications.

Audit Server

The audit server collects audit and log records and writes them to disk. Audit records refer to security-related events (for example, create user) and are controlled by audit policies. Log records record system events (for example, server startup) and are written unconditionally. All the AssureAccess components generate audit and logging calls at appropriate times. In addition, user applications can use the audit API to log their own events.

Authentication Server

The Authentication Server manages user authentication to any source of user information as specified by authentication policies. The Authentication Server also acts as an Attribute Authority. It assembles user attribute information from all authoritative sources and creates an Attribute Certificate (AC). This digitally signed information object is used as the basis of authorization decisions and to support single sign-on. By means of pluggable authentication providers, AssureAccess can support user populations maintained in LDAP, Windows NT, or custom sources such as a proprietary database.

Management Server

The Management Server is the proxy within the trusted computing base for the Java Swing Management Console. As an RMI/SSL server, it ensures that administration information is protected. Access ports to the server are fully configurable, allowing administration from outside a firewall.

Policy Store

AssureAccess uses a standard LDAP directory to store all of its configuration and policy information that is shared among server and adapter replicas. Using a directory allows Assure Access to take advantage of LDAP features such as replication and fail-over. In addition, LDAP can be the user store for the system. Where LDAP has already been deployed, AssureAccess is compatible with industry standard products.

Authentication

At a basic level, authentication involves verifying user credentials. AssureAccess supports authenticating users with a wide variety of credentials, typically from diverse user stores. Enterprises can employ multiple authentication schemes at once, ranking them by their security using the AssureAccess Authentication Strength functionality. For applications accessed via browser, custom forms can be served through AssureAccess adapters directly. When used in conjunction with Authentication Strength, Forms Based login can be used to manage authentication such that users can maintain multiple identities simultaneously. Finally, AssureAccess provides Forced Logout functionality to invalidate user sessions in real time.

User Store Support

One of the key features of AssureAccess is the ability to seamlessly support multiple types of Authentication. Password–based login is the most commonly used method today, and AssureAccess allows the use of any existing repository of usernames and passwords. User names can be stored in LDAP, Windows Domain, RDBMS or almost any other type of data store. Multiple repositories can also be used. This means that there is no need to copy or migrate this data into a specialized store. Multiple types of password hashing or encryption can also be used.

Other types of Authentication can also be used. Some examples include hardware token cards, Public Key Certificates, Smartcards and biometric devices that conform to the PKCS#11 or Microsoft CSP interfaces. External login by firewalls or VPNs can also be supported. Any

combination of these methods can be supported in the same environment. Individual users can be enabled for more than one method of Authentication. The method used can be used to decide if access is allowed. This is described below under Authentication strength.

Like everything else in AssureAccess, Authentication is controlled by policy. Authentication policy can serve two basic purposes. An authentication policy can be used to add additional conditions or restrictions on Authentication. For example, it could be used to disable an account after a specified number of bad passwords have been typed. Or it could be used to check a central system or database to see if the account in question had been revoked. Or it could restrict login by date/time or network location.

The second major use of Authentication policy is to enhance the attributes associated with the user session. As described below, under single sign-on, once a successful Authentication has occurred, all information about the user is collected into an Attribute Certificate (AC) which can then be used for Authorization policy decisions or accessed directly from an application via the API.

Authentication Strength

Many applications have resources that have different security requirements. These security requirements may be related to how a user authenticates to the system. For example, a user may only be required to supply a username and password to access some resources but may be required to use a smart card to access more sensitive resources.

AssureAccess provides the ability to set an authentication strength for each authentication provider. A resource can then be assigned a policy that contains an "authentication strength" rule that specifies the level of authentication required to access the resource. If an authenticated user requests a resource that requires a higher authentication strength, the user will be prompted to re-authenticate at the higher level. The Web or J2EE adapter automatically serves the login form for the authentication provider with the required strength. In this way, AssureAccess automates the process of creating a multi-tiered authentication front end for applications with complex security requirements.

Forms Based Login

Managing sign-on to an application is a common problem that application developers must solve. AssureAccess automates this process by providing configurable templates for forms-based authentication used by the AssureAccess Web and J2EE adapter. The templates are simple HTML forms that can be modified to include any number of required input fields. Each authentication provider can have its own form which is invoked when that provider is used. A form can also be shared between two or more providers. Policy decisions now have "grant", "deny", and "require authentication" results. A "require authentication" result from a policy evaluation signals the Web and J2EE adapters to serve the login form.

Figure 1 shows an example login form served by the AssureAccess Web adapter. In this case, the user has made a request for a resource that is protected by a policy that requires that the user be authenticated. The policy evaluation gives a "require authentication" result. AssureAccess then determines which authentication provider should be used and serves the appropriate form for that provider. The user enters proof materials (e.g. username and password). If the login is successful, the browser is re-directed to the originally requested page.

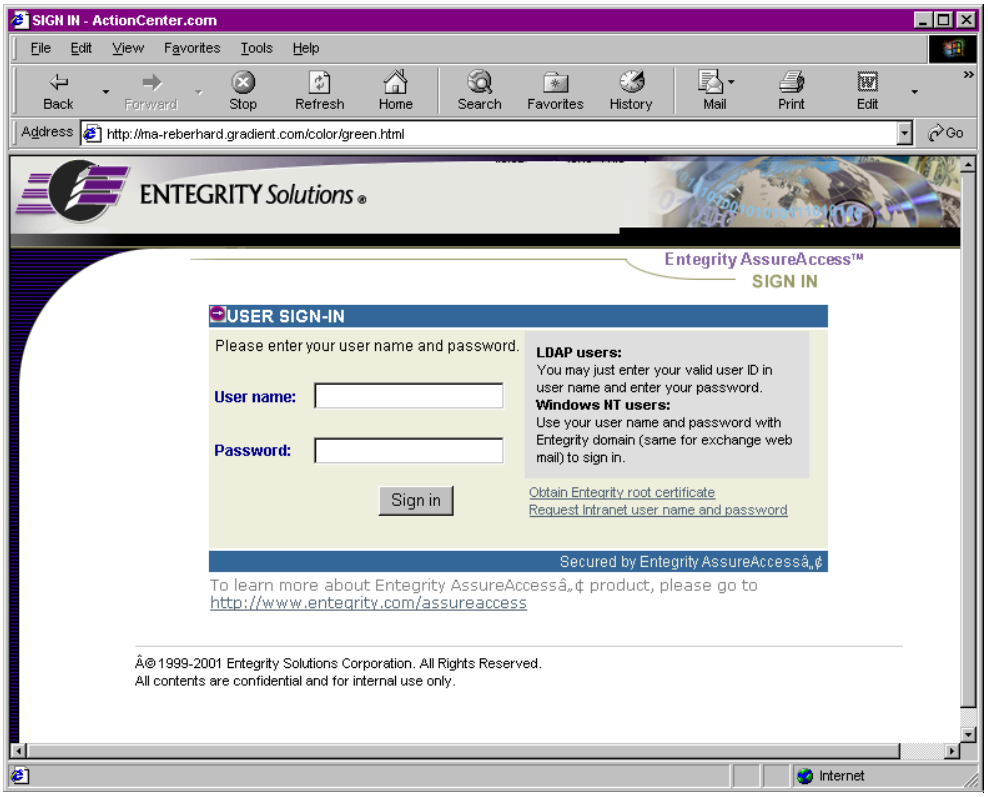


Figure 1: AssureAccess provides configurable forms to create a multi-tiered authentication front end for your application

Forms based login can be combined with authentication strength to create a multi-tiered authentication front end for the application.

Forced Logout

AssureAccess permits a secure logout by either a user or privileged administrator. It is true that a user can effectively logout by exiting from his or her browser, causing the cookie containing the Authenticator to be destroyed. However, this does not allow the AssureAccess infrastructure to initiate a logout or be certain that a copy of the Authenticator doesn't still exist somewhere.

AssureAccess provides the capability for a user to log out of his or her current session. Also an authorized administrator can force the logout of any user. Both these capabilities are available from the AssureAccess API. This means that applications can easily provide a user logout function.

The administrative logout capability is also provided by the Management Console. This not only allows an administrator to clean up unwanted sessions, but also is very useful when user attributes have been changed. By forcing the user to login again, the administrator can ensure that the updated information will be associated with the user.

When a user or administrator requests the logout of a user session, all copies of the associated Attribute Certificate (AC) are removed from the single sign-on server. In addition, all adapters are informed of the logout, causing them to delete any cached copies of the AC as well.

Single Sign-On

Single sign-on is a specific use case of authentication. Once a user has logged into an application, single sign-on propagates their identity to the entire system. AssureAccess enables not only single sign-on between load balanced servers, but also between diverse applications potentially running in different DNS domains.

General Capabilities

AssureAccess provides single sign-on across any number of web and application servers. What this means is that a user need only login one time, using any Authentication method and he or she will be recognized as a valid and active user by all applications. All the user's attributes will be available for Authorization policy decisions and will be accessible by calls to the API from anywhere.

This provides several benefits. Computer systems can be deployed as a pool, running the same applications. Users will not be aware of which system is actually processing their requests. This allows smooth scaling of the configuration, graceful degradation in the event of a hardware failure and the ability to add and remove individual computers at will to meet operational needs.

Single sign-on also allows multiple applications to be seamlessly combined into a single application or portal. Authentication logic does not have to be built into individual applications. Applications do not have to deal with unique interfaces or attributes. Applications built with static html pages, cgi scripts, COM objects, Java Server Pages or Enterprise Java Beans will all function together in a smoothly integrated whole, regardless of the system they execute on.

The key to AssureAccess' single sign-on capabilities is the Attribute Certificate. When a user successfully Authenticates, AssureAccess creates an Attribute Certificate (AC) that contains all of the user's attributes. The AC is cached in memory for use when policy decisions are made. At the same time a secure Authenticator is created that refers to the AC. The authenticator is returned to the user in a web cookie. When the user makes another request, the cookie is presented, the Authenticator is used to find the user's AC and a policy decision is made about whether to allow the requested access.

Single sign-on is implemented by having the Adapter that performed the original login store a copy of the AC in the single sign-on server. When the user presents the Authenticator to a different adapter, the corresponding AC will not be found. The adapter therefore requests the AC from the single sign-on server and loads it into its own cache. It can then be used for access control decisions.

Cross DNS Domain Single Sign-On

AssureAccess also allows single sign-on capabilities to operate across web and application servers in different DNS domains, for example: www.widget.com and www.gizmo.com. To understand how cross-domain single sign-on works, it will help to review some of the challenges to implementing the functionality.

Implementing single sign-on across domains is complicated by two factors. First, a cookie created by one server cannot be read by a server in a different DNS domain. This behavior is specified by the HTTP protocol to help protect user privacy. Second, when servers are in different domains they usually will not be able to access the same single sign-on server. Cross-

domain requests are also likely to pass through one or more firewalls, which may impose further restrictions on the protocols used.

With cross-domain single sign-on, one web server must be designated as the Login Server. All the other servers will act as clients and are configured with the URL of the Login Server. Only the login server can process logins. It is up to the application to redirect users that need to login to the Login Server.

After validating the user's credentials, the Login Server creates an AC as usual. The login server also creates a cookie containing the Authenticator, for its own future use. Because the servers in other domains will not be able to read this cookie, the Login Server modifies the original request URL, adding the Authenticator as an encoded parameter.

When the request is redirected to the original CDSSO client, it uses the encoded parameter to create its own cookie for future requests. It also calls the Login Server to fetch a copy of the AC. This call is made using HTTP or HTTP with SSL, thus minimizing the issues of firewall traversal.

When a user who has previously logged in makes a request of a new server, that server redirects the request to the Login Server. The Login Server determines from the cookie that the user is already logged in. It therefore simply encodes the Authenticator into the URL and redirects the call back to the CDSSO client. That server repeats exactly the same process as above, creating its own cookie containing the Authenticator and fetching the AC from the Login Server.

Authorization

Authorization consists of two distinct components:

- Policy decision determines whether a given access request should be allowed.
- Policy enforcement causes the access to occur or an error to be returned, based on the policy decision.

Policy Decision

AssureAccess makes policy decisions as follows:

- When a server starts and periodically thereafter, all policies relevant to resources controlled by the server are loaded from the AssureAccess Policy Store.
- When a user successfully authenticates, all of his or her attributes are assembled into a set of digitally signed credentials called Attribute Certificates.
- When a request is made, all the policies associated with that request are identified.
- The policies are evaluated in order using the information in the credentials and environmental information as appropriate.
- Once the result of the evaluation has been determined, the result is returned.

Available Policies

Advanced policy definition and enforcement is at the heart of AssureAccess. All four key processes, Authentication, Authorization, Audit and Administration are controlled by policies. The use of dynamic policies provides the ultimate in flexibility and integration.

AssureAccess policies have the following properties:

- Policies are named, allowing them to be applied in multiple places.
- Authorization and Audit Policies are associated with Resources (Associations). A Resource can refer to a Web page, Java method, or any application-defined entity.
- Authentication Policies are associated with individual users or classes of users.
- Administration Policies are associated with an AssureAccess Domain or set of Domains.
- Multiple policies can apply to the same Resource, user class, or Domain set.
- The wildcard character "*" can be used to apply policies to multiple application resources.

A policy consists of one or more rules.

- Rules can be combined with the logical **AND** and **OR** operators. (X **AND** Y) **OR** (A **OR** B)
- Rules can be negated. (**NOT** Q)
- Rules are evaluated in strict order and evaluation stops once the result is determined.
- Rules can test for a user attribute. (Is a member of ADMIN Group)
- Rules can test the value of an attribute. (Limit > $1000)
- Rules can test the current date and time or day of the week. (DayofWeek is Saturday, Sunday)
- Rules can test the IP address of either the client or the server. (ClientHostIP is 192.168.38.*)
- Rules can test the strength of authentication used and the strength of encryption used.
- Audit rules can test whether the request succeeded or failed.
- Custom rules, written in Java, can be used at any point.

Policy Enforcement

Once the policy decision has been made, it must be enforced. AssureAccess provides a choice between the maximum convenience of no required programming or the maximum flexibility of complete programmatic control of decision enforcement.

- If the request applies to a standard resource, such as access to a Web page, Servlet or Enterprise Java Bean, AssureAccess enforces the decision.
- If the request applies to an application-defined resource, the application must enforce the decision.

Bolt-On Solutions

AssureAccess provides a number of security solutions called "Bolt-On" solutions because they do not require applications to be modified before they are protected. In the J2EE environment, the Servlet Filter and Universal Java Plug-In protect resources served from both the Servlet and EJB containers. AssureAccess also supports common Web servers using filters that implement the server APIs.

Web Adapter

AssureAccess provides adapters designed to work with iPlanet, Microsoft and Apache web servers. They are implemented as web filters which inspect incoming requests, execute the policy decision and either allow the request to be passed on to the web server or return an error

to the browser. AssureAccess allows an administrator to specify a custom error page to be displayed when access is denied.

Servlet Filter

A filter mechanism for intercepting and acting on requests for JSPs/servlets has not been available in J2EE application servers until recently, when the Servlet API 2.3 specification was approved.

AssureAccess provides a servlet filter that makes use of the new Servlet API standard. This enables bolt-on integration for resources in the JSP/servlet container of the application server. This functionality allows application developers to protect Web-based resources in the J2EE application server without having to write any code. Like Web resources, JSPs and servlets can be protected by complex security policies. Using this feature, policies are created and assigned to specific resources in the AssureAccess management console. Web-like features such as forms based login and authentication strength are also available in the servlet filter.

Universal Java Plug-In

The J2EE specification provides the capability to define security policy in terms of granting permission to access resources based on role. It is then up to the application server vendor to provide a proprietary method for granting roles to principals. Unfortunately, protections are static and do not allow for dynamic policy decision. Furthermore, access control changes require modification of deployment descriptor files and subsequent re-deployment of the application.

The AssureAccess Servlet filter provides a solution for resources in the servlet container of the application server. But what about the EJB container? Most access management products provide some type of solution for protecting EJBs. Historically, there have been two approaches. The first approach involves modifying EJB source code to include API calls to the security service. This is a time consuming process that requires re-coding the application. The second approach involves the use of a pre-compiler together with proprietary classes that replace the original J2EE server classes. The problem with this approach is that the pre-compiler and replacement classes must be supplied and maintained by the security vendor for every version of each vendor's application server.

AssureAccess takes a wholly new approach. It provides a breakthrough technology called the Universal Java Plug-in which allows the administrator to incorporate policy-based access control into their J2EE applications with no code changes and no pre-compile steps. In fact, this powerful new technology even works for applications where the source code is unavailable.

The AssureAccess Universal Java Plug-in works in both the servlet and EJB containers allowing you to protect all Java resources the same way. It automatically maintains user identity between the servlet and EJB containers as well as between application servers – even if those servers are from different vendors. The Universal Java Plug-in in AssureAccess supports integration with Java applications on BEA's WebLogic Server 6.x and IBM's WebSphere 4.0 servers. Support for other J2EE application servers will be added in the near future.

The main screen for selecting applications to secure using the AssureAccess Universal Java Plug-in is shown in Figures 2. The user starts by selecting a Java application that has been deployed on a WebLogic or WebSphere server. AssureAccess will then scan the application to find all the resources in the application – JSPs, servlets, EJBs, and methods. The deployed application is then instrumented to include AssureAccess integration points. The final step is to create policies to control access and assign those policies to the resources you wish to protect.

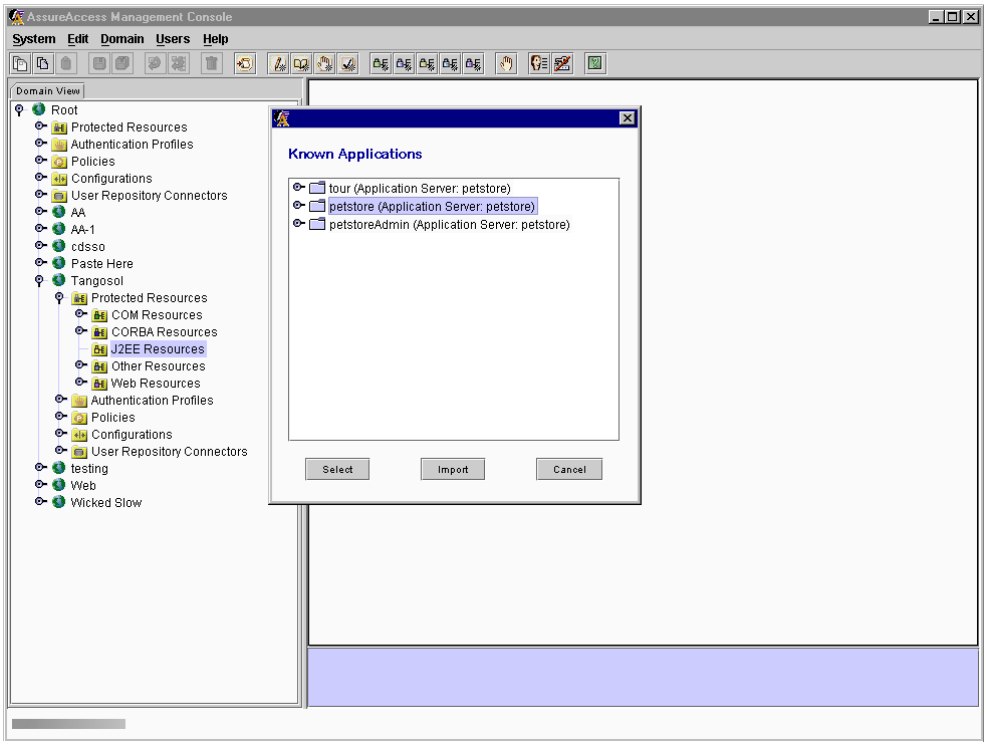


Figure 2: To use the AssureAccess universal Java Plug-in, you start by selecting a Java application you wish to protect.

Integrated Solutions

Although Bolt-On solutions can protect many resources, AssureAccess stands out from the competition when integrated into applications to provide fine-grained access control. Applications that embed policy decision and enforcement points using the JSP Tag Library or Java API in the J2EE environment, or the COM adapter in the Microsoft environment, have far greater versatility in security options while still abstracting security logic from the applications and allowing central management.

Java API

In addition to the automatic enforcement of authorization, AssureAccess provides a comprehensive Java API. The API enables all the capabilities of the product, thus allowing unlimited flexibility in applications or management utilities. In fact, the Management Console, described below, is implemented entirely using the API.

The Java API allows applications to call the policy decision engine and then enforce access decisions on arbitrary resources that are completely internal to the application. The API also allows access to common functions such as accessing user attributes for display or internal processing.

In addition to the comprehensive Java API, two other programming interfaces are provided for the convenience of developers in the JSP and ASP environments, respectively.

JSP Tag Library

Internet applications typically have presentation logic coded as a set of application JSPs while business logic is normally contained in a set of application EJBs. AssureAccess provides fine-

grained access control for elements in a JSP through the use of the API embedded in the JSP code. This allows the developer to control which elements on the page would be available to a particular user. Those elements might include text, graphics, or links on the page.

However, AssureAccess makes it much easier for JSP developers to incorporate AssureAccess functionality in their applications by providing a JSP tag library. The JSP tag library simplifies the development of JSP pages that need to authentication and authorize users, check attribute values, or conditionally omit HTML code (personalization) through authorization or attribute value checks. The tags also allow the developer to choose to redirect to another page or to emit HTML code on condition failure.

A typical use of the AssureAccess tag library is shown in Figure 3.

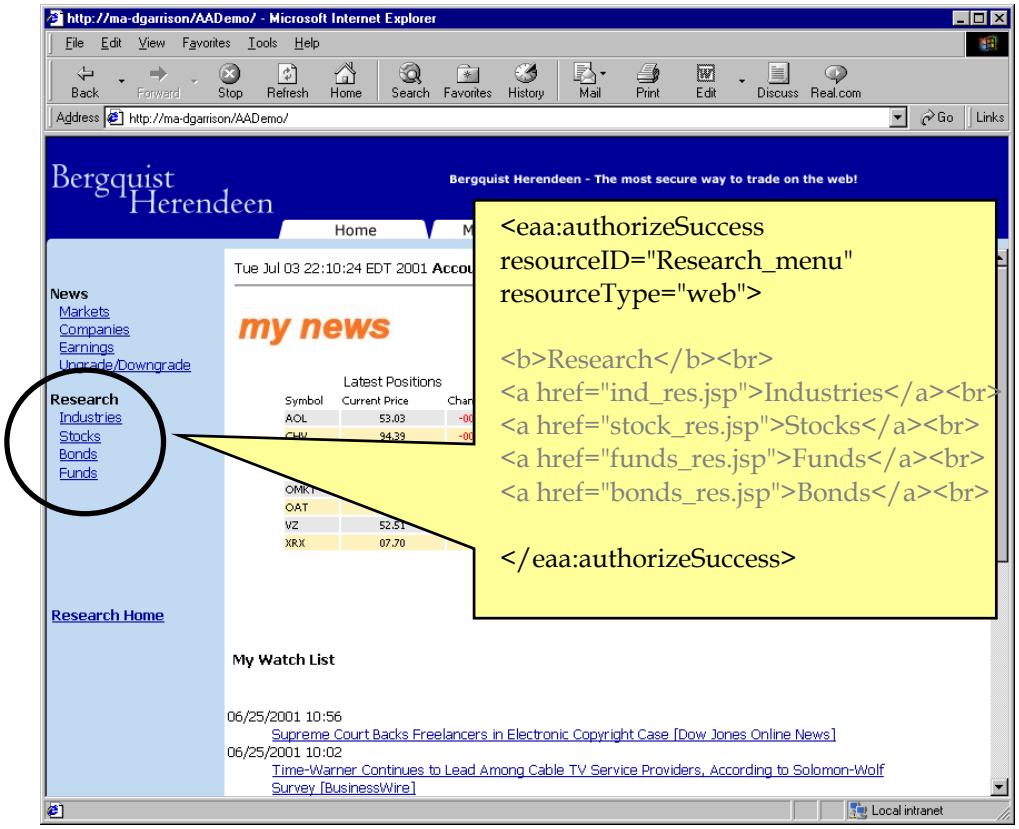


Figure 3: The AssureAccess tag library makes it easy to control access to individual page elements in a JSP

In this example, the developer wants to control access to a menu of links in a JSP. By using the AssureAccess tag library, the developer can allow access to certain customers and not to others. Access to the "Research_menu" resource is controlled by a policy set in the AssureAccess management console. The administrator can change that policy at any time without re-coding the application.

COM Object

The key features of AssureAccess are now available in Microsoft Windows environments, in the form of a COM object. This makes it easy for ASP applications, scripts, VisualBasic and C++ programs to take advantage of the powerful functionality of AssureAccess.

The COM object is only available on Windows NT and Windows 2000. The COM Object provides the following features in a convenient form:

- User login and logout
- Single Sign-on
- Check if access is allowed
- Get username or any other user attribute
- Write audit file if specified by policy
- Write log file
- Create and delete user accounts
- Change user passwords
- View and change user groups and other attributes

Audit

When configuring security in an application, monitoring user activity is often as important as controlling access. Audit records can be used to identify attacks on the system and provide a deterrence to insider abuse. Audit records can also be used as a form of receipting.

In the same way that Authorization is controlled by policy, the AssureAccess audit system is also policy based. Administrators filter the information sent to the Audit server through the same administrative GUI used to protect application resources. Additionally, the format and destination of audit records are fully configurable.

AssureAccess has separate audit and log channels. While the information sent to the audit channel is fully configurable, the system information sent to the log channel is fixed. Logging is limited to initialization activities to prevent runtime performance degradation.

Policy Based Audit

AssureAccess audit policies can use the same rules available during authorization decisions, including user attributes, time-based rule, authentication strength, client connection strength, and both client and server location. In addition, audit rules can test whether the request was allowed or denied. All audit policies are applied to both resources and user profiles through the Management Console.

Custom rules can be added to audit policies, allowing audit decisions to be based on real time information gathered from databases and legacy systems.

Auditing is event driven. Actions like login attempts and protected resource requests invoke the policy engine, allowing administrators to closely monitor the system. For example, failed login attempts and all logins after business hours are commonly audited.

Developers using any of the AssureAccess integrated solutions can add policy based auditing to applications, increasing the versatility of the audit subsystem.

Configurable Output

By default, the AssureAccess audit server writes text files containing XML audit records. The format of the files as well as their number, size, and location can be fully configured using the administrative utilities.

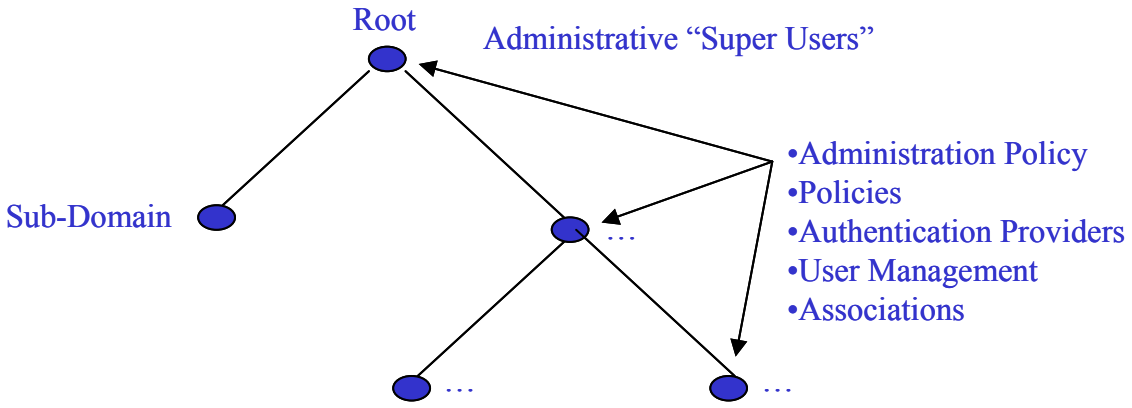
AssureAccess allows custom audit and log channels to be plugged into the audit server meaning that records can be written to other locations, like databases rather than to the filesystem.

Administration

Once a secure application is deployed, system administration becomes the primary interaction with the security. AssureAccess uses an intuitive administrative structure that can mimic the organizational structure and enable delegated administration. All runtime administration occurs through a Java Swing client application.

Administrative Paradigm

AssureAccess makes extensive use of hierarchical domains. Customers can create an unlimited number of domains, in a tree-structure of unlimited depth. Typically, the hierarchy would reflect the organizational structure of the enterprise and perhaps its geographic structure as well.



All of the elements of AssureAccess, such as policies, protected resources, user attributes, user repository connectors, and infrastructure servers can be defined at any level of the hierarchy. In general, elements defined in a domain are inherited downward into all the sub-domains below it. For example, an authorization policy defined in a domain corresponding to a division of a company would be applied to the sub-domains corresponding to the departments within that division. This would be in addition to policies defined at the department level.

When evaluating Authorization policies defined at multiple domain levels, access must be permitted at all levels for the resource to be served. Conversely, when evaluating Audit policies, if audit is required at any level, the event will be logged to the audit trail.

The domain hierarchy is a very powerful concept, which can be used in a variety of ways to meet the requirements of large organizations. Some examples include:

- Policies of various types can be defined at the appropriate level of the organization. For example, Audit policy might be set exclusively at the corporate level, while

Authentication policies are applied at several levels and Authorization policies are entirely under the control of the organization that owns the resource.

- By placing infrastructure servers, such as Authentication Servers, Audit Servers, and LDAP replicas in different domains, application servers can be directed to the nearest ones. This can optimize the use of network resources and maximize performance.
- Different aspects of administration can be delegated to individuals in different parts of the organization. This is critical for large-scale operations and can be used to implement functionality such as self-registration.

Administrative Tools

AssureAccess provides a Java Swing management console that streamlines the process of administering protected resources, policies, and configurations. It supports authenticating administrative users, displaying and managing domains and sub-domains, displaying and managing configurations and user repository connectors, and displaying and managing rules, policies, and protected resources.

Figure 4 below shows an example screen from the AssureAccess management console.

The left pane of the console provides a navigator that allows the user to browse the AssureAccess security domains. The accessibility of domain information is determined by administrative policy, which is used to delegate administration in the organization. Each domain contains folders for each of the five types of AssureAccess objects – protected resources, authentication profiles, policies, user repository connectors, and configurations.

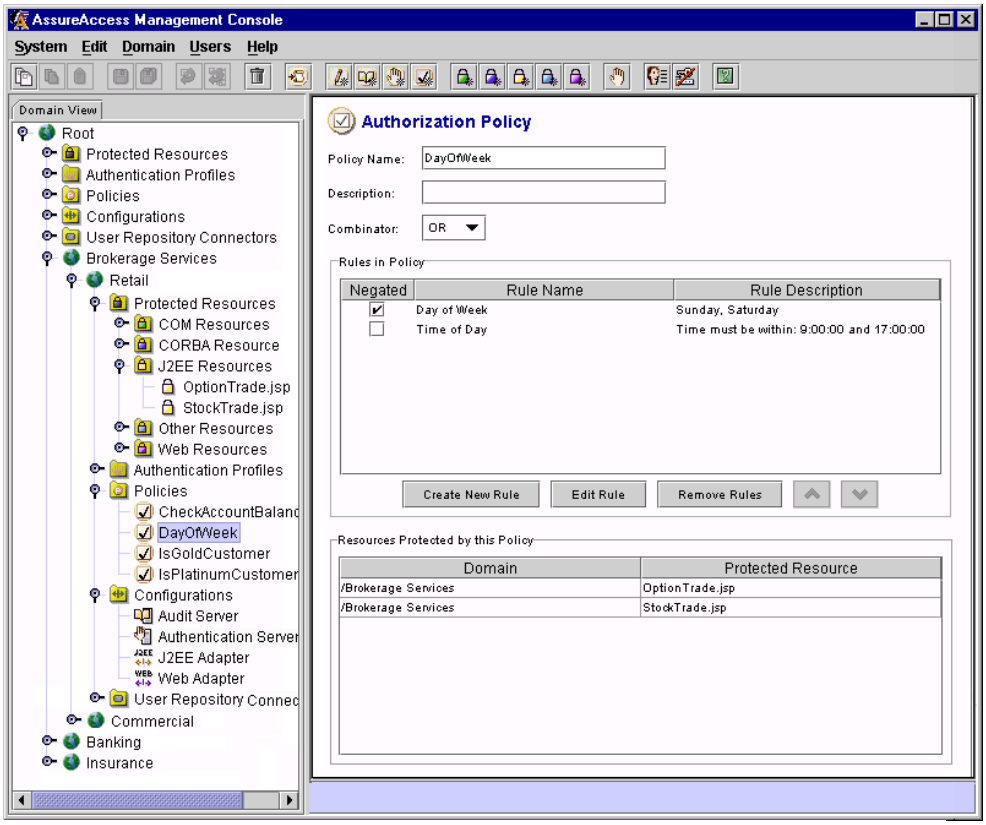


Figure 4: The AssureAccess management console makes it easy to configure single sign-on, configure user repository connectors, and create complex access policies

The right pane of the console contains the input form used to set parameters for each type of component. This screen is where policies are defined. The policy contains two rules and is used to ensure that stock and option trades are only executed during business hours.

Summary

AssureAccess redefines the standard for ease of use and ease of integration of access management capabilities into Web and Java based applications. The installation and configuration wizards make it easy to install and set up the product in even the most complex, distributed environments. The Java Swing management console makes the creation and maintenance of protected resources and access control policies simple. Forms based login and authentication strength provide an automated way of creating a multi-tiered authentication front end without writing a single line of code. The servlet filter provides Web-like bolt-on security for Web resources in the servlet container of the application server. The JSP tag library simplifies the process of implementing fine-grained access control of JSP page elements. The Universal Java Plug-in is a breakthrough technology that provides true bolt-on integration to all J2EE resources in the application and preserves user identity between the servlet container and the EJB container as well as between application servers – even if those servers come from different vendors.

About Entegrity Solutions

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

Entegrity Solutions Offices

West Coast:
2077 Gateway Place, Suite 200
San Jose, CA 95110
Tel: 408.487.8600 ext. 161

East Coast:
410 Amherst Street, Suite 150
Nashua, NH 03063
Tel: 603.882.1306 ext.2701

Mid-Atlantic:
10500 Little Patuxent Parkway, Suite 550
Columbia, MD 21044
Tel: 410.992.7600 ext. 3012

Europe:
Gainsborough House
58-60 Thames Street
Windsor
Berkshire SL4 1TX
United Kingdom
Tel: +44(0) 1753 272 072