IBM Tivoli Federated Identity Manager
Version 6.2.2.7

Installation, Configuration, and Administration Guide for Risk-Based Access

SC27-4445-02

IBM Tivoli Federated Identity Manager
Version 6.2.2.7

Installation, Configuration, and Administration Guide for Risk-Based Access

SC27-4445-02

Edition notice

Note: This edition applies to version 6, release 2, modification 2.7 of IBM Tivoli Federated Identity Manager (product number 5724-L73) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About this publication

Risk-based access is a pluggable and configurable component for IBM® Tivoli® Federated Identity Manager. Risk-based access provides access decision and enforcement that is based on a dynamic risk assessment or confidence level of a transaction.

The *IBM Tivoli Federated Identity Manager Installation, Configuration, and Administration Guide for Risk-Based Access* describes the procedures for installing, configuring, and administering risk-based access. It includes an overview of the process and functions of risk-based access and reference information for implementing risk-based access.

Access to publications and terminology

This section provides:

- A list of publications in the "IBM Tivoli Federated Identity Manager library."
- Links to "Online publications."
- A link to the "IBM Terminology website" on page x.

IBM Tivoli Federated Identity Manager library

The publications in the IBM Tivoli Federated Identity Manager library are:

- *IBM Tivoli Federated Identity Manager Quick Start Guide, CF38BML*
- *IBM Tivoli Federated Identity Manager Installation Guide, GC27-2718-01*
- *IBM Tivoli Federated Identity Manager Configuration Guide, GC27-2719-01*
- *IBM Tivoli Federated Identity Manager Administration Guide, SC23-6191-02*
- *IBM Tivoli Federated Identity Manager Installation, Configuration, and Administration Guide for Risk-Based Access, SC27-4445-00*
- *IBM Tivoli Federated Identity Manager Web Services Security Management Guide, GC32-0169-04*
- *IBM Tivoli Federated Identity Manager Auditing Guide, GC32-2287-04*
- *IBM Tivoli Federated Identity Manager Error Message Reference, GC32-2289-04*
- *IBM Tivoli Federated Identity Manager Troubleshooting Guide, GC27-2715-01*

Online publications

IBM posts product publications when the product is released and when the publications are updated at the following locations:

IBM Tivoli Federated Identity Manager Information Center
The http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tivoli.fim.doc_6.2.2/ic/ic-homepage.html site displays the information center welcome page for this product.

IBM Security Information Center
The http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp site displays an alphabetical list of and general information about all IBM Security product documentation.

IBM Publications Center

The http://www-05.ibm.com/e-business/linkweb/publications/servlet/pbi.wss site offers customized search functions to help you find all the IBM publications you need.

IBM Terminology website

The IBM Terminology website consolidates terminology for product libraries in one location. You can access the Terminology website at http://www.ibm.com/software/globalization/terminology.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully. With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

Technical training

For technical training information, see the following IBM Education website at http://www.ibm.com/software/tivoli/education.

Support information

IBM Support provides assistance with code-related problems and routine, short duration installation, or usage questions. You can directly access the IBM Software Support site at http://www.ibm.com/software/support/probsub.html.

Note: The **Community and Support** tab on the product information center can provide additional support resources.

Chapter 1. Roadmap for getting started with risk-based access

Before you begin to use risk-based access, familiarize yourself with key concepts, tasks, and requirements.

The following process describes how to understand and work with risk-based access.

Table 1. Roadmap for getting started with risk-based access

Task	For more information
Read the overview of risk-based access and its features.	See: • Chapter 2, "Overview of risk-based access," on page 3. • "Business scenarios" on page 3. • "Features" on page 4.
Study the architecture diagrams and information about risk-based access transactions.	See: • "Functional overview" on page 4. • "Transaction flow" on page 6.
Understand the concepts and process of risk score calculation.	See: • "Risk management" on page 23. • Risk score calculation.
Verify that you meet the software requirements.	See "Software requirements" on page 9.
Ensure that the prerequisites are fulfilled.	See "Installation Prerequisites" on page 10.
Install and deploy risk-based access.	See: • "Installing risk-based access" on page 12. • "Deploying risk-based access" on page 14.
Configure the risk-based access components, attributes, properties, and policies.	See "Configuration roadmap" on page 19.
Verify your installation and configuration.	See "Verifying the configuration" on page 21.
Use the risk-based access command-line interface to manage devices, policies, attributes, and properties.	See "Command reference" on page 63.

Chapter 2. Overview of risk-based access

Risk-based access provides access decision and enforcement that is based on a dynamic risk assessment or confidence level of a transaction. Risk-based access uses behavioral and contextual data analytics to calculate risk.

Risk-based access is a pluggable and configurable component for IBM Tivoli Federated Identity Manager.

Risk-based access:

- Improves security during authentication and authorization of business transactions.
- Assesses risk based on static, contextual, and analytically calculated attributes.
- Calculates a risk score based on multiple weighted attributes.
- Provides policy rules that determine whether an access request must be permitted, denied, or challenged.

You can configure risk-based access to:

- Silently register or require users to register devices that they commonly use.
- Associate the registered devices with user credentials.
- Present a challenge or request additional authentication, if the user attempts to authenticate with the same credentials from another unregistered device.
- Use the behavioral patterns of the user as a factor in risk score calculation. For example, a user might attempt to access a protected resource at a time outside of normal business hours. You can configure the risk-based access policy to deny access or force the user access to authenticate with a secondary challenge.

Business scenarios

Business transactions that have an increased security risk factor can benefit by implementing risk-based access.

The following examples are some scenarios where you can use risk-based access to provide a higher level of confidence for the transaction:

- A user tries to access sensitive information where a simple user ID and password authentication is not sufficient. However, the data is not sensitive enough to use a more complex authentication mechanism, such as token IDs.
- Users require access from remote locations that are not trusted and they use devices such as mobile devices and notebooks. To ensure that mobile users are authenticated sufficiently, the business requires a second factor authentication.
- Users need to access an application that provides sensitive business information. They might access the information outside of their regular work patterns.
- A user accesses a resource from a device that the user previously used and maintains typical usage patterns. Risk-based access improves the user experience by limiting secondary authentication mechanisms.

Features

Risk-based access provides several capabilities to identify potential risk and limit the ability for an attacker to use stolen credentials.

- Silent device registration where the system does not require any user interaction.
- Ready-to-use policy attributes that are specific to risk-based access.
- A risk-scoring engine that calculates a risk score for the current transaction. The risk score is based on configurable weights that are assigned to context attributes and behavior attributes. If the risk score is high, further challenges are presented to the user or access is denied. If the risk score is low, the user is permitted access.
- Command-line interface for configuring and managing the risk-based access policies and policy attributes. The commands are available through the WebSphere® Application Server AdminTask framework, which supports Jacl scripting, Jython scripting, and programmatic Java™ API.

Functional overview

Risk-based access includes an external authorization service (EAS), runtime authorization service, and attribute collection service.

The following diagram illustrates the architecture of risk-based access. The diagram also shows how the various components plug into WebSEAL in IBM Tivoli Access Manager for e-business and into IBM Tivoli Federated Identity Manager.

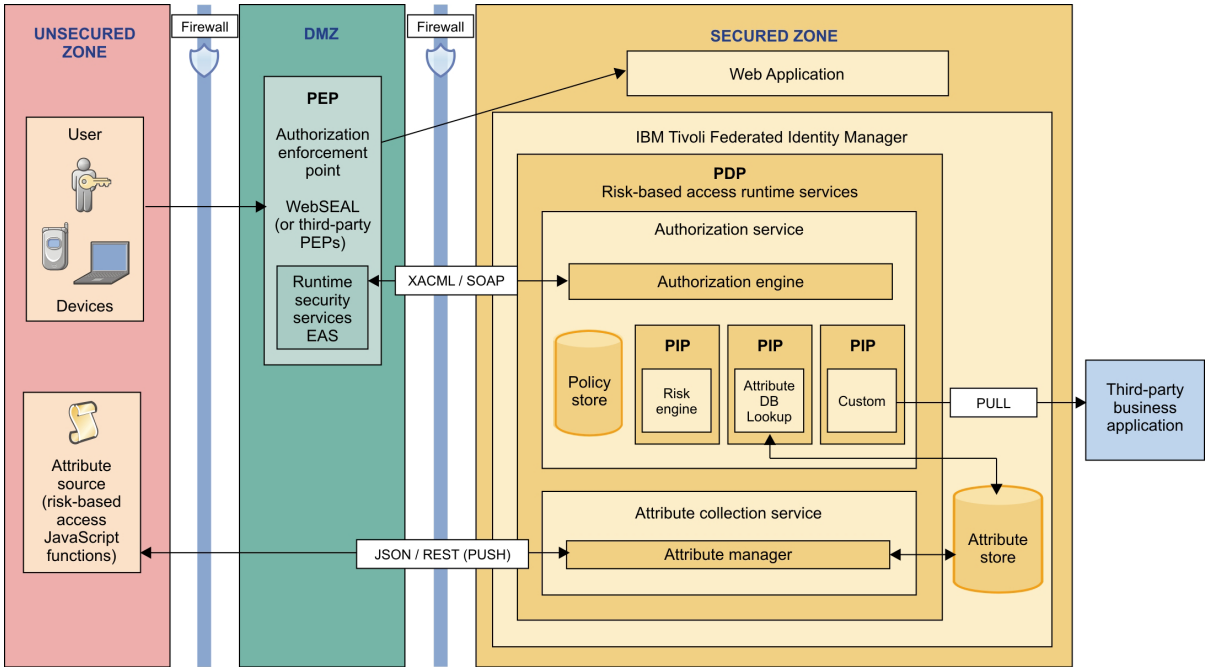

Figure 1. Risk-based access architecture

Risk-based access runtime services

Risk-based access provides the following runtime services:

Authorization service

The risk-based access authorization service is a component of the IBM Tivoli Federated Identity Manager runtime environment. The

runtime authorization service stores the policy, calculates the risk score, and makes the access decision. The authorization service exposes an XACML over SOAP web service that third-party enforcement points can call to get authorization decisions.

Attribute collection service
The attribute collection service is a Representational State Transfer (REST) service that collects web browser and location attributes from the user. The attribute collection service is a push service. You can configure the risk-based access runtime service to use the collected attributes as the policy attributes for calculating risk. You can also use the Java ADK to plug in your custom implementation for a pull service that retrieves attributes from the user.

Risk-scoring engine
The risk-scoring engine calculates the risk or confidence level. It provides a single integer that represents the risk score for the current transaction in the form of a percentage. The risk score is calculated based on the weights that are assigned to one or more of the following policy attributes:

- Device identification or fingerprint, such as details of hardware, IP address, location information, operating system, web browser type, web browser version, and screen resolution.
- Behavioral patterns, such as frequency of login, time of access, frequency of access, and type of transactions.
- Custom attributes that you can configure and manage through a pluggable interface. The risk-based access authorization service is extensible and can also include external sources for attributes.

The risk engine returns the final risk score as a policy attribute, which is the basis of the final authorization decision.

Policy enforcement point (PEP)
WebSEAL is the policy enforcement point for risk-based access. Risk-based access integrates with the existing WebSEAL authentication mechanisms, such as cross domain authentication service (CDAS) and external authentication interface (EAI). Risk-based access also integrates with both custom and IBM Business Partner implementations. Third-party enforcement points are supported. For example, a custom application can be used as an enforcement point.

External authorization service
The runtime security services EAS plug-in for WebSEAL enforces the policy decision. The EAS takes the request data and sends an authorization decision request to the risk-based access authorization service. The authorization service maps the authorization decision response to the appropriate WebSEAL action, such as permit, deny, or step-up authentication. You can manage the EAS with entries in the `webseald.conf` file with the WebSEAL stanza syntax.

Policy information points (PIPs)
Policy information points are components of the risk-based access authorization service. They provide all the policy attributes that are not provided in the initial access request. The risk score and attributes that are pushed to the attribute collection service are provided to the authorization service through PIPs.

You can configure PIPs with the runtime authorization service of IBM Tivoli Federated Identity Manager. The interface collects third-party policy attributes to provide data to the policy decision processor and other PIPs.

Risk-based access includes ready-to-use PIP implementations that provide the policy attributes that are required. You can also provide custom policy attributes to the authorization service through a custom PIP.

Policy decision point (PDP)
The runtime authorization service is the policy decision point for risk-based access. This service is configured to use the PEP risk-based access plug-in. The authorization decision is based on an authorization policy that uses policy attributes and PIPs that are specific to risk-based access. The PIPs provide information, such as risk score, user location, and device type.

The policy administration point (PAP)
IBM Tivoli Federated Identity Manager is the policy administration point for risk-based access. Risk-based access provides **`wsadmin`** commands for configuring and managing the policies, rules, attributes, and weights, which are required for calculating risk.

Note: If you are an existing user of IBM Tivoli Security Policy Manager, you can continue to use it to manage risk-based access policies.

Transaction flow

In a typical risk-based transaction, the user requests access to a protected resource. Risk-based access calculates the risk score and determines whether access is permitted, denied, or permitted with an obligation.

In the following example of a risk-based access transaction, the risk score indicates that the user must be presented with a challenge. The user successfully completes the challenge and receives permission to access the protected resource.

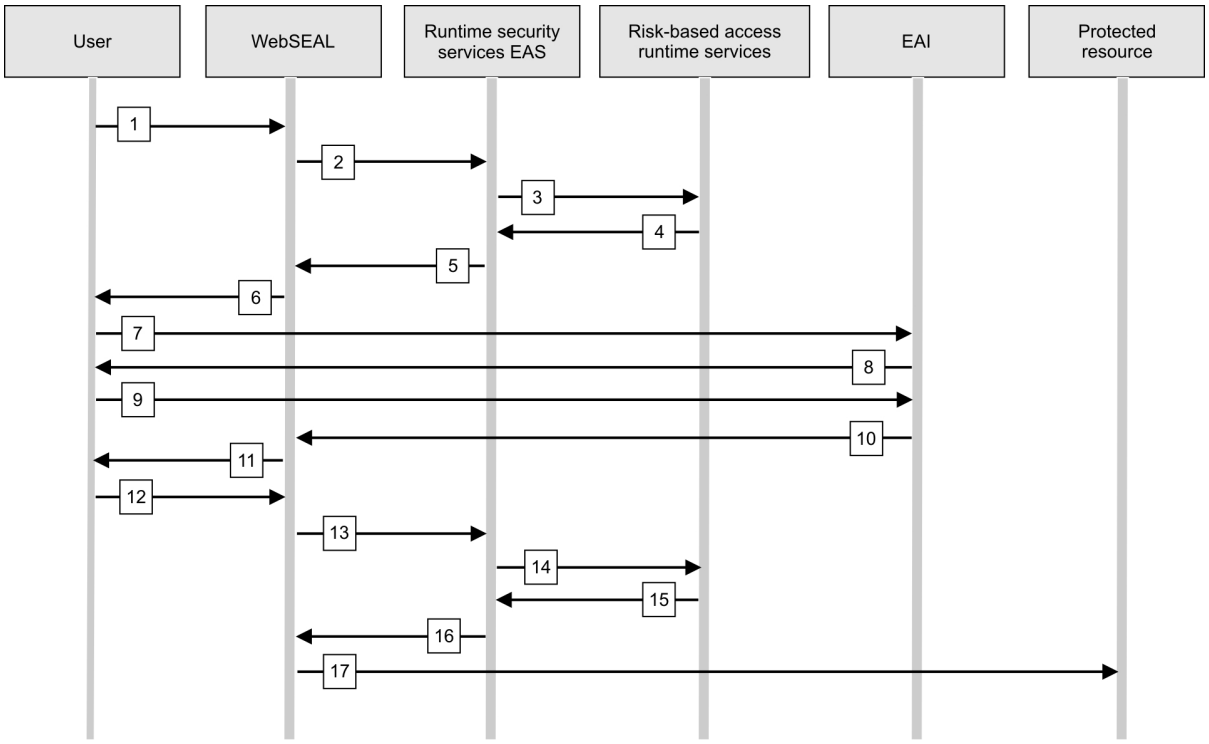

Figure 2. Risk-based access transaction flow example

The following process explains the flow of the transaction in the example scenario:

1. The user interacts through a web browser to submit authentication information and requests access to a protected resource. The protected resource is a junctioned web application that WebSEAL protects.
2. WebSEAL inspects the request.

   WebSEAL is the reverse proxy server that interacts with all transactions. For the protected resource that the user requests, the WebSEAL policy (POP) is configured to call the runtime security services EAS plug-in to authorize the request. The EAS is a shared library plug-in, which is internal to the WebSEAL process, so there is no on-the-wire callout between WebSEAL and the EAS.
3. The EAS first checks the local access manager policy.
   a. If the access manager denies access, then the EAS returns the response and does not continue with a forbidden response.
   b. If the access manager permits access, the EAS collects the context information about the user and the request. The WebSEAL **azn-decision-info** stanza has the specifications to create an XACML over SOAP authorization decision request.
   c. The EAS sends the request to the risk-based access runtime authorization service of IBM Tivoli Federated Identity Manager for the authorization decision.
4. The runtime authorization service (PDP) runs the configured policy and calls the appropriate PIPs based on the current policy.
   a. If a risk-score policy attribute is requested, then the risk engine is called.
   b. The risk engine takes the context details, requests additional policy attributes, if required, and then calculates a risk score.
   c. The authorization service applies the risk score policy attribute against the authorization policy and returns an appropriate decision response to the runtime security services EAS.
   d. The EAS supports three decision types: permit, deny, and permit with obligation. In this example, a decision to permit with obligation is returned.
      1) If the policy decision is a simple permit or deny, the decision is mapped to the WebSEAL `permit` and `not_permit` decisions. The WebSEAL decisions are returned from the **azn_svc_decision_access_allowed_ext** call.
      2) If the response is to permit access, WebSEAL permits the request to continue to the requested resource.
      3) If the response is to deny access, WebSEAL displays an error page, which indicates that access is forbidden.
5. The runtime security services EAS parses the response and returns one of the following EAS responses to WebSEAL: permit, deny, step-up authentication levels, or reauthenticate. In this example, the response is to step-up authentication levels, so that WebSEAL can enforce the appropriate authentication challenge.

   The risk-based access EAS also provides a configuration to map the returned obligation to a specific WebSEAL external authentication interface (EAI) mechanism. The EAI can be either a WebSEAL EAI or CDAS external authentication mechanism, which the customer or IBM Business Partner can implement.
6. WebSEAL specifies the appropriate authentication mechanism to the user.

7. The user is redirected to the authentication mechanism, which builds a challenge response.
8. The challenge response is presented to the user.
9. The user responds to the challenge.
10. If the response of the user to the challenge request is successful, the challenge is processed. The EAI application returns the credentials of the user and a successful step-up level in the HTTP response to WebSEAL.
11. WebSEAL updates the session of the user with the new credential details that the EAI application provides and redirects the user to the protected resource.
12. The request from the user to access the protected resource is sent via 302 redirect. The request does not require any user interaction.
13. WebSEAL inspects the request and directs the request to the runtime security services EAS for authorization.
14. The EAS collects all context information about the user and the request and creates an XACML over SOAP decision request. The EAS sends the request to the runtime authorization service.
15. The risk-based access runtime service takes the context and other policy attributes and calculates a risk score. The runtime service applies the risk score against the configured authorization policy and returns a policy decision response to permit access.
16. The runtime security services EAS interprets the response and returns the permit decision to WebSEAL.
17. WebSEAL permits the original request to continue without going back to the web browser of the user. The user is not aware of the transaction process.

Chapter 3. Installing

Use the IBM Update Installer for WebSphere Software to install risk-based access. After you install risk-based access, use the **manageRbaConfiguration** command to deploy the various components.

Software requirements

Before you install and configure risk-based access, you must ensure that your environment meets the requirements for operating systems, databases, and web browsers.

Supported operating systems for servers

AIX®

- AIX 6.1 POWER® System
- AIX 7.1 POWER System

Linux

- Red Hat Enterprise Linux Server Version 6, on x86-64
- Red Hat Enterprise Linux Version 5 Advanced Platform, on x86-64
- SUSE Linux Enterprise Server Version 10, on x86-64
- SUSE Linux Enterprise Server Version 11, on x86-64

Microsoft Windows

- Microsoft Windows Server 2008 Enterprise Edition, on x86-64
- Microsoft Windows Server 2008 R2 Enterprise Edition, on x86-64

Supported web browsers

- Mozilla Firefox 11.0 or later
- Google Chrome
- Microsoft Internet Explorer 8 or later

Note: If you plan to use location attributes for risk score calculation, you require Internet Explorer Version 9 or later, which supports the Geolocation API.

Supported databases

- IBM DB2®, Enterprise Server Edition, Version 9.7 with fix pack 2 or later
- IBM solidDB® Version 7.0

Installation Prerequisites

Before you install risk-based access, you must prepare your environment by installing and configuring prerequisite software.

Complete the following tasks before you install risk-based access.

Table 2. Checklist of installation prerequisites

Task	For more information
Download and install the latest version of the IBM Update Installer for WebSphere Software.	See the IBM support page http://www-01.ibm.com/support/docview.wss?uid=swg24020448.
Install WebSphere Application Server, Network Deployment, Version 7.0 with fix pack 15 or later. **Note:** If WebSphere Application Server is in a clustered environment, you must install WebSphere Application Server, Network Deployment, Version 7.0 with fix pack 23 or later.	See the WebSphere Application Server Version 7.0 Information Center at http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp. Search for *installing the product*.
Start the WebSphere Application Server if you did not already do so.	
Install IBM Tivoli Access Manager for e-business Version 6.1.1 with fix pack 6 or later and the WebSEAL component.	
Install IBM Tivoli Federated Identity Manager Version 6.2.2 with fix pack 1 or later.	See the IBM Tivoli Federated Identity Manager Version 6.2.2 Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tivoli.fim.doc_6.2.2/ic/ic-homepage.html.
Create a domain and deploy the runtime node for IBM Tivoli Federated Identity Manager.	See the *Configuration Guide* in the IBM Tivoli Federated Identity Manager Version 6.2.2 Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tivoli.fim.doc_6.2.2/ic/ic-homepage.html. Search for *creating and deploying a new domain*.
Decide if you want to do an automatic or manual setup of your database.	See "Database considerations." See also "Manually configuring the database" on page 11.

Database considerations

The risk-based access database stores the configuration properties, device fingerprints, and session attributes. You can choose between an automatic setup of an embedded solidDB database or manual setup of a supported database.

Risk-based access provides the following two options to set up the database:

Embedded solidDB database setup
This setup is a quick and easy solution that is ready for immediate use. It

does not require any manual installation or configuration of a database. However, this setup works only in a stand-alone WebSphere Application Server configuration.

Manual database setup
This setup is preferred for a production environment. It requires you to install and configure your database, and manually set up the database schema. This setup is possible regardless of whether you have a WebSphere Application Server stand-alone configuration or Network Deployment with clustered setups.

Using the embedded solidDB database

The embedded solidDB database setup does not require any manual installation or configuration of a database.

About this task

Note: This automatic setup of embedded solidDB database works only in a stand-alone WebSphere Application Server configuration.

Procedure

1. Before you deploy risk-based access, ensure that you *do not* create the JNDI context named `jdbc/rba` in WebSphere Application Server.
2. If the **`deploy`** operation does not detect any database that is installed and pre-configured with the JDNI name `jdbc/rba`, a solidDB database is automatically installed.
3. The database schema is also automatically created in the solidDB database. No further manual steps to run scripts to set up the schema are required.
4. Optional: After you run the **`deploy`** operation, it is suggested that you immediately change the embedded solidDB database password. Use the following **`manageRbaConfiguration`** command:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName dbpassword -propertyValue new-password}
```

What to do next

Install and deploy risk-based access.

Manually configuring the database

If you are not using the embedded solidDB database, you must install and configure the database and setup the schema. This manual database setup is preferred for a production environment.

Before you begin

Important: If you chose the automatic setup of the embedded solidDB database, you are *not required* to complete this procedure for manual setup.

Procedure

1. Install a supported database.
2. Create and configure a JNDI context named `jdbc/rba` in WebSphere Application Server. For more information, see the WebSphere Application Server Version 7.0 Information Center at http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp. Search for *configuring a data source*.

3. Set custom schema properties by completing the following steps:
   a. In the administrative console, click **Resources** > **JDBC** > **Data sources**.
   b. Click the name of the data source that was created in step 2 on page 11 to open the **Configuration** page.
   c. Click **Custom properties**.
   d. Click **currentSchema**.
   e. In the **Value** field, type `RBA_DB`.
   f. Click **OK**.
   g. Click **Save directly to the master configuration**.
4. Run the script to create the database schema for risk-based access.

   AIX® or Linux systems

   The scripts to create the database are in the `/opt/IBM/FIM/rba/dbscripts/` directory.
   a. Run the `create_schema.sh` shell script that is in the folder that corresponds to your database.
   b. When you run the script, specify the database user name, which is a required parameter. For example, for DB2:

      ```
      /opt/IBM/FIM/rba/dbscripts/db2/create_schema.sh database_user_name
      ```

   Windows systems

   The SQL files and the batch file to create the database schema are in the `C:\Program Files\IBM\FIM\rba\dbscripts\` directory.
   a. Edit the `create_schema.sql` file and replace `&DBUSER` with the database user name.
   b. Run the `create_schema.bat` batch file that is in the folder that corresponds to your database. Specify the `create_schema.sql` file as the input parameter. For example, for DB2:

      ```
      C:\Program Files\IBM\FIM\rba\dbscripts\db2\create_schema.bat create_schema.sql
      ```
5. When the SQL script starts executing, you are prompted for a password. Enter the password of the database user to proceed.

What to do next

Install and deploy risk-based access.

Installing risk-based access

Use the Update Installer for WebSphere Software to install the components and plug-ins for risk-based access.

Before you begin

Complete the installation prerequisites.

Procedure

1. Take one of the following actions, based on your operating system:

   AIX® or Linux systems
   Log in as root.

   Windows systems
   Log in as a member of the administrator group.

2. Download the 6.2.2.-TIV-ITFIM-FP0000-RBA.pak file from http://www.ibm.com/support/docview.wss?uid=swg24032796.
3. To start the Update Installer wizard, run the update.sh shell script or update.exe application file from the root directory of the Update Installer.

   For example:

   AIX® or Linux systems

   ```
   /opt/IBM/WebSphere/UpdateInstaller/update.sh
   ```

   Windows systems

   ```
   C:\Program Files\IBM\WebSphere\UpdateInstaller\update.exe
   ```

   Note: During the installation process, the Update Installer might display warning or information messages. Ignore the messages and proceed with the installation.
4. On the Welcome page, click **Next**.
5. In the **Directory Path** field, specify the IBM Tivoli Federated Identity Manager home directory and click **Next**.

   For example:

   AIX® or Linux systems

   ```
   /opt/IBM/FIM
   ```

   Windows systems

   ```
   C:\Program Files\IBM\FIM
   ```
6. Select **Install maintenance package** and click **Next**.
7. Specify the directory where you placed the 6.2.2-TIV-ITFIM-FP0000-RBA.pak file and click **Next**.
8. Select 6.2.2-TIV-ITFIM-FP0000-RBA.pak and click **Next**.
9. Select **Verify my permissions to perform the installation**, confirm the installation specifications, and click **Next**.
10. When the installation completes, click **Finish**.
11. Restart the application server or the deployment node in which you installed risk-based access. You must restart because the command-line interface extends the AdminTask framework of the WebSphere Application Server.

Results

Risk-based access is now installed.

What to do next

Deploy risk-based access..

Enabling language support for EAS messages

To enable language support for external authorization services (EAS) messages, set your locale in the operating system.

Before you begin

Install risk-based access.

About this task

Even though a different language locale is specified, the messages that are related to the runtime security services EAS are displayed in English. You can enable messages in a different language for the runtime security services EAS.

Procedure

1. Create the following directory if it does not exist:

   AIX or Linux operating systems
   `/opt/pdwebrte/nls`

   Windows operating systems
   `C:\Program Files\Tivoli\PDWeb\nls`
2. Extract the contents of *FIM_INSTALL_DIRECTORY*`/rba/eas/nls.zip` to that directory.
3. Set your locale in the operating system by updating the environment variables. For example, to enable Czech on a Linux operating system, set:
   - `LC_ALL=cs_CZ`
   - `LANG=cs_CZ`
4. Restart your workstation if it is required by your operating system.

What to do next

Verify that your EAS messages now display in the specified language.

Deploying risk-based access

After you install risk-based access, use the `deploy` operation of the **`manageRbaConfiguration`** command to set up the risk-based access database and deploy the various components.

Before you begin

1. Install risk-based access.
2. For the installation to take effect, restart the application server or the deployment node in which you installed risk-based access.
3. Decide if you want to do an automatic or manual setup of your database. See "Database considerations" on page 10.

About this task

The **`wsadmin`** commands for risk-based access are called by a Deployment Manager node in a managed environment with clusters.

If the `deploy` operation detects that the runtime security service is configured with IBM Tivoli Security Policy Manager, the existing instance of the runtime security service is not overwritten. In such a scenario, you can use IBM Tivoli Security Policy Manager as the policy administration point (PAP) for managing risk-based access policies. You cannot use the **`manageRbaPolicy`** command to manage polices. Some **`manageRbaPolicy`** operations are disabled in this scenario. The runtime security services of IBM Tivoli Federated Identity Manager and the runtime security services of IBM Tivoli Security Policy Manager must share the WebSphere Application Server profile for risk-based access.

Procedure

1. Open a command window and access the directory where your WebSphere Application Server profile is located. For example:

   AIX® or Linux systems
   ```
   /opt/IBM/WebSphere/AppServer70/profiles/AppSrv01/bin
   ```

   Windows systems:
   ```
   C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   ```

2. Start the **wsadmin** tool with one of the following commands:

   AIX® or Linux systems
   ```
   ./wsadmin.sh -username username -password password
   ```

   Windows systems:
   ```
   wsadmin.bat -username username -password password
   ```

3. To deploy the risk-based access runtime environment and plug-ins run the following **wsadmin** command:
   ```
   $AdminTask manageRbaConfiguration {-operation deploy}
   ```

   A message states that risk-based access is deployed successfully.

4. Enable security for the runtime security services.
   a. In WebSphere Application Server, create a group.
   b. On the WebSphere Application Server administrative console, select **Applications** > **Application Types** > **WebSphere enterprise applications** > **IBM Tivoli Runtime Security Services**.
   c. Under Detail properties, click **Security role to user/group mapping**. A list of roles is displayed.
   a. Select the **tscc-admin** role and click **Map Groups**.
   b. Select the group that you created and click **OK**.
   c. Click **OK** and save the configuration.
   d. Use the **manageRbaConfiguration** command to configure the `rtss.admin.basic.authn.username` and `rtss.admin.basic.authn.pwd` properties to match the user name and password in the group that you assigned to the **tscc-admin** role.
   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.basic.authn.username -propertyValue user_name}
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.basic.authn.pwd -propertyValue password}
   ```

5. Secure the RTSS service URL if these conditions apply to your IBM Tivoli Federated Identity Manager deployment:
   - RTSS is deployed on the server.
   - The product is junctioned behind WebSEAL.

   If these conditions apply, attach an access control list to `/FIM/rtss/admin` by using IBM Tivoli Access Manager for e-business, version 6.1.1 or later.

   Attaching an access control list ensures that the page is not available to everyone.

   See the IBM Tivoli Access Manager for e-business documentation.

Results

Risk-based access is deployed.

If no database is installed and configured with the JNDI context name, `jdbc/rba`, an embedded solidDB database is installed, and the schema is created.

What to do next

Configure the risk-based access attributes, policies, and runtime properties.

Uninstalling risk-based access

Use the **manageRbaConfiguration** command to remove the risk-based access components. After you remove the components, use the IBM® Update Installer for WebSphere Software to uninstall risk-based access.

About this task

If the `undeploy` operation detects that the runtime security service is configured with IBM Tivoli Security Policy Manager, then the instance of runtime security service is not removed or undeployed.

Procedure

1. To remove the runtime configuration files and plug-ins, run the following command:

   ```
   $AdminTask manageRbaConfiguration {-operation undeploy}
   ```

   A message states that risk-based access is undeployed successfully.

   Note: If you are using the embedded solidDB database, the **undeploy** operation creates a backup of the database at *FIM_install_dir*`/rba/solidDB/backup-`*timeStampInteger*`.`
2. To uninstall risk-based access, start the Update Installer wizard. Run the `update.sh` shell script or `update.exe` application file from the root directory of the Update Installer. For example:

   AIX® or Linux systems

   ```
   /opt/IBM/WebSphere/UpdateInstaller/update.sh
   ```

   Windows systems

   ```
   C:\Program Files\IBM\WebSphere\UpdateInstaller\update.exe
   ```
3. On the Welcome page, click**Next**
4. In the **Directory path** field, specify the directory where risk-based access is installed and click **Next**. For example:

   AIX® or Linux systems

   ```
   /opt/IBM/FIM
   ```

   Windows systems

   ```
   C:\Program Files\IBM\FIM
   ```
5. Select **Uninstall maintenance package** and click **Next**.
6. Select `RBAFP0000.pak` and click **Next**.
7. Review the summary information and click **Next**.
8. When the uninstallation completes, click **Finish**.
9. Manually delete the Java Archive (JAR) file that contains the command-line interface for risk-based access. The uninstallation wizard cannot remove this file until the application server is stopped. The file is locked because the application server is using the classes in the JAR file.

a. When the application server is stopped, go to the `plugins` directory in the WebSphere Application Server installation directory. For example:

   AIX® or Linux systems

   ```
   /opt/IBM/WebSphere/AppServer/plugins
   ```

   Windows systems

   ```
   C:\Program Files\IBM\WebSphere\AppServer\plugins
   ```

b. Delete the JAR file named `com.tivoli.am.rba.cli_7.0.0.jar`.

Known issues and workarounds

Risk-based access issues can include problems with retrieving or collecting attributes, connection errors, and performance issues. Use the workarounds to troubleshoot issues that you might encounter when you use risk-based access.

Geographic location attributes not retrieved

Problem

Location attributes are not retrieved from a device with a wired internet connection after you configure the attribute collection service to collect location attributes.

Workaround

To retrieve location attributes, ensure that the global positioning service (GPS) on the device of the user is enabled.

If the device does not have GPS, then ensure that the device is connected to a WiFi network to retrieve location attributes.

Connection not available

Problem

Errors that are similar to the following message are found in the `SystemOut.log` file: `Connection not available, timed out waiting for. . .`

Workaround

In WebSphere Application Server, increase the maximum connection pool size to a value that is greater than three times the expected maximum concurrent load.

For example, if you expect a maximum of 10 concurrent users at any point in time, set the maximum connection pool size as higher than 30.

Ensure that your maximum thread pool size is at least twice the maximum connection pool size.

For more information, see the WebSphere Application Server Version 7.0 Information Center at http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp. Search for *connection pool settings* and *thread pool settings*.

Attribute retrieval error

Problem

The `SystemOut.log` file contains errors that are related to the retrieval of risk-related attributes.

Workaround

In WebSphere Application Server Java virtual machine settings, increase your Java heap size.

Set the initial heap size to 1 GB and the maximum heap size to 2 GB.

For more information, see the WebSphere Application Server Version 7.0 Information Center at http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp. Search for *Java virtual machine settings*.

JavaScript file fails to load

Problem

The `info.js` fails to load.

Workaround

In WebSphere Application Server Java virtual machine settings, increase your Java heap size.

Set the initial heap size to 1 GB and the maximum heap size to 2 GB.

For more information, see the WebSphere Application Server Version 7.0 Information Center at http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp. Search for *Java virtual machine settings*.

Too many open files

Problem

WebSphere Application Server processes requests slowly. The error message `Too many open files` is displayed several times in the `SystemOut.log` file.

Workaround

Increase the limit for the number of open file descriptors.

In AIX and Linux systems, run the **`ulimit`** command:

```
ulimit -n nnnn
```

where *nnnn* is the required number of files, for example, 4096.

After you update **`ulimit`**, restart WebSphere Application Server from the same shell.

Risk-based access does not present a basic authentication challenge

Problem

When risk-based access does not present a basic authentication challenge, the external authorization service (EAS) and runtime security service experience a communication failure. Modifying the SSL settings solves this problem.

Workaround

1. Go to the WebSphere Application Server administrative console.
2. Select **SSL Certificate and Key Management** > **SSL Configurations** > **Node Default SSL Settings** > **Quality of Protection**.
3. Set **Client Authentication** to **Supported**.
4. Set **Protocol** to **SSL_TLS**.
5. Select **Global Security** > **Web Security – General Settings**.
6. Ensure that the following option is selected: **Default to basic authentication when certificate authentication for the HTTPS client fails**.

Chapter 4. Configuring

You must configure your standard authentication and authorization mechanisms to include risk-based access decisions. You can verify that your configuration is successful by accessing a resource that is protected by risk-based access. Confirm that your device is registered and the session attributes are captured for risk score calculation.

Configuration roadmap

Before you start to use risk-based access, you must configure the external authorization service (EAS), attribute collection service, the required attributes, properties, and policies.

After you install and deploy risk-based access, you must complete the following configuration tasks.

Table 3. Roadmap for configuring risk-based access

Task	For more information
Configure the WebSEAL runtime security services EAS.	See "Configuring the runtime security services external authorization service in WebSEAL" on page 27.
Configure the attribute collection service.	See "Configuring the attribute collection service" on page 34.
Configure the attributes that you require for risk score calculation.	See "Configuring the risk profile" on page 35.
Use the sample script to create a policy for risk-based access. The ready-to-use sample script in the authorization policy generation language provides basic rules for an end-to-end functioning policy. **Note:** If you are an existing user of IBM Tivoli Security Policy Manager, you can continue to use it as the policy administration point to manage risk-based access policies. For more information, see "Policy management with IBM Tivoli Security Policy Manager" on page 50.	See: • "Sample script for risk-based access policy rules" on page 48. • "Creating a risk-based access policy" on page 49.
Enable logs and traces.	See "Enabling logs and traces" on page 20.
Verify the configuration.	See "Verifying the configuration" on page 21.

Note: After you use the **manageRbaConfiguration** or **manageRbaRiskProfile** commands to configure properties or attributes, you must run the **manageRbaConfiguration** command with the **reload** option for the changes to take effect.

Enabling logs and traces

After you install risk-based access, you must enable the tracing and logging functions for risk-based access components in WebSEAL and IBM Tivoli Federated Identity Manager. Use the information in log files to isolate and debug problems.

Procedure

1. Enable logging for the runtime security services external authorization service (EAS) in WebSEAL.

   Use the **pdadmin** shell to enable trace and logs for the EAS.

   ```
   pdadmin> server task webseal_server_name
   trace set EAS_component_name 9 file path=log_file_path
   ```

   For example:

   ```
   pdadmin> server task default-webseald-epcloud127
   trace set pdweb.rtss 9 file path=/tmp/rtss.log
   ```

   Note: The setting is not persistent. The tracing is disabled when you restart WebSEAL.

   The log file is at the absolute file path of the location that you specified.

2. Use the WebSphere Application Server administrative console to enable logging for the IBM Tivoli Federated Identity Manager runtime security service.

   a. On the WebSphere Application Server console, select **Troubleshooting > Logs and Trace > *server_name***.

   b. Click the **Change Log Detail Levels** link.

   c. Under Additional properties, click **Change Log Detail Levels**.

   d. On the **Runtime** tab, add the following trace string:

   ```
   *=info:com.ibm.tscc.rtss.*=all:com.tivoli.am.rba.*=all:com.ibm.sec.authz.*=all
   ```

   e. Select **Save runtime changes to configuration as well**.

   f. Click **Apply** and save the changes.

   g. Restart WebSphere Application Server.

   You can find the log file at the following location: *WAS_HOME*/profiles/*profile_name*/logs/server1/trace.log.

3. To enable fine-grained logging of database operations, set the **db.logging.enabled** property to true.

   ```
   $AdminTask manageRbaConfiguration {-operation update
        -propertyName db.logging.enabled -propertyValue true}
   ```

   The WebSphere Application Server trace file starts logging every individual query along with connection information.

4. To troubleshoot issues related to risk calculation, set the **generate.risk.calculation.reports** property to true.

   ```
   $AdminTask manageRbaConfiguration {-operation update
        -propertyName generate.risk.calculation.reports
        -propertyValue true}
   ```

   Every time that risk is calculated, HTML risk reports are generated. The report files are stored in the temporary directory of the WebSphere Application Server Java virtual machine (JVM) in a folder named rba. The value of the **java.io.tmpdir** system property indicates the path of the JVM temporary directory. For example, on AIX® or Linux systems the files might be stored in the /tmp/rba directory.

What to do next

Verify the configuration of risk-based access.

Verifying the configuration

Verify that risk-based access services are running and devices are registered successfully. If the verification steps are successful, risk-based access is installed and configured correctly on your system.

Before you begin

- Install and deploy risk-based access.
- Configure the WebSEAL runtime security services EAS.
- Configure the attribute collection service.
- Configure the attributes that you require for risk score calculation.
- Create a policy and configure it for risk-based access.
- Enable logs and traces.

Procedure

1. Verify that the runtime security service is configured to load the risk-based access plug-in.

   In the *WAS_HOME*/profiles/*profile_name*/logs/server1/trace.log file, search for the following entries:

   ```
   AuthzRuntimeS 3   Successful registry finder : RBA AttributeSession PIP
   AuthzRuntimeS 3   Successful registry finder : USER FINGERPRINT COUNT PIP
   AuthzRuntimeS 3   Successful registry finder : RBA RiskCalculator PIP
   ```

   If the entries exist, the risk-based access plug-in is loaded successfully.

   If you do not see the entries, verify that the trace setting for the IBM Tivoli Federated Identity Manager runtime security service is correct. See "Enabling logs and traces" on page 20.
2. If you configured the ready-to-use sample policy, verify that the risk-based access policy is working.
   a. Visit the URL that you configured as a protected resource and log in. You are presented with a secondary challenge such as a one-time password (OTP). After you respond successfully with the secondary challenge, you would be given access to the resource. Your device fingerprint is registered in the risk-based access database and the risk score is initialized to zero. The runtime security services EAS logs would show the authorization decisions that are received from the policy decision point (PDP).
   b. Log out and log in again with the same user ID that you used in the previous step and from the same device. You would be given access to the resource without any secondary challenge.
   c. With the same user ID that you used in the previous step, access the same resource from a different device. If a secondary challenge is presented, the risk-based access step-up authentication is working. When the attributes do not match the stored fingerprint, a potential risk is identified and the user is presented with a challenge. Only after the user responds to the challenge successfully, the new device is registered. The runtime security services EAS logs would indicate that a request for a secondary challenge was received from the PDP.
3. Verify that your device fingerprint is registered. Run the following **wsadmin** command to list registered devices:

```
$AdminTask manageRbaDevices {-operation search}
```

4. Verify that your session attributes are collected. Run the following **wsadmin** command to list session attributes:

```
$AdminTask manageRbaSessions {-operation search}
```

Chapter 5. Administering

Manage the various components of risk-based access, such as the external authorization service (EAS), attribute collection service, session attributes, devices, and policies.

Risk management

Risk-based access policy decisions are based on the risk score. The risk score is calculated based on attributes that are retrieved from the user.

The following process describes the key aspects of risk management:

1. Decide on a set of attributes that you require for risk-score calculation.
2. Configure the methods to retrieve them.
   - You can retrieve many of these attributes from standard HTTP headers. Configure these attributes in the WebSEAL configuration file. See "Runtime security services external authorization service" on page 26.
   - For some attributes, you might require client-side JavaScript. To collect these special attributes, use the attribute collection service, which is a Representational State Transfer (REST) service. The attribute collection service provides read-to-use JavaScript, which captures the attributes that you configure. See "Attribute collection service" on page 32.
   - You might require other attributes that cannot be retrieved from standard HTTP headers or attributes that the attribute collection service does not capture. For such attributes, write custom JavaScript that provides attribute data to the attribute collection service. You can also use a SWF file for collecting information from web browsers that support the SWF format.
3. Configure and set weights for all attributes regardless of how they are retrieved. Use the **manageRbaRiskProfile** command to configure attributes. See "Configuring the risk profile" on page 35.
4. Specify how you want to store the attributes. See "Attribute storage" on page 37.
5. Configure the properties that specify settings for attribute matchers, such as location, IP address, and login time matchers. See "Attribute matchers" on page 38.

Risk score calculation

Risk score calculation is the process by which the risk engine determines a risk score. The risk score demonstrates the level of risk that is associated with permitting a request to access the resource. This risk score is compared to a threshold score that is set in a policy. A decision is made based on the result of this comparison.

Overview

The risk engine determines a risk score by comparing sets of attributes that identify devices. These sets of attributes are called device fingerprints. Device fingerprint attributes include items such as IP address, location, and screen size.

Each registered device has one device fingerprint. Because the user accesses the resource in different locations and on different devices, the user can have many registered devices.

The following process describes how risk assessment works:

1. The incoming device requests access to the resource.
2. The risk engine collects as many device fingerprint attributes as it can from the request device.
3. After the attributes are collected, the risk engine:
   - Determines the device fingerprint.
   - Calculates the risk score. The *risk score*
     - Is a number.
     - Represents the amount of risk that is associated with the incoming request.
     - Indicates the likelihood that the incoming request represents the user.
4. The risk engine:
   - Compares the incoming fingerprint with each registered device fingerprint.
   - Uses the attributes that are contained in the larger fingerprint for each comparison.
   - Calculates a risk score for each comparison.
5. To determine the final risk score, the risk engine:
   - Chooses the lowest risk score of the comparisons between the incoming fingerprint and the registered fingerprint.
   - Measures the final risk score against a threshold score or range that the administrator sets in a policy.
6. Depending on the way the administrator writes the policy, one of the following outcomes occurs:

   Permit
   The risk score for the incoming request is well below the threshold score. The user is granted access to the resource. For example, the risk score is `30`, and the threshold score that is set by the administrator is `40`.

   Permit with obligation
   The user is asked to complete an extra security measure, such as step up authentication. For example, the risk score is `40`, and the policy that the administrator wrote requires users that operate devices with scores `30 - 90` to step up.

   Deny The risk score for the incoming request is above the threshold score or range. The user is denied access to the resource. For example, the risk score is `50`, and the threshold score that is set by the administrator is `40`.

The risk score is calculated through the following formula:

Risk Score = (total weight of mismatched attributes / total weight of all attributes) × 100

When the values that belong to the incoming device fingerprint and the registered device fingerprint are the same, the values are matched. When the values that belong to the incoming device fingerprint and the registered device fingerprint are not the same, the values are mismatched.

Sometimes, the fingerprints contain attributes that are not matched or mismatched. These attributes are called indeterminate attributes. When there are indeterminate attributes present, the following formula is used to calculate the risk score:

Risk Score = (total weight of mismatched attributes /
(total weight of all attributes-total weight of indeterminate attributes)) × 100

Scenarios

The following example scenarios demonstrate risk score calculation.

Both scenarios assume that the administrator

- Wrote a policy that specifies that any risk score at or below 40 is permitted, and any risk score above 40 is denied.
- Gave equal weight values to all of the attributes in the tables.
  - The attributes in the tables have the same weight value of 10.

Scenario 1: Authentication is permitted

The example information in the table is used to calculate the risk score.

Attribute names	Weight values	Incoming device fingerprint values	Registered device fingerprint values
platform	10	Win32	Win32
screenHeight	10	1080	1080
screenWidth	10	1920	1920
colorDepth	10	32	32
userAgent	10	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:15.0) Gecko/20120427 Firefox/15.0a1	Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/28.0.1468.0 Safari/537.36
language	10	en-US	en-US
ipaddress	10	42.29.144.5	42.29.144.5

- All of the fingerprint attribute values match except for the incoming device fingerprint values and existing device fingerprint values for userAgent.
- Because userAgent is the only attribute that has any mismatched values, the total weight of the mismatched attributes is 10.
- The total weight of all of the attributes is 70 because each attribute has a weight value of 10.
- According to the risk score calculation formula: (10/70)×100=14. Therefore, the risk score is 14.
- Because the risk score is below 40, authentication is permitted.

Scenario 2: Authentication is denied

The example information in the table is used to calculate the risk score.

Attribute names	Weight values	Incoming device fingerprint values	Registered device fingerprint values
platform	10	Linux	Win32
screenHeight	10	1050	1080

The example information in the table is used to calculate the risk score.

Attribute names	Weight values	Incoming device fingerprint values	Registered device fingerprint values
screenWidth	10	1680	1920
colorDepth	10	24	32
userAgent	10	Mozilla/5.0 (X11; Linux i686 (x86_64)) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/27.0.1453.93 Safari/537.36	Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/28.0.1468.0 Safari/537.36
language	10	en-US	en-US
ipaddress	10	9.53.18.164	42.29.144.5

- None of the fingerprint attribute values match except for the incoming device fingerprint value and existing device fingerprint value for language.
- Because all of the attributes except for language have mismatched values, the collective weight of the mismatched attributes is 60.
- The total weight of all of the attributes is 70 because each attribute has a weight value of 10.
- According to the risk score calculation formula: (60/70)×100=86. Therefore, the risk score is 86.
- Because the risk score is above 40, authentication is denied.

Runtime security services external authorization service

The runtime security services external authorization service (EAS) for risk-based access is a modular authorization service plug-in. You can use IBM Tivoli Access Manager for e-business authorization as an add-on to your own authorization models when you have the EAS.

The runtime security services EAS provides the policy enforcement point (PEP) functionality for risk-based access. You can configure the runtime security services EAS to include risk-based access decisions as part of the standard authorization on WebSEAL requests. WebSEAL becomes the authorization enforcement point for access to resources that risk-based access protects.

The runtime security services EAS plug-in uses the IBM Tivoli Federated Identity Manager risk-based access capabilities. For more information about EAS, see the *Authorization C API Developer Reference Guide* in the IBM Tivoli Access Manager for e-business Version 6.1.1 Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc/welcome.htm. Search for *External authorization service plug-ins*.

The runtime security services EAS constructs a request that it sends to the policy decision point (PDP), which is IBM Tivoli Federated Identity Manager. Based on the policy decision that is received from the PDP, the EAS takes one of the following actions:

Table 4. Runtime security services EAS access decisions

Action	Description
Permit	Grants access to the protected resource.

Table 4. Runtime security services EAS access decisions (continued)

Action	Description
Permit with obligations	Grants access to the protected resource, after the user successfully authenticates with a secondary challenge.
Deny	Denies access to the protected resource.

You must configure the WebSEAL runtime security services EAS to enforce policy decisions that the PDP returns.

Configuring the runtime security services external authorization service in WebSEAL

Configure the runtime security services external authorization service (EAS) to enforce the policy decisions that the policy decision point (PDP) returns. Use the **tfimcfg** tool to update the WebSEAL configuration file.

Before you begin

- Install and deploy risk-based access.
- Ensure that IBM Tivoli Access Manager for e-business Version 6.1.1 or later is installed with the WebSEAL component. If version 6.1.1 is installed, ensure that fix pack 6 or later is applied.
- To run the `tfimcfg.jar` file, use **PDJrte** to configure the JRE, which starts the tool.
- To use the **tfimcfg** tool with IBM Security Access Manager for Web Version 7.0 or later, you must meet the following conditions:
  - Obtain an IBM JRE, version 6.0, Update 10 or later.
  - Configure the `com.ibm.security.cmskeystore.CMSProvider` provider in the `java.security` file, which is in `$JAVA_HOME/lib/security`, of the IBM JRE.
  - Use **PDJrte** to configure the JRE, which starts the tool, into the Security Access Manager domain.
  - The **ikeyman** tool in the `$JAVA_HOME/bin` must be on the path before you run the tool.
- Run the tool on the same computer where the WebSEAL instance is configured.
- If you are using WebSEAL Version 6.1.1, copy the runtime security services EAS shared library from the risk-based access installation directory to the WebSEAL installation directory.

  For example:

  AIX systems

  Copy the `/opt/IBM/FIM/rba/eas/6.1.1/`*platform_name*`/librtsseas.a` file to the `/opt/pdwebrte/lib` directory.

  Linux systems

  Copy the `/opt/IBM/FIM/rba/eas/6.1.1/`*platform_name*`/librtsseas.so` file to the `/opt/pdwebrte/lib` directory.

  Windows systems

  Copy the `C:\Program Files\IBM\FIM\rba\eas\6.1.1\WIN\rtsseas.dll` file to the `C:\Program Files\Tivoli\PDWebrte\lib` directory.

About this task

The **tfimcfg** tool ensures that the correct data is passed to the runtime security services EAS for each request.

To include risk-based access decisions as part of the standard authorization on WebSEAL requests, use the **tfimcfg** tool. Then, manually update the configuration file and attach the POP. This process is described in the following procedure.

Procedure

1. (Optional) Copy the issuer certificate from the WebSphere Application Server instance that hosts the runtime security service to a key database file. This step is part of the server-side SSL authentication configuration.

   Note the location of this file so that you can specify it for the **Optional runtime security services SSL keyfile** within the **tfimcfg** tool in step 2.

2. Run the **tfimcfg** tool.

   Use the following **tfimcfg** attributes:

   - For a WebSEAL server:

     ```
     java -jar tfimcfg.jar -action tamconfig -cfgfile WebSEAL_filename
     ```

   - For a Web Gateway Appliance server:

     ```
     java -jar tfimcfg.jar -action wgaconfig -cfgurl Web_Gateway_Appliance_URL
     ```

   See the "**tfimcfg** reference" for information about the tool parameters in the Federated Identity Manager Information Center.

   If you are using the JRE provided with WebSphere Application Server, version 8.0 or later, you might encounter an error when you use **tfimcfg**. See the Fix Pack 6 readme file for information about more parameters **tfimcfg** requires.

   While you are running this tool, note the following risk-based access specific prompts:

   a. For the **Federation to configure** prompt, select **Risk-based access**.

      Note: The tool configures the same ACLs for federations and risk-based access. Also, the `itfim_rba_anyauth` ACL is attached to the attribute collection service.

   b. Provide the following runtime security services parameters:

      - Runtime security services user
      - Runtime security services password
      - Runtime security services URL
      - Optional runtime security services SSL keyfile
      - Optional runtime security services SSL stash file
      - Optional runtime security services SSL keyfile label

      The **tfimcfg** tool attempts to connect to the runtime security services server. If the connection is invalid, you are prompted to enter the parameters again.

      If you want to configure server-side SSL authentication, specify the SSL parameters.

3. Attach the POP, **rba-pop**, to the resource that risk-based access must protect. For example, if `MyJct` is the resource to protect, run the following command:

   ```
   #pdadmin -a sec_master
   Enter password: passw0rd
   pop attach /WebSEAL/localhost-default/MyJct rba-pop
   server replicate
   quit
   ```

   Note: Attach the POP in the object tree such that the runtime security services EAS is used only for the resources that are protected by risk-based access. If you attach the risk-based POP to a root object in the object space, the POP is

used for every request, including administrative tasks. Most of these calls to the runtime security services EAS are not required and might result in lowered performance throughput.

4. Update the WebSEAL configuration file to append the `[obligations-levels-mapping]` stanza. Specify the mapping between obligation levels that the PDP returns and the authentication levels in WebSEAL. For example, append:

   ```
   [obligations-levels-mapping]
   otp=3
   consent=4
   ```

   See the *WebSEAL Administration Guide* in the IBM Tivoli Access Manager for e-business Version 6.1.1 Information Center. Search for "specifying authentication levels".

5. Verify that each header that was configured by the **tfimcfg** tool under the `[azn-decision-info]` stanza has a corresponding attribute configuration in the risk-based access database. The values that are specified after the equal sign (=) in the entries must match the attributes that you configure with the **manageRbaRiskProfile** command.

   Ensure that the case of the attribute matches the corresponding entry under the `[azn-decision-info]` stanza because the entries are case-sensitive. The Subject-UUID entry is an exception. You do not have to configure **cookie:ac.uuid** as an attribute with the **manageRbaRiskProfile** command. The **Subject-UUID** entry is required only for the attribute collection service to work with the runtime security services EAS.

   See the *WebSEAL Administration Guide* in the IBM Tivoli Access Manager for e-business Version 6.1.1 Information Center. Search for "`azn-decision-info`".

Results

- The POP, **rba-pop**, is created.
- The `itfim_rba_anyauth` ACL is attached to the attribute collection service.
- The runtime security services EAS is configured.

What to do next

See "Configuring the attribute collection service" on page 34.

Sample configuration data for runtime security services external authorization service

The sample **[rtss-eas]** stanza contains the configuration details for the runtime security services external authorization service (EAS) to enforce policy decisions. Add the sample entries under the **[rtss-eas]** stanza in the default WebSEAL configuration file named `webseald-default.conf`.

Note: The formatting, commenting, and line breaks in the following code might change when you copy and paste from a PDF file. Compare the code that you copied and pasted with the following code to ensure that it conforms to the correct syntax.

```
[rtss-eas]
# Specify the name of the IBM Tivoli Access Manager for e-business
# trace component that the EAS uses

trace-component = pdweb.rtss

# Set this property to true if you want the EAS
# to first check with IBM Tivoli Access Manager for e-business
# whether the user has permission to access the resource based
# on the ACL set.
```

```
apply-tam-native-policy = true

# Specify the context-id that is used in the requests that are
# sent by the EAS to the runtime security service.
# If the context-id parameter is not set, the WebSEAL server-name is used
# as the default value.
#
#context-id =
#
# Specify the audit logging configuration. This entry consists
# of an agent identifier that is followed by a comma-separated list
# of key-value pairs of attributes that are associated with the agent.
#
# For example, to configure the auditing of records to a file:
# audit-log-cfg = file path=/tmp/rtss-audit.log,flush=20,
#      rollover=2000000,buffer_size=8192,queue_size=48
# To send audit logs to STDOUT:
# audit-log-cfg = STDOUT
#
# If this attribute is missing or not configured, no audit
# events are logged.

# audit-log-cfg =

# Specify the name of the runtime security services SOAP cluster
# that contains this runtime security services SOAP service.
# Also specify a corresponding [rtss-cluster:cluster]
# stanza with the definition of the cluster.

[rtss-cluster:cluster1]
# Specify the definitions for a cluster of runtime security services
# SOAP servers in this stanza.

# Define the specifications of the server that you use to communicate
# with a single runtime security services SOAP server,
# which is a member of this cluster.
# Values for this entry are defined as:
#       {[0-9],}URL
# where the first digit (if present) represents the priority of the server
# in the cluster (9 being the highest, 0 being lowest). A priority of 9 is assumed
# if you do not specify a priority.  The URL can be any
# well-formed HTTP or HTTPS URL.

# You can specify multiple server entries for failover and load balancing
# purposes.  The complete set of these server entries defines the
# membership of the cluster for failover and load balancing.

server = 9,https://localhost:9443/rtss/authz/services/AuthzService

# Specify the maximum number of cached handles that are used when
# communicating with runtime security services SOAP.

handle-pool-size = 10

# Specify the length of time, in seconds, before an idle handle is removed
# from the handle pool cache.

handle-idle-timeout = 240

# Specify the length of time, in seconds, to wait for a response from
# runtime security services SOAP.

timeout = 240

# You can use the following optional configuration entries if
# the runtime security services SOAP server is configured to require
```

```
# basic authentication. If you leave these entries blank,
# the basic authentication header is not provided when communicating
# with the runtime security services SOAP server.

# Specify the name of the user for the basic authentication header.

basic-auth-user =

# Specify the password for the basic authentication header.
# Note: To obfuscate the the value of a stanza/key,
# use the following pdadmin command:
# pdadmin> config modify keyvalue set -obfuscate
#          config-file stanza key value
# For example:
# pdadmin> config modify keyvalue set -obfuscate
#          /opt/pdweb/etc/webseald-default.conf
#          rtss-eas basic-auth-passwd passw0rd
# For more information about obfuscation, see Command Reference in
# the IBM Tivoli Access Manager for e-business information center.
# Search for config modify in the pdadmin commands section.

basic-auth-passwd =

# The following SSL entries are optional and are only required if:
#    1. At least one server entry indicates that SSL is to be used,
#       that is, it starts with https:)
#    2. A certificate other than what is used by this server is required
#       when communicating with the policy server. The details of the
#       default certificate can be found in the [ssl] stanza of this
#       configuration file.
# If these entries are required and are not found under this stanza,
# the default [ssl] stanza is searched.

# The name of the key database file that houses the client certificate
# that must be used.

# ssl-keyfile =

# The name of the password stash file for the key database file.

# ssl-keyfile-stash =

# The label of the client certificate in the key database.

# ssl-keyfile-label =

# This configuration entry specifies the distinguished name (DN) of the
# server (obtained from the server SSL certificate) that is accepted.
# If no entry is configured all DNs are considered as valid.
# Multiple DNs can be specified by including multiple
# configuration entries of this name.

# ssl-valid-server-dn =


# This entry controls whether Federal Information Processing
# Standard (FIPS) communication is enabled. If no configuration entry
# is present, the global FIPS setting (as determined by
# the IBM Tivoli Access Manager policy server) takes effect.

# ssl-fips-enabled =


# Define the mappings between the obligation levels that the policy decision
# point (PDP) returns and the WebSEAL step-up authentication levels.
# The mapping must be one-to-one and the user must be permitted to authenticate
# only through the appropriate obligation mechanisms. These entries ensure that the
```

```
# EAS maps the obligations to the authentication levels and vice versa correctly.

[obligations-levels-mapping]
otp=3
consent=4
```

Session configuration

You can capture the session attributes of a user with the attribute collection service. Risk-based access uses these static and contextual attributes to calculate the risk score.

Attribute collection service

The attribute collection service is a Representational State Transfer (REST) service. It can collect web browser and location information from the user for calculating the risk score.

Process overview

The following process describes the attribute collection service and how to use it:

1. Make REST calls to store and delete attributes in the database. The initial request to the service receives a correlation ID. The correlation ID is used to make further REST calls.
2. Use JavaScript to collect the web browser attributes. You can place the HTML page that calls the JavaScript functions on any server.
   - Ajax collects information in the background. It does not slow down page loading.
   - You can make standard Ajax requests only to the same domain. With Cross Origin Resource Sharing (CORS), you can make Ajax requests across domains.
   - The CORS response header contains the settings for the following specifications:
     – The server from which requests are accepted.
     – The types of requests that are accepted.
3. Configure the attributes that you want to collect for risk score calculation. They are collected and listed in *FIM_install_dir*/pages/C/ac/info.js.

Request types

GET and Post requests create a correlation ID to identify the session in the database. A correlation ID is a UUID that is stored in a cookie. The attribute collection service process uses the following request types:

GET Retrieves information about an attribute session from the database. GET requests are disabled by default. Requests use a URL with a REST path, such as https://webseal/FIM/sps/ac/rest/*UUID*.

POST Creates an attribute session in the database. POST requests use a URL such as https://webseal/FIM/sps/ac/*UUID*.

The session attributes are sent as a JSON string with the request. In a response, the server sets a cookie that contains the correlation ID.

For example, the POST /sps/ac/9d37e806-24cf-4398-a3b9-d7f13fb2231f request creates a session in the database with the UUID of 9d37e806-24cf-4398-a3b9-d7f13fb2231f.

You can also configure the risk-based access properties to use an existing cookie.

DELETE Deletes an attribute session from the database.

Risk-based access runtime properties

Use the **manageRbaConfiguration** command to configure the risk-based access properties that are required for attribute collection service.

The following properties specify information about the server from which requests are received and the location of the attribute collection service:

ac.request.server
The server from which requests are received.

For example: `http://mywebsealhost.company.com`.

ac.service.location
The location of the attribute collection service.

For example: `http://mywebsealhost.company.com/`*`webseal-junction-name`*`.`

session.timeout
A numeric value that represents the number of seconds before a risk-based access session automatically expires unless the session is updated.

If any attribute in the session is updated, the expiry time for the session is extended by the number of seconds specified in this property. The default value is `3600`.

This value must be greater than or equal to the WebSEAL maximum session lifetime. If not, collected device attributes expire, and the RBA policies fail.

The following configuration properties are used by the attribute collection service to access the `GET` method of the REST service:

ac.get.attributes.enabled
Indicates whether access to the `GET` method is allowed.

The default value is `false`.

ac.get.attributes.allowed.clients
A comma-separated list of IP addresses or host names that can access the `GET` method.

If this property is not set and `ac.get.attributes.enabled` is set to `true`, anyone can access the `GET` method.

If this property is set but `ac.get.attributes.enabled` is set to `false`, then this property is ignored.

Use the **manageRbaConfiguration** command to set these properties and their values in the risk-based access runtime configuration file. For more information, see "manageRbaConfiguration command" on page 63.

JavaScript functions

Use the JavaScript functions in the *FIM_install_dir*`/pages/C/ac/info.js` file to make requests to the server. Include the `info.js` file in the HTML login page of your application. When `info.js` is loaded, it calls the following functions:

sendSession()

Makes a POST request to the delegate service.

The sendSession() function collects the web browser attributes and sends them to the server. They are stored in the database. Call this function when a user logs in.

deleteSession()

Makes a DELETE request for a specified correlation ID.

The POST request from the sendSession() returns a correlation ID. Based on the correlation ID, the deleteSession() function deletes the attributes from the database. Call this function when the user logs out or when the current session times out.

getLocation()

Detects the location of the device from which the requests are made. If the location information is sent to the server, call the getLocation() function before the sendSession() function. The following web browsers support the detection of location: Mozilla Firefox, Google Chrome, Opera, Apple Safari, and Microsoft Internet Explorer 9.

Note: For the JavaScript functions to work in Microsoft Internet Explorer, include the following statement in the HTML page from which you call the function. The following statement forces Microsoft Internet Explorer to use the standards mode:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
     "http://www.w3.org/TR/html4/loose.dtd">
```

For configuration steps and examples, see "Configuring the attribute collection service."

Configuring the attribute collection service

Before you can collect risk calculation information, you must specify the server and location of the collection service. You also must specify a JavaScript file to collect the session attributes.

Before you begin

Run the **tfimcfg** tool to configure the runtime security services EAS. See "Configuring the runtime security services external authorization service in WebSEAL" on page 27.

Procedure

1. To publish pages and plug-ins and load the Tivoli Federated Identity Manager runtime environment, run the following command:

   ```
   $AdminTask manageRbaConfiguration {-operation reload}
   ```

2. Create and configure the risk-based access properties for the attribute collection service.
   a. Create and configure **ac.request.server** to specify the location of the attribute collection service. Run the following command and replace the variable in *italics* with your values:

      ```
      $AdminTask manageRbaConfiguration {-operation create
           -propertyName ac.request.server
           -propertyValue request_server_url}
      ```

Note: The value of *request_server url* is a space-separated list of hosts from which requests are permitted. Host names must begin with `http://` or `https://`.

The following command shows a sample value for the property:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ac.request.server
     -propertyValue http://mywebsealhost.company.com}
```

b. Create and configure **`ac.service.location`** to specify the location of the attribute collection service. Run the following command and replace the variable in *italics* with your values:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ac.service.location
     -propertyValue https://host_name/webseal-junction-name}
```

The following command shows a sample value for the property:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ac.service.location
     -propertyValue https://mywebsealhost.company.com/FIM}
```

3. Add the URL of `info.js` to the `<head>` block in the HTML landing page of your application. The `info.js` file calls functions that are required to collect session attributes. Follow this format:

```
<script src="https://host_name/webseal-junction-name/sps/ac/js/info.js"></script>
```

Note: When the `info.js` file is included on an HTML page, attribute collection by Ajax calls can take time to complete. To avoid issues, attribute collection must end before you move away from the page. For example, if the attribute collection is still running, and a user name and password are entered, the policy fails to resolve session attributes. To prevent this issue, modify the JavaScript to prevent the user from continuing until after the Ajax call completes.

Results

The basic configuration of the attribute collection service for risk-based access is complete. For more information, see "Attribute collection service" on page 32.

What to do next

Configure attributes and the weights that you require for risk score calculation. For more information, see "manageRbaRiskProfile command" on page 73.

Risk profile configuration

You must create risk attributes and specify their weights. You can also configure the ready-to-use attribute matchers or write your own custom attribute matcher.

Configuring the risk profile

You can retrieve attributes from standard HTTP headers or use the attribute collection service to collect attributes. To use these attributes for risk score calculation, you must set appropriate weights for each attribute.

Before you begin

Complete the following configuration steps:

- "Configuring the runtime security services external authorization service in WebSEAL" on page 27
- "Configuring the attribute collection service" on page 34

About this task

Use the **manageRbaRiskProfile** command to create the attributes and configure weights for the attributes that you require for risk score calculation.

Procedure

1. Create the attributes and specify weights for the attributes that you configured in the WebSEAL configuration file named `webseald-default.conf` for the runtime security services external authorization service (EAS). The attributes that you configure must be a subset of the entries that you added in the **[azn-decision-info]** stanza.

   Use the following examples to configure the attributes that are captured from standard HTTP headers:

   ```
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId header:user-agent
        -weight 75}
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId header:content-type
        -weight 50}
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId header:accept-charset
        -weight 80}
   ```

2. Create the attributes that you want the attribute collection service to retrieve and specify weights for the attributes.

   Use the following examples to configure the attributes and weights:

   ```
   $AdminTask manageRbaRiskProfile {-operation create -attributeId language
        -weight 95}
   $AdminTask manageRbaRiskProfile {-operation create -attributeId platform
        -weight 60}
   $AdminTask manageRbaRiskProfile {-operation create -attributeId plugins
        -weight 25}
   $AdminTask manageRbaRiskProfile {-operation create -attributeId fonts
        -weight 80}
   ```

   Note: If you configure the **fonts** attribute, you must also add the `fcollector.swf` SWF object in the login page of your application, for example, the WebSEAL `login.html` page. Add the following object element in the body element of your login page at the end, just before the `</BODY>` end tag. Replace the *variables* in italics with your values:

   ```
   <object width="1" height="1">
       <param name="movie" value="https://host_name/webseal_junction_name/sps/ac/js/fcollector.swf">
       <embed src="https://host_name/webseal_junction_name/sps/ac/js/fcollector.swf" width="1" height="1">
       </embed>
   </object>
   ```

   For example:

   ```
   <object width="1" height="1">
       <param name="movie" value="https://myservicehost.company.com/FIM/sps/ac/js/fcollector.swf">
       <embed src="https://myservicehost.company.com/FIM/sps/ac/js/fcollector.swf" width="1" height="1">
       </embed>
   </object>
   ```

3. Optional: To enable risk-based access based on location, IP address or login time, you can configure the following additional attributes. These attributes are required for ready-to-use matchers that are provided with risk-based access. For more information, see "Attribute matchers" on page 38.

   Use the following examples to configure the attributes for the specified attribute matchers:

Location matcher

```
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId longitude -weight 75}
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId latitude -weight 75}
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId accuracy -weight 75}
```

IP address matcher

```
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId ipaddress -weight 50}
```

Login time matcher

```
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId accessTime -weight 45 –behavior true}
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId urn:oasis:names:tc:xacml:1.0:resource:resource-id
     -weight 0 –behavior true}
$AdminTask manageRbaRiskProfile {-operation create
     -attributeId cookie:ac.uuid -weight 0 –behavior true}
```

What to do next

1. Configure the specific properties that are required for the following attribute matchers:
   - IP address matcher
   - Location matcher
   - Login time matcher
2. Create and configure a risk-based access policy. Use one of the following methods:
   - Specify the policy rules in a script with the authorization policy generation language. The ready-to-use sample script in the authorization policy generation language provides basic rules for an end-to-end functioning policy. Use the script to create a risk-based access policy and configure it for use with risk-based access.
   - If you are an existing user of IBM Tivoli Security Policy Manager, you can continue to use it as the policy administration point to manage risk-based access policies. For more information, see “Policy management with IBM Tivoli Security Policy Manager” on page 50.

Attribute storage

Risk-based access uses the device fingerprint, behavior, and session attributes of the user to calculate the risk score. These attributes are persisted or stored in the risk-based access database.

Device fingerprint data
: Consists of attributes that are stored when a device is registered. The incoming device fingerprint is compared against this stored repository of trusted device fingerprints.

Session data
: Consists of the session attributes of the user that are stored temporarily until the session times out. However, if the device is registered, the session attributes are also stored as part of the device fingerprint.

Behavior data
: Is historic data that is stored in the database and used for behavior-based attribute matching. For example, the login timestamps of the user over the previous three months.

When you configure attributes with the **manageRbaRiskProfile** command, you can also specify how each attribute must be stored in the risk-based access database. Use the **-device**, **-session**, and **-behavior** parameters to store the attributes as device fingerprint, session, or behavior attributes.

For example, if you want the `browser` attribute to be stored as behavior data, run the following command:

```
$AdminTask manageRbaRiskProfile {-operation create -attributeId browser
     -weight 75 –behavior true}
```

For more information, see "manageRbaRiskProfile command" on page 73.

Attribute matchers

An attribute matcher compares the values of a specified attribute in the incoming device fingerprint with the existing device fingerprint of the user. Risk-based access uses the information returned by the attribute matchers to calculate the risk score.

The exact matcher is the default attribute matcher. It is used if you do not configure any attribute matcher. It checks whether the values of a single attribute are the same in both the incoming and existing device fingerprints.

In some scenarios, multiple attributes or a set of composite attributes must be matched. For example, longitude, latitude, and accuracy are three attributes that are related to location. Assume that two device fingerprints are considered a match if the distance between two location points is not greater than a specified threshold value. In this scenario, the comparison of only the longitude attribute does not provide accurate results. The matcher must do a more complex comparison or composite matching, where it matches multiple attributes from both fingerprints.

The matcher either returns a match or a mismatch result. A mismatch increases the risk score based on the weight assigned to the attributes.

The matcher might abstain from being a part of the risk calculation in the following situations:

- The incoming device fingerprint does not contain the required attributes.
- The historical data is not available for a matcher to make a match or mismatch decision.

Risk-based access provides ready-to-use attribute matchers that compare composite attributes or analyze a range of attribute values. You can configure one or more of the attribute matchers that are described in the following sections.

Configuring the location matcher:

The location matcher checks whether the location of a device is within a specific distance from the previous known locations of the device. Configure the location matcher properties to specify the accuracy range and how to compare the location information.

About this task

Procedure

1. Add the `longitude`, `latitude`, and `accuracy` attributes to the fingerprint configuration and specify a numeric value for the weights that you require. Use the following syntax for the **manageRbaRiskProfile** command:

   ```
   $AdminTask manageRbaRiskProfile {-operation create -attributeId unique_ID
        -weight weight}
   ```

   Limitation: The retrieval of location attributes depends on the web browser and the settings that the user specifies in the web browser. The web browser must support the Geolocation API. Also, in some web browsers, an error might occur if a user tries to access a protected resource from a device with a wired internet connection.

   Note: The location-based analysis processes all three location attributes (longitude, latitude, and accuracy) collectively when it determines the match for the location. Though weights are assigned to all three attributes, the weight for only the longitude attribute is considered. The weights assigned to the supporting latitude and accuracy attributes are ignored.

2. Configure the following risk-based access properties for the location matcher:

   location.comparison

   Indicates how you want the attribute matcher to calculate the accuracy range of the location coordinates.

   Set the property to one of the following values:

   - Specify the value as `closest` to calculate the distance between the closest points on the accuracy range of two locations. This calculation is the most restrictive calculation.
   - Specify the value as `midpoint` to calculate the distance between the midpoints of the circles without considering accuracy.
   - Specify the value as `farthest` to calculate the distance between the farthest points on the accuracy ranges of the two locations. This calculation is the least restrictive calculation.

   The following figure illustrates the closest points, midpoints, and farthest points of the accuracy ranges of two locations. In this figure, the circle represents the accuracy range and the center of the circle represents the location.

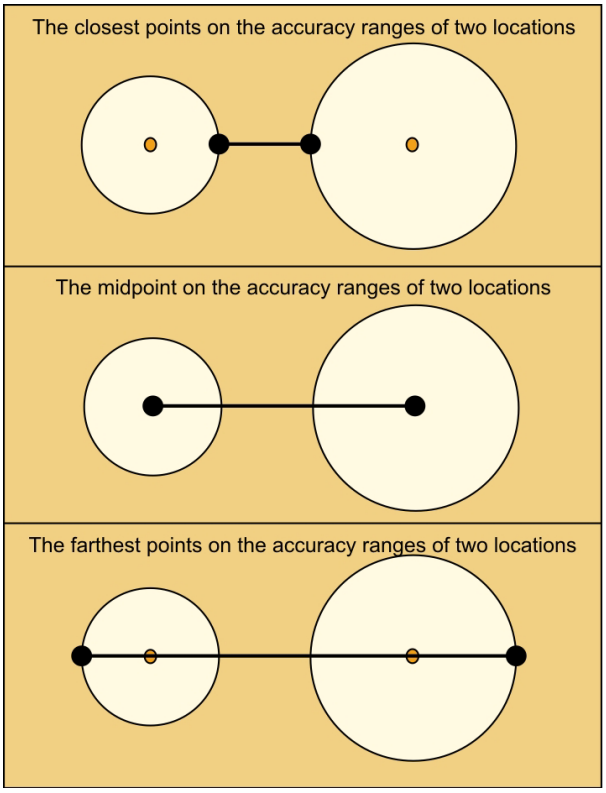

Figure 3. The closest points, midpoints, and farthest points on the accuracy ranges of two locations

location.allowable.distance
The maximum distance between the new location and the historic locations. The unit of the numeric value is kilometers.

The default value is 40.

Use the following syntax for the **manageRbaConfiguration** command to configure these properties:

```
$AdminTask manageRbaConfiguration {-operation create -propertyName property_name
     -propertyValue property_value}
```

Example

The following example shows how to configure the risk-based access properties for the location matcher:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName location.comparison -propertyValue midpoint}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName location.allowable.distance -propertyValue 100}
```

Configuring the IP address matcher:

Configure properties that specify IP address lists so that you can use IP addresses as attribute matchers to calculate the risk score.

About this task

The IP address matcher compares the IP address of a request with a white list (inclusion list) or black list (exclusion list) of IP addresses or with the historical IP addresses of the device.

Procedure

1. Add the `ipaddress` attribute to the fingerprint configuration and specify a numeric value for the weight that you require. Use the following syntax for the **manageRbaRiskProfile** command:

   ```
   $AdminTask manageRbaRiskProfile {-operation create -attributeId unique_ID
        -weight weight}
   ```

2. Configure the following risk-based access properties for the IP address matcher:

   ip.whitelist
   A list of comma-separated IP addresses or subnets that you want to permit. Include `X.X.X.X` as a value for `-propertyValue` to compare the incoming IP address with the IP address with which the device is registered.

   ip.whitelist.netmask
   A list of comma-separated netmasks for the IP addresses that you listed in the `ip.whitelist` property. Specify the list of netmasks in the same order that you specified the list of IP addresses. The total number of netmasks must correspond exactly to the total number of IP addresses.

   ip.blacklist
   A list of comma-separated IP addresses that you want do not want to permit.

   ip.blacklist.netmask
   A list of comma-separated netmasks for the IP addresses that you listed in the `ip.blacklist` property. Specify the list of netmasks in the same order that you specified the list of IP addresses. The total number of netmasks must correspond exactly to the total number of IP addresses.

   Use the following syntax for the **manageRbaConfiguration** command to configure these properties:

   ```
   $AdminTask manageRbaConfiguration {-operation create -propertyName property_name
        -propertyValue property_value}
   ```

Example

Use these examples to configure the risk-based access properties for the IP address matcher.

The following configuration permits requests that are made from any IP address in the 9.0.0.0 subnet, except for IP addresses that begin with 9.5:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.whitelist -propertyValue 9.0.0.0}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.whitelist.netmask -propertyValue 255.0.0.0}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.blacklist -propertyValue 9.5.0.0}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.blacklist.netmask -propertyValue 255.255.0.0}
```

To compare the incoming IP address with previous IP addresses of the device, add an address with X for the numbers in the `ip.whitelist` and `ip.whitelist.netmask` values.

The following configuration permits requests that are made from the 9.0.0.0 subnet, except for requests from the 9.5.0.0 subnet. The configuration also permits requests from IP addresses whose first three numbers are the same as the historical device fingerprints.

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.whitelist
     -propertyValue 9.0.0.0, X.X.X.X}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.whitelist.netmask
     -propertyValue 255.0.0.0, 255.255.255.0}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.blacklist
     -propertyValue 9.5.0.0}
$AdminTask manageRbaConfiguration {-operation create
     -propertyName ip.blacklist.netmask
     -propertyValue 255.255.0.0}
```

Configuring the login time matcher:

Use the login time matcher to compare and analyze historical login time data of the user with the current login time of the user. You can use the data as input for risk score calculation.

About this task

The login time matcher primarily detects the logins per session that is based on the attribute called **accessTime**.

The first of the several access times that are captured within the session is considered the login time of the user. The result of the analysis determines the probability of a fraudulent user.

Procedure

1. Configure the following attributes as behavior attributes and specify their weights by using the **manageRbaRiskProfile** command:
   - `accessTime` attribute with the weight that you require.
   - `urn:oasis:names:tc:xacml:1.0:resource:resource-id` attribute with `0` as its weight.
   - `cookie:ac.uuid` attribute with `0` as its weight.

   For example:

   ```
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId accessTime -weight weight –behavior true}
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId urn:oasis:names:tc:xacml:1.0:resource:resource-id
        -weight 0 –behavior true}
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId cookie:ac.uuid -weight 0 –behavior true}
   ```

   Note: The `cookie:ac.uuid` property enables the login time matcher to be aware of session boundaries for the access times that are captured. If you specified a value other than `ac.uuid` for the `ac.uuid` configuration property, then you must use that value as the attribute name instead of `cookie:ac.uuid`. For example, if you set `ac.uuid` to `ac_cookie_name`, then run the following command:

   ```
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId cookie:ac_cookie_name -weight 0 –behavior true}
   ```

2. Configure the login probability value:

   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName login.time.probability.threshold
        -propertyValue property_value}
   ```

Where *property_value* specifies the login probability as a numeric value from `0` to `1`. The default value is `.3`.

This value indicates how probable it is for a user to log in within an hour of the previous login times. With a lower value, the odds of the matcher returning true are higher, and the risk score is lower. With a higher value, the odds of the matcher returning true are lower, and risk score is higher. For example, if you set a value of `.5`, the matcher almost always returns false.

3. Configure the number of times a user must log in before the matcher provides a result:

```
$AdminTask manageRbaConfiguration {-operation create
     -propertyName min.usage.history.required
     -propertyValue property_value}
```

Where *property_value* specifies the number of times the same user must login before the matcher provides input for risk score calculation. The default value is `8`. The default value means the login time analysis collects data for eight login times by the same user before it provides input for risk score calculation.

Until this value is met, the login time matcher returns an `INDETERMINATE` result, and its weight is considered as zero.

4. Load the login time matcher changes:

```
$AdminTask manageRbaConfiguration {-operation reload}
```

Implementing custom attribute matchers:

Create custom JavaScript to specify attributes and processing instructions that the JavaScriptMatcher must follow instead of the default attribute matcher processing. The results are included in the risk score calculation.

Before you begin

Custom matching works only with attributes that are stored as session attributes. For example, to include a device attribute, you must first store it as a session attribute. See “Attribute storage” on page 37.

About this task

You can use multiple JavaScript files for custom matchers. However, do not use multiple scripts to process the same attribute.

An example script is provided for assistance.

You can reference the following parameters in the custom file:

`IFingerprint source`
The existing device fingerprint from the database. Use with device attributes.

`IFingerprint target`
The device fingerprint from the current login session. Use with session attributes.

`IUsageData[] history`
An array that contains all of the historical attributes that are stored in the database for the user. Use with behavior attributes.

`String attributeName`
The name of the attribute to match.

boolean processed
Indicates whether the JavaScript processed the current attribute. Set `processed` to `true` in the custom script.

string matched
The result that is returned by the matcher. Valid values are `MATCHED`, `MISMATCHED`, and `INDETERMINATE`.

Procedure

1. Create a JavaScript file to identify the attributes and processing actions that you want to complete according to your instructions. Follow these guidelines when you write the JavaScript file:
   a. Import the following packages:
      - `Packages.com.tivoli.am.rba.fingerprinting`
      - `Packages.com.tivoli.am.rba.extensions`
   b. Identify the name of the attribute by including the following code:
      ```
      if(attributeName.equals("attribute_name"))
      ```
      where *attribute_name* is the attribute that you want to create a custom match specification.
   c. Identify the value of each attribute by including one of the following lines of code:

      For session attributes:
      ```
      if(target.getValue(attributeName).equals("attribute_value"))
      ```

      For device attributes:
      ```
      if(source.getValue(attributeName).equals("attribute_value"))
      ```

      For behavior attributes:
      ```
      if(history[0].getValue(attributeName).equals("attribute_value"))
      ```
      where *0* is any index in the array of IUsageData objects.
   d. Set each attribute to `processed = true` to ensure the JavaScriptMatcher uses your specifications instead of the default processing.
   e. Set the `matched` property to indicate your attribute match preference. Valid values are `MATCHED`, `MISMATCHED`, and `INDETERMINATE`.
2. Save the JavaScript file in the following directory:
   ```
   WAS_PROFILE_HOME/config/itfim/rba/javascript/matchers/
   ```
3. Run the following command to enter the attributes that are identified in the custom JavaScript file into the `javascript.attributes` runtime property:
   ```
   $AdminTask manageRbaConfiguration {-operation create -propertyName javascript.attributes
   -propertyValue attribute_name,attribute_name}
   ```
   where *attribute_name* is the name of the attribute.

   Note: Separate multiple attributes by comma. For example, if you want to check language and color, enter the following commands:
   ```
   $AdminTask manageRbaConfiguration {-operation create
   -propertyName javascript.attributes -propertyValue language,color}
   ```
   This step ensures that the processor uses the custom instructions for identified attributes instead of the default processing.
4. Load the property changes:
   ```
   $AdminTask manageRbaConfiguration {-operation reload}
   ```

Results

Attributes that are identified in the custom file and set to `processed = true` are now processed according to your specifications.

After a JavaScript file successfully processes a specified attribute and returns a result, the remaining JavaScript files are not evaluated for that attribute. Because the processing sequence of the files is not defined, you might not be able to identify the JavaScript that processed the attribute.

Example

A sample JavaScript file is in the following directory:

/WAS_PROFILE_HOME/config/itfim/rba/javascript/matchers/language.js

This code checks if your language is United States English. If it is, a value of "MATCHED" is returned. If it is not, a value of "MISMATCHED" is returned.

```
importPackage(Packages.com.tivoli.am.rba.fingerprinting);
importPackage(Packages.com.tivoli.am.rba.extensions);

if(attributeName.equals("language")) {
processed = true;
if(target.getValue(attributeName).equals("en-US")) {
matched = "MATCHED";
}
else {
matched = "MISMATCHED";
}
}
```

Configuring the hash algorithm for attribute storage

Hashing encodes a character string as a fixed-length bit string for comparison. Risk-based access hashes certain attributes by default. You can change the hash algorithm and specify additional attributes that you want to hash.

About this task

By default, when attributes are stored in the risk-based access database, the attributes that exceed the maximum length according to the schema are hashed. You can also specify any other attribute that you require to be hashed. For example, you might want to hash values that are considered confidential or private.

The default hash algorithm that risk-based access uses for storing these attributes is SHA256. You can specify any other hash algorithm that Java Security supports.

Procedure

1. Use the **manageRbaConfiguration** command to specify the name of the hash algorithm that you want to use.

   For example:

   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName attributes.hash.algorithm
        -propertyValue SHA256}
   ```

2. Specify the additional attributes for which you want to enable hashing. Use the **manageRbaConfiguration** command to specify a comma-separated list of attributes.

   For example:

```
$AdminTask {manageRbaConfiguration -operation create
     -propertyName attributes.hash.enabled
     -propertyValue header:user-agent,fonts,plugins}
```

Policy management

Use the authorization policy generation language to author policy rules. You can create and configure an authorization policy based on these rules for risk-based access.

Note: If you are an existing user of IBM Tivoli Security Policy Manager, you can continue to use it as the policy administration point to manage risk-based access policies. You can either create a risk-based access policy with IBM Tivoli Security Policy Manager or import the existing risk-based access policies.

Authorization policy generation language

The authorization policy generation language is a simplified format that helps you specify the policy rules in a script. You can use this script to create and configure an authorization policy for risk-based access.

Use any text editor to create the script. The script consists of the following main parts:

- "Policy"
- "Rule"
- "Variables" on page 47
- "Logic" on page 47

See "Sample script for risk-based access policy rules" on page 48.

Policy

Every authorization policy has an outermost `Policy` block where you must specify the properties and rules of the policy. Use the `Policy` block in the authorization policy generation language to create a policy set in the policy file that contains one or more policies.

Specify the following properties in the `Policy` block:

Name The name of the policy.

In an IBM Tivoli Access Manager for e-business environment that uses WebSEAL, the policy name must match the server name specified in the WebSEAL configuration file. For example, `localhost-default`.

Rule

Define the policy rules in the `Rule` block, which is in the `Policy` block. The `Rule` block must contain the following properties:

Name The name of the rule.

Target The subject, action, and resource for the scope of the rule.

Subject

The users, groups, or roles to which the rule applies.

To specify that the rule applies to anyone who tries to log in, use the value `any`.

To specify users, groups, or roles, use the following comma-separated list format:

```
Subject = {user: username1, username2, . . .
           group: groupname1, groupname2, . . .
           role: rolename1, rolename2, . . .
          }
```

The subject block must contain at least one user, group, or role.

Action

The actions that are permitted for the resources to which the rule applies.

Specify the value as `any`, a specific action, or a comma-separated list of actions.

Resource

One or more resources to which the rule applies.

Specify the value as `any`, a specific resource, or a comma-separated list of multiple resources.

Variables

Variables are defined just before the logic of the rules.

Use the following syntax for defining a variable:

```
resource|action|subject|environment integer|double|date|time variable_name
```

A variable has three parts:

- The part of the request from where the variable comes. Valid values are `resource`, `action`, `subject`, and `environment`.
- The data type of the variable. Valid values are `integer`, `double`, `date`, and `time`.
- The name of the variable. Valid values are alphanumeric characters and underscore. The variable name must begin with an alphabet or an underscore.

For example, `action integer Risk` declares an integer named `Risk` found in the action part of the request.

Logic

After you define the name, target, and variables, you must define the logic for the rules as a series of `if-else` statements.

Each `if` and `else` statement corresponds to a separate rule.

The following table lists the logical operators that you can use in the authorization policy generation language.

Table 5. List of logical operators

Logical operators	**Symbol**
And	&
Or	|
Comparison operators	**Symbol**
Greater than	>
Lesser than	<

Table 5. List of logical operators (continued)

Logical operators	Symbol
Greater than or equal to	>=
Lesser than or equal to	<=
Equal to	==
Not equal to	!=

When you compare two values, ensure that they are of the same data type, otherwise an error occurs.

The results of the `if-else` statements can be one of the following decisions:

permit
The request must be permitted to pass.

deny The request must be denied and not permitted to pass.

permit-with-obligation
The user is required to do something before the request is permitted to pass.

In the obligation block, declare a name that is a Uniform Resource Locator (URL) or Uniform Resource Name (URN). The URL or URN must specify what must be done for the request to be permitted.

Optional: You can also define a series of attributes. Each attribute must have a name, type, and value, which are returned with the obligation. The name and type must be a URL or URN.

The following example shows the format of an obligation declaration:

```
permit obligation {
     name=urn:oasis:names:tc:xacml:example:obligation:email
     attribute {
          name=xyz
          type=http://www.w3.org/2001/XMLSchema#string
          value=qwerty
     }
}
```

After you write the script with policy rules, you can use the script to create a risk-based access policy.

Sample script for risk-based access policy rules

Write a script in the authorization policy generation language and use the script to generate a risk-based access policy. The ready-to-use sample script defines the basic rules that are required for an end-to-end functioning policy.

The sample script has the following policy rules:

- If there are no devices registered for the user and the authentication level is 3, for example, when a user logs in for the first time:
  - Register the device silently.
  - Permit the user to access the resource after successfully responding to a secondary challenge.
- If there are one or more registered devices for the user and the authentication level is 3, permit access to the resource.
- If the risk score is equal to or lesser than 40, permit access to the resource.

- If the risk score is greater than 40, permit access to the resource after the user responds successfully to the secondary challenge.

```
Policy {
    name = localhost-default

    Rule {
        name = RBADemoApplicationPolicyRule1

        Target {
            subject = any
            action = any
            resource = any
        }

        resource integer http://security.ibm.com/attributes/resource/riskScore
        subject integer http://security.ibm.com/attributes/subject/registeredDeviceCount
        subject string  AUTHENTICATION_LEVEL

        if http://security.ibm.com/attributes/subject/registeredDeviceCount == 0 & AUTHENTICATION_LEVEL == "3"
            permit obligation {
                name=http://security.ibm.com/obligations/registerDevice
            }
        if http://security.ibm.com/attributes/subject/registeredDeviceCount >= 1 & AUTHENTICATION_LEVEL == "3"
            permit
        if http://security.ibm.com/attributes/resource/riskScore <= 40 & AUTHENTICATION_LEVEL == "1"
            permit
        if http://security.ibm.com/attributes/resource/riskScore > 40 & AUTHENTICATION_LEVEL == "1"
            permit obligation {
                name=otp
            }
    }
}
```

For more information about the script elements, see "Authorization policy generation language" on page 46. Use the policy rule script to create a risk-based access policy.

Creating a risk-based access policy

After you write a script that specifies the policy rules, you must create and configure the authorization policy for risk-based access.

Procedure

1. Use the **manageRbaConfiguration** command to configure the runtime security service. Run the following commands to create properties and replace the sample values in *italics* with the values that you require.

   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.basic.authn.pwd
        -propertyValue passw0rd}
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.basic.authn.username
        -propertyValue wasadmin}
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.port
        -propertyValue 9080}
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName rtss.admin.ws.host
        -propertyValue localhost}
   ```

2. Use the policy generation language to create a script that specifies the policy rules. For more information, see "Authorization policy generation language" on page 46.
3. Use the **manageRbaPolicy** command to create the policy. Specify the path and file name of the script file as the value of the **policyFile** parameter.

   For example:

   ```
   $AdminTask manageRbaPolicy {-operation create -policyFile /tmp/rbapolicy}
   ```

   Note: You cannot use the **manageRbaPolicy** command to manage risk-based access policies, if the runtime security service is configured with IBM Tivoli Security Policy Manager as the policy administration point. For more information, see "Policy management with IBM Tivoli Security Policy Manager" on page 50.

Results

The output is an authorization policy file that is based on the rules that you specified in the script. Risk-based access is now configured to use the authorization policy file.

What to do next

Enable logs and traces for risk-based access.

Policy management with IBM Tivoli Security Policy Manager

If the runtime security service is configured with IBM Tivoli Security Policy Manager, you can use it as the policy administration point (PAP) to manage risk-based access policies.

You must first configure IBM Tivoli Security Policy Manager for risk-based access policy management.

You can then author a risk-based access policy with IBM Tivoli Security Policy Manager or import existing risk-based access policies.

Note: You cannot use the **manageRbaPolicy** command to manage risk-based access policies if the runtime security service is configured with IBM Tivoli Security Policy Manager as the policy administration point.

For detailed information about creating and managing policies with IBM Tivoli Security Policy Manager, see:

- The IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html
- The IBM Tivoli Security Policy Manager wiki at https://www.ibm.com/developerworks/wikis/display/tivolisecuritypolicymanager/Home

Enabling risk-based access in IBM Tivoli Security Policy Manager

Before you create and manage policies, you must configure the risk-based access policy information points (PIPs) and create the required services, obligations, and rule parameters.

Before you begin

Install and deploy risk-based access.

Note: When you deploy risk-based access, the **deploy** operation detects that the runtime security service is configured with IBM Tivoli Security Policy Manager. The existing instance of the runtime security service is not overwritten. The runtime security services of IBM Tivoli Federated Identity Manager and the runtime security services of IBM Tivoli Security Policy Manager must share the same WebSphere Application Server profile for risk-based access.

Procedure

1. Ensure that the runtime security services server is registered as a policy distribution target (PDT). For detailed steps, see the *Configuration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at

http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *registering the runtime security services server*.

2. Enable the risk-based access PIPs in IBM Tivoli Security Policy Manager.
   a. Stop the WebSphere Application Server instance where IBM Tivoli Security Policy Manager is installed.
   b. To configure the PIPs, modify the `security-services.xmi` file in the *WAS_profiles_dir*`/config/cells/`*cell_name*`/rtss` directory. Add the following information in the `<subComponents name="AttributeRetrievalServices">` section:

```
<subComponents name="AttributeRetrievalServices">
    <items name="RBA AttributeSession PIP">
        <properties>
            <values name="return.section" value="Resource" type="java.lang.String"/>
            <values name="type" value="custom" type="java.lang.String"/>
            <values name="id" value="com.tivoli.am.rba.pip.AttributeSessionPIP"
                    type="java.lang.String"/>
            <values name="enabled" value="true" type="java.lang.String"/>
        </properties>
    </items>
    <items name="RBA RiskCalculator PIP">
        <properties>
            <values name="return.section" value="Resource" type="java.lang.String"/>
            <values name="return.attributeId" value="risk.score" type="java.lang.String"/>
            <values name="type" value="custom" type="java.lang.String"/>
            <values name="id" value="com.tivoli.am.rba.pip.RiskCalculatorPIP"
                    type="java.lang.String"/>
            <values name="enabled" value="true" type="java.lang.String"/>
        </properties>
    </items>
    <items name="USER FINGERPRINT COUNT PIP">
        <properties>
            <values name="return.section" value="Resource" type="java.lang.String"/>
            <values name="type" value="custom" type="java.lang.String"/>
            <values name="id" value="com.tivoli.am.rba.pip.UserFingerprintsCountPIP"
                    type="java.lang.String"/>
            <values name="enabled" value="true" type="java.lang.String"/>
       </properties>
    </items>
</subComponents>
```

      For information about the precautions that you must take before editing the `security-services.xmi` file, see the *Configuration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *the security-services.xmi file*.
   c. Save a copy of this updated `security-services.xmi` file for your reference.
   d. Restart IBM Tivoli Security Policy Manager or restart WebSphere Application Server.
   e. Check the `security-services.xmi` file to ensure that the changes you made still exist. Also check the WebSphere Application Server server log files to verify that the PIP plug-in JAR file is loaded successfully.
3. Add the rule parameters that are required for risk-based access policy attributes.

   The following predefined rule parameters that are specific to risk-based access policies are provided with IBM Tivoli Security Policy Manager, Version 7.1 with fix pack 4 or later:

   IBM Security risk score
   The risk score that is computed by the risk-based access `RiskCalculatorPIP`. This policy information point (PIP) computes the risk associated with an incoming request.

   IBM Security registered device count
   The device count that is computed by the risk-based access

UserFingerprintCountPIP. This PIP computes the count of devices that are registered for a specific user ID in the risk-based access database.

a. To enable the predefined rule parameters, copy the required Java archive (JAR) file from the following location:

   TSPM_install_dir/rba/com.ibm.tspm.osgi.policy.rba.attributes_7.1.0.jar

   Copy the JAR file to the following WebSphere Application Server directory where IBM Tivoli Security Policy Manager is installed:

   WAS_profiles_dir/config/tspm/eclipse/plugins

b. IBM Security risk score and IBM Security registered device count are now available in the **Rule parameter** list in the IBM Tivoli Security Policy Manager console.

c. Add any custom rule parameters that you might require for the risk-based access policy rules. For example: TRANSACTION_AMOUNT. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *adding a rule parameter* and follow the steps to create custom rule parameters for risk-based access.

4. Create a service for risk-based access. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *creating a new service* and complete the steps to create a service for risk-based access. The following details are specific to risk-based access:

   a. On the Name and Description page, enter the WebSEAL server name in the **Application name** and **Application ID** fields. For example: localhost-default.

   b. On the Specify Actions page, select **No actions defined (define below)**.

   c. On the Service Structure page, in the **Element ID** field, enter the resource that must be protected by risk-based access, for example, the WebSEAL junction name, /FIM.

5. Add the required obligations.

   a. On the IBM Tivoli Security Policy Manager console, click **Identity and Access > Obligations**.

   b. On the Obligations page, click **Add**.

   c. Specify the name, identifier, and description of the obligation. For example:

      Name: Device Registration

      Identifier: http://security.ibm.com/obligations/registerDevice

      Description: Register device

What to do next

You can either migrate existing risk-based access policies or author risk-based access policies in the IBM Tivoli Security Policy Manager console.

Migrating policies to IBM Tivoli Security Policy Manager

If the runtime security service is configured with IBM Tivoli Security Policy Manager, you can use it as the policy administration point to manage risk-based access policies. You can export the policies from the risk-based access runtime security services, import them into IBM Tivoli Security Policy Manager.

Before you begin

- Install and deploy risk-based access.
- Enable risk-based access in IBM Tivoli Security Policy Manager.

Procedure

1. To export all of the policies in the risk-based access runtime security service, run the following command:

   ```
   $AdminTask manageRbaPolicy {-operation export -fileId file_path}
   ```

   where the variable *file_path* represents the path and file name of the Java archive (JAR) file, for example `/tmp/policy.jar`.

   The policies are exported in XML format to a JAR file.
2. Extract the individual policies from the JAR file.
3. Import the individual policies into IBM Tivoli Security Policy Manager. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *importing an authorization policy* and follow the steps to import the XML policy files.
4. Attach the imported policies to a service. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *attaching a service to an existing policy*. The following steps describe how you can attach multiple policies to a service:
   a. In the IBM Tivoli Security Policy Manager console, click **Identity and Access > Services**.
   b. On the Services page, click the service that you created for risk-based access. For example: `localhost-default`.
   c. Under **Policies**, add the risk-based access policies that you imported.
5. Deploy the risk-based access polices. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *deploying authorization policies* and follow the tasks to configure and distribute authorization policies.
6. Use IBM Tivoli Security Policy Manager to manage the risk-based access policies. For more information, see the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *managing authorization policies*.

What to do next

You can also create risk-based access policies with IBM Tivoli Security Policy Manager.

Creating risk-based access policies with IBM Tivoli Security Policy Manager

If the runtime security service is configured with IBM Tivoli Security Policy Manager, you can use it to author and manage risk-based access policies.

Before you begin

- Install and deploy risk-based access.
- Enable risk-based access in IBM Tivoli Security Policy Manager.

About this task

Note: You cannot use the **manageRbaPolicy** command to create risk-based access policies if the runtime security service is configured with IBM Tivoli Security Policy Manager as the policy administration point.

Procedure

1. Create an authorization policy. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *creating a rule authorization policy* and complete the steps to create a policy for risk-based access. The following details are specific to risk-based access:
   a. In the **Name** field, enter the name of the risk-based access policy. You must prefix the policy name with the string `RPS:`. For example: `RPS:RBA_OneTimePassword`.
   b. Under Policy Definition, in the Services section, specify the service that you created for risk-based access. For example: `localhost-default`.
   c. Define the **Rules** by using predefined rule parameters or custom rule parameters that you created for risk-based access. For example:

```
If    IBM Security risk score > 40
and   IBM Security registered device count <= 70
and   AUTHENTICATION_LEVEL=1
Then  Permit access
```

      Note: You can use the following predefined rule parameters that are specific to risk-based access policies. These parameters are provided with IBM Tivoli Security Policy Manager, Version 7.1 with fix pack 4 or later:

      IBM Security risk score
      The risk score that is computed by the risk-based access `RiskCalculatorPIP`. This policy information point (PIP) computes the risk associated with an incoming request.

      IBM Security registered device count
      The device count that is computed by the risk-based access `UserFingerprintCountPIP`. This PIP computes the count of devices that are registered for a specific user ID in the risk-based access database.

      Restriction: The `!=` (not equal to) operator is not supported in the IBM Tivoli Security Policy Manager console.
   d. Specify the **Obligations**. For example: `Return "Device Registration" on permit`.
2. Deploy the risk-based access polices. See the *Administration Guide* in the IBM Tivoli Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *deploying authorization policies* and follow the tasks to configure and distribute authorization policies.
3. Use IBM Tivoli Security Policy Manager to manage the risk-based access policies. For more information, see the *Administration Guide* in the IBM Tivoli

Security Policy Manager Version 7.1 Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tspm.doc_7.1/welcome.html. Search for *managing authorization policies*.

Device registration

Device registration is the process that stores the device fingerprint of the user in the risk-based access database. The device fingerprint contains information required for risk score calculation.

The rules that you specify in the risk-based access policy determine whether a device is registered silently or only after the user consents to the registration. Use one of the end-to-end sample policies to specify the basic rules for either silent or consent-based registration.

You can also configure the maximum number of devices that can be registered for a user.

Silent device registration

Silent device registration is the process of registering the device after the user responds successfully to a secondary challenge. Silent registration does not require any further interaction or consent from the user. The risk-based access policy that you configure determines whether a device must be registered silently.

You can use the "Sample script for risk-based access policy rules" on page 48 to configure a policy that silently registers the device of the user. The ready-to-use script defines the basic rules that are required for an end-to-end functioning policy.

Consent-based device registration

Consent-based device registration is the process of registering the device fingerprint only after the user consents to the device registration. Use the sample rules to configure a policy that allows the user to specify whether to add the device to the list of registered devices.

A typical scenario that would require consent-based device registration is when a user attempts to access a protected resource from a public access environment. For example, a user might log in from an internet café or airport kiosk. After the user logs in and successfully responds to the secondary challenge, a consent form is presented. The consent form can be an HTML page where users can specify that they consent to the device registration.

- If the user consents to the registration of the device, the device is registered and access is permitted. The next time that the user logs in from the same device, the consent form is not presented because the device is already registered.
- If the user does not consent to the registration of the device, the device is not registered and the access is permitted. If the user logs in from the same device again, the secondary challenge and the consent form are presented again. The process is repeated because the risk score is high when a user logs in from a device that is not registered.

Configuring consent-based device registration

Configure the required attributes and properties so that you can prompt a user for consent to register a device.

Before you begin

Update the WebSEAL configuration file:

1. Append the [obligations-levels-mapping] stanza. Specify the mapping between obligation levels that the PDP returns and the authentication levels in WebSEAL. For example, append:

   ```
   [obligations-levels-mapping]
   otp=3
   consent=4
   ```

2. Enter a new trigger URL in the [eai-trigger-urls] stanza for the consent form. Use the same URL as the one used for step 12 on page 57. For example, enter:

   ```
   trigger = /FIM/sps/ac/html/consent-form.html*
   ```

3. Update the [authentication-levels] stanza to specify that the consent authentication level must use EAI. The examples in this procedure show an authentication level of 4. For example, to specify an EAI authentication level of 4, enter the following text:

   ```
   level = unauthenticated
   level = password
   level = ext-auth-interface
   level = ext-auth-interface
   level = ext-auth-interface
   ```

See the *WebSEAL Administration Guide* in the IBM Tivoli Access Manager for e-business information center for more information.

Procedure

1. Enable consent-based registration by adding http://security.ibm.com/attributes/environment/userConsent as a session attribute:

   ```
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId http://security.ibm.com/attributes/environment/userConsent
        -weight 0 -session true -device false}
   ```

2. Configure the property for the external authentication interface (EAI) header for the authentication level. The property value must match the value that is set for eai-auth-level-header in the [eai] stanza of the WebSEAL configuration file.

   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName eai.header.auth.level -propertyValue am-eai-auth-level}
   ```

3. Configure the property for the EAI header for the user ID. The property value must match the value that is set for eai-user-id-header in the [eai] stanza of the WebSEAL configuration file.

   ```
   $AdminTask manageRbaConfiguration {-operation create
        -propertyName eai.header.user.id -propertyValue am-fim-eai-user-id}
   ```

4. Optional: Create a device.name attribute so that the user can choose any device name to represent their device during consent-based device registration.

   ```
   $AdminTask manageRbaRiskProfile {-operation create
        -attributeId device.name -weight 0 -session true -device true}
   ```

5. Load the changes:

   ```
   $AdminTask manageRbaConfiguration {-operation reload}
   ```

6. Create a policy that contains consent-based registration rules. The following sample policy shows rules for consent-based registration. This example calls an extra external authentication interface, such as a consent form, if the device is not registered in the risk-based access database.

```
Policy {
    name = localhost-default
    Rule {
            name = RBADemoApplicationPolicyRule1
            Target {
                    subject = any
                    action = any
                    resource = any
            }
            resource integer http://security.ibm.com/attributes/resource/riskScore
            subject string AUTHENTICATION_LEVEL
            environment string http://security.ibm.com/attributes/environment/userConsent

            if AUTHENTICATION_LEVEL == "1" & http://security.ibm.com/attributes/resource/riskScore > 40
                    permit obligation {
                    name=otp
            }
            if AUTHENTICATION_LEVEL == "3"
                    permit obligation {
                    name=consent
            }
            if AUTHENTICATION_LEVEL == "4" & http://security.ibm.com/attributes/environment/userConsent ==
"true" & http://security.ibm.com/attributes/resource/riskScore > 40
                    permit obligation {
                    name=http://security.ibm.com/obligations/registerDevice
            }
            if AUTHENTICATION_LEVEL == "4" & http://security.ibm.com/attributes/environment/userConsent == "false"
                    permit

            if http://security.ibm.com/attributes/resource/riskScore < 40
                    permit
        }
```

7. Save the policy.
8. Make the policy changes active by running one of the following commands:
   - For a new policy, run the following command:

     `$AdminTask manageRbaPolicy {-operation create -policyFile policy_file}`
   - For an updated policy, run the following command:

     `$AdminTask manageRbaPolicy {-operation update -policyFile policy_file}`

   where *policy_file* is the path and policy file name that was updated or created.
9. Create a protected object policy (POP) to define the authentication level for a protected resource.

   `pdadmin> pop create pop_name`
10. Modify the authentication level in the POP to restrict access to the resource to only users with an authentication level of 3:

    `pdadmin> pop modify pop_name set ipauth anyothernw 3`
11. Attach the POP to the protected resource:

    `pdadmin> pop attach object_name pop_name`

    where *object_name* is the name of the protected resource, such as `/WebSEAL/webseal_instance/webseal_junction_name/sps/ac/html/consent-form.html`.
12. Use the following URL to redirect the user to the ready-to-use consent HTML form that is provided with risk-based access:

    `http://myhost.company.com/FIM/sps/ac/html/consent-form.html?eai-auth-level=stepup_level`

    where:
    - *myhost.company.com* is the host name.
    - *eai-auth-level* matches the property value of `eai-auth-level-header` in the `[eai]` stanza of the WebSEAL configuration file.
    - *stepup_level* is the value that maps to the consent obligation, which is specified in the WebSEAL configuration file under the `[obligations-levels-mapping]` stanza.

Results

The user is now prompted to provide consent to register a device.

If you used the option in step 4 on page 56, the user can register their device by using a device name of their choice.

Configuring the number of devices registered for a user

To minimize the risk of fraudulent access, a limit must be set on the maximum number of devices that are registered for a user. By default, a maximum of 10 devices are registered for each user. You can change this number by modifying the value of the `max.no.of.registered.devices` configuration property.

Procedure

Use the **`manageRbaConfiguration`** command to configure the `max.no.of.registered.devices` property. Specify a value that indicates the maximum number of devices that a user can register. The default value is `10`. Valid values are `1` to `20`.
For example:

```
$AdminTask manageRbaConfiguration {-operation update
     -propertyName max.no.of.registered.devices -propertyValue 12}
```

Results

When a user registers a device, risk-based access counts the number of devices that are already registered for the user and takes the following actions:

- If the count is equal to or greater than the value of the `max.no.of.registered.devices` property, the registration information for the oldest device is deleted. The new device is then registered.
- If the count is lesser than the value of the `max.no.of.registered.devices` property, the new device is registered.

Chapter 6. Auditing

Risk-based access generates audit records of critical events. The audit records are written to the audit log file. Use the audit logs to monitor the activities and track potential security problems that are related to risk-based access.

Audit records are generated for commands that manage the risk-based access configuration, policies, attributes, and sessions. Audit records are also generated for critical operations related to risk-based access policy information points (PIPs) and device registration.

Managing audit log settings

You can use specific configuration properties to enable or disable auditing and to change the default audit log settings. For example, you can change the settings for the name, location, and maximum size of the audit log files.

About this task

By default, auditing is enabled for risk-based access. Default settings are specified for the audit log files. You can use the following configuration properties if you want to change the default audit settings:

com.tivoli.am.rba.audit.enableAuditing

Enables or disables auditing.

The default value is `true`.

com.tivoli.am.rba.audit.file.name

Specifies how the audit log files are named.

The default prefix of the log file name is `rba_audit_log`.

com.tivoli.am.rba.audit.file.location

Specifies the location of the audit log file.

The default location is the `rba_audit` directory in the WebSphere Application Server default `logs` directory.

AIX® or Linux systems

`/opt/IBM/WebSphere/AppServer/profiles/`*`profile_name`*`/logs/rba_audit`

Windows systems

`C:\Program Files\IBM\WebSphere\AppServer\profiles\`*`profile_name`*`\logs\rba_audit`

The location that you specify for the `com.tivoli.am.rba.audit.file.location` property is relative to the default `logs` directory for WebSphere Application Server.

For example, on a Windows system, if you specify the value as `rba`, then the log files are stored in the `C:\Program Files\IBM\WebSphere\AppServer\profiles\`*`profile_name`*`\logs\rba` directory.

com.tivoli.am.rba.audit.file.size

Specifies the maximum audit file size in megabytes (MB).

The default value is `100000` MB.

com.tivoli.am.rba.audit.number.of.files

Specifies the maximum number of audit files that are created before the oldest file is overwritten.

The default value is `10`.

Procedure

1. Use the **manageRbaConfiguration** command to configure the risk-based access properties for auditing.

   ```
   $AdminTask manageRbaConfiguration {-operation update
        [-propertyName property_name]
        [-propertyValue property_value]}
   ```

   Where:

   property_name is one of the properties described in About this task.

   property_value is the associated value for the property described in About this task.

2. To configure audit records for runtime security services external authorization service (EAS) requests, specify the **audit-log-cfg** entry under the **[rtss-eas]** stanza of the default WebSEAL configuration file named `webseald-default.conf`. For example, you can add the following entry:

   ```
   audit-log-cfg = file path=/tmp/rtss-audit.log,flush=20,rollover=2000000,
        buffer_size=8192,queue_size=48
   ```

   For more information, see "Sample configuration data for runtime security services external authorization service" on page 29.

Audit logs

The audit log files are generated in eXtensible Markup Language (XML). The log files contain elements that record the details of each audit event, such as the user, time, action, and outcome.

For information about how to locate the audit log files, see the description of the **com.tivoli.am.rba.audit.file.location** property in "Managing audit log settings" on page 59.

The following table lists the audit elements for the audit events of risk-based access.

Table 6. Audit event elements

Audit event element	Description
`userInfoList`	The user who triggered the event
`creationTime`	The date and time when the event was generated
`actionInfo`	The event name, which indicates the operation For example: `POLICY_CREATE_EVENT`
`outcome`	The outcome of the operation specified as `SUCCESSFUL` or `FAILURE`

Table 6. Audit event elements (continued)

Audit event element	Description
globalInstanceId	The identifier for the audit event
extensionName	The name of the event type For example, for risk-based access auditing, the event type is SECURITY_RBA_AUDIT_AUTHZ.
messages	Message details of the event action

The following risk-based access operations are recorded in the example snippet of the audit log file:

- Attribute search was successful.
- Creation of an attribute failed.
- Risk score was calculated successfully.
- Device was registered successfully.
- Device registration information was deleted successfully.

```
<CommonBaseEvent creationTime="2012-08-01 04:45:56 CDT" extensionName="SECURITY_RBA_AUDIT_AUTHZ"
          globalInstanceId="uuiddc425c98-0138-1039-b7ba-882c9b14694c"
          msg="The command manageRbaConfiguration's operation was succuessful with the given arguments
                    {operation=search}"
          version="1.0.1">
    <extendedDataElements name="AUDIT_SCHEMA_VERSION" type="string">
        <values>1.1</values>
    </extendedDataElements>
    <extendedDataElements name="actionInfo" type="noValue">
        <children name="urn:oasis:names:tc:xacml:1.0:action:action-id" type="string">
            <values>RUNTIME_CONFIGURATION_SEARCH_EVENT</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="outcome" type="noValue">
        <children name="result" type="string">
            <values>SUCCESSFUL</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="userInfoList" type="noValue">
        <children name="appUserName" type="string">
            <values>defaultWIMFileBasedRealm/wasadmin</values>
        </children>
    </extendedDataElements>
</CommonBaseEvent>

<CommonBaseEvent creationTime="2012-08-01 04:45:56 CDT" extensionName="SECURITY_RBA_AUDIT_AUTHZ"
          globalInstanceId="uuiddc51c52c-0138-1620-b4ab-882c9b14694c"
          msg="The command manageRbaConfiguration's operation was unsuccuessful with the given arguments
                    {operation=create}" version="1.0.1">
    <extendedDataElements name="AUDIT_SCHEMA_VERSION" type="string">
        <values>1.1</values>
    </extendedDataElements>
    <extendedDataElements name="actionInfo" type="noValue">
        <children name="urn:oasis:names:tc:xacml:1.0:action:action-id" type="string">
            <values>RUNTIME_CONFIGURATION_CREATE_EVENT</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="outcome" type="noValue">
        <children name="result" type="string">
            <values>FAILURE</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="userInfoList" type="noValue">
        <children name="appUserName" type="string">
            <values>defaultWIMFileBasedRealm/wasadmin</values>
        </children>
    </extendedDataElements>
</CommonBaseEvent>

<CommonBaseEvent creationTime="2012-08-01 04:45:56 CDT" extensionName="SECURITY_RBA_AUDIT_AUTHZ"
          globalInstanceId="uuid93126693-0137-1b69-9a86-ae316fbc293b"
          msg="FBTRBA200I Calculated risk percentage: "100"." version="1.0.1">
    <extendedDataElements name="AUDIT_SCHEMA_VERSION" type="string">
        <values>1.1</values>
    </extendedDataElements>
    <extendedDataElements name="actionInfo" type="noValue">
        <children name="urn:oasis:names:tc:xacml:1.0:action:action-id" type="string">
            <values>CALCULATE_RISK_SCORE_EVENT</values>
        </children>
    </extendedDataElements>
```

```
    <extendedDataElements name="outcome" type="noValue">
        <children name="result" type="string">
            <values>SUCCESSFUL</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="userInfoList" type="noValue">
        <children name="appUserName" type="string">
            <values>cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com</values>
        </children>
    </extendedDataElements>
</CommonBaseEvent>

<CommonBaseEvent creationTime="2012-08-01 04:45:56 CDT" extensionName="SECURITY_RBA_AUDIT_AUTHZ"
          globalInstanceId="uuid93126720-0137-1684-923c-ae316fbc293b"
          msg="FBTRBA201I A new device registered for user:
                    "cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com"." version="1.0.1">
    <extendedDataElements name="AUDIT_SCHEMA_VERSION" type="string">
        <values>1.1</values>
    </extendedDataElements>
    <extendedDataElements name="actionInfo" type="noValue">
        <children name="urn:oasis:names:tc:xacml:1.0:action:action-id" type="string">
            <values>DEVICE_REGISTRATION_EVENT</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="outcome" type="noValue">
        <children name="result" type="string">
            <values>SUCCESSFUL</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="userInfoList" type="noValue">
        <children name="appUserName" type="string">
            <values>cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com</values>
        </children>
    </extendedDataElements>
</CommonBaseEvent>

<CommonBaseEvent creationTime="2012-08-01 04:45:56 CDT" extensionName="SECURITY_RBA_AUDIT_AUTHZ"
          globalInstanceId="uuid9312b115-0137-1a81-a981-ae316fbc293b"
          msg="FBTRBA203I A device deleted for user:
                    "cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com"." version="1.0.1">
    <extendedDataElements name="AUDIT_SCHEMA_VERSION" type="string">
        <values>1.1</values>
    </extendedDataElements>
    <extendedDataElements name="actionInfo" type="noValue">
        <children name="urn:oasis:names:tc:xacml:1.0:action:action-id" type="string">
            <values>DEVICE_DELETION_EVENT</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="outcome" type="noValue">
        <children name="result" type="string">
            <values>SUCCESSFUL</values>
        </children>
    </extendedDataElements>
    <extendedDataElements name="userInfoList" type="noValue">
        <children name="appUserName" type="string">
            <values>cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com</values>
        </children>
    </extendedDataElements>
</CommonBaseEvent>
```

The following log is an example of an audit record for a runtime security services external authorization service (EAS) request. One audit record is generated for each EAS request.

```
<rtss-event>
   <operation>r</operation>
   <pdp-decision>Permit</pdp-decision>
   <pdp-obligation>otp</pdp-obligation>
   <protected-resource>/WebSEAL/localhost-Default2/WAS</protected-resource>
   <server-name>localhost-Default2</server-name>
   <stepup-level>3</stepup-level>
   <tam-policy-decision>AZN_C_PERMITTED</tam-policy-decision>
   <timestamp>2012-05-06 13:55:22 GMT</timestamp>
   <user>sec_master</user>
   <user-agent>Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.1 (KHTML, like Gecko)
          Chrome/21.0.1145.0 Safari/537.1</user-agent>
</rtss-event>
```

Chapter 7. Reference

Reference information is organized to help you locate particular facts quickly. It includes information about the commands to manage risk-based access, attributes for configuring the risk profile, and properties for configuring risk-based access.

Command reference

The risk-based access command-line interface uses the **wsadmin** command framework. Use the command syntax and examples to configure and manage attributes, policies, runtime properties, fingerprints, and devices for risk-based access.

manageRbaConfiguration command

Use the **manageRbaConfiguration** command to create and configure risk-based access properties. The properties specify the configuration for the database, attribute collection service, runtime security service, attribute matchers, and other risk-based access components.

Purpose

Use the **manageRbaConfiguration** command to create, update, delete, or search for configuration properties. You can also use this command to deploy or undeploy risk-based access and load or reset configuration settings.

For a list of all risk-based access properties, see "Configuration properties" on page 78.

Note: If you configure a property with a name that ends with `assword` or `pwd`, the value of the property is obfuscated when it is stored in the database.

Syntax

```
$AdminTask manageRbaConfiguration {-operation operator [-propertyName property_name]
     [-propertyValue property_value] [-fileId response_file_path]}
```

Parameters

Use the following parameters with the **manageRbaConfiguration** command:

-operation *operator*
Specifies the operation for risk-based access configuration. This parameter is required.

The valid values are `deploy`, `undeploy`, `configure`, `create`, `delete`, `update`, `unconfigure`, `search`, `reload`, and `createResponseFile`.

Table 7. Required and optional parameters for operators of the **manageRbaConfiguration** *command*

Operator	Description	Required parameters	Optional parameters
`deploy`	Deploys risk-based access by installing the runtime EAR file and configuration files on the application server. For more information, see "Deploying risk-based access" on page 14. **Note:** If the `deploy` operation detects that the runtime security service is configured with IBM Tivoli Security Policy Manager, the existing instance of the runtime security service is not overwritten.	None	None
`undeploy`	Undeploys risk-based access by removing the runtime EAR file and the risk-based access configuration files from the configuration repository of the application server. **Note:** If the `undeploy` operation detects that the runtime security service is configured with IBM Tivoli Security Policy Manager, then the instance of runtime security service is not removed or undeployed.	None	None
`create`	Creates a risk-based access configuration property and sets its value.	**-propertyName**, **-propertyValue**	None
`delete`	Deletes a risk-based access configuration property.	**-propertyName**	None

Table 7. Required and optional parameters for operators of the **manageRbaConfiguration** *command (continued)*

Operator	Description	Required parameters	Optional parameters
`update`	Updates an existing risk-based access configuration property.	**-propertyName**, **-propertyValue**	None
`unconfigure`	Unconfigures or resets the risk-based access configuration settings.	None	None
`search`	Searches for a specified property or lists all properties and values.	None	**-propertyName**
`reload`	Publishes pages and plug-ins and reloads the Tivoli Federated Identity Manager runtime environment.	None	None
`createResponseFile`	Creates a response file template.	**-fileId**	None
`configure`	Configures risk-based access by using a response file to run batch operations for configuration properties.	**-fileId**	None

-propertyName *property_name*
The name of the risk-based access configuration property, for example, `dbdatasource`.

-propertyValue *property_value*
The value of the risk-based access configuration property.

-fileId *response_file_path*
The path of the input or output response file:

- To output an XML response file template for the **manageRbaConfiguration** command, use the `createResponseFile` operator with the **fileId** parameter.
- To input an XML response file to configure risk-based access, use the `configure` operator with the **fileId** parameter. The specified response file must contain the information that you would typically specify on the command line. You can edit the XML response file with a text editor to specify the values that you require. For more information, see "Response file example for manageRbaConfiguration command" on page 67.

Note: You can use the **fileId** parameter only with the **operator** parameter. Any other parameters that you specify are ignored.

Examples

The following examples show the correct syntax for various operations of the **manageRbaConfiguration** command:

Deploy risk-based access:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation deploy}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation deploy]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'deploy'])
```

For more information, see "Deploying risk-based access" on page 14.

Undeploy risk-based access:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation undeploy}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation undeploy]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'undeploy'])
```

Configure risk-based access:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation configure -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation configure -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'configure',
    '-fileId', '/tmp/resp.xml'])
```

Unconfigure or reset the risk-based access configuration settings:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation unconfigure}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation unconfigure]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'unconfigure'])
```

Create a risk-based access configuration property and specify its value:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation create -propertyName dbdatasource
    -propertyValue jdbc/rbadb}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation create -propertyName dbdatasource
    -propertyValue jdbc/rbadb]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'create',
    '-propertyName', 'dbdatasource', '-propertyValue', 'jdbc/rbadb'])
```

Update the value of an existing risk-based access configuration property:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation update -propertyName dbdatasource
    -propertyValue jdbc/rba2db}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation update -propertyName dbdatasource
    -propertyValue jdbc/rba2db]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'update',
    '-propertyName', 'dbdatasource', '-propertyValue', 'jdbc/rba2db'])
```

Delete a risk-based access configuration property and its value:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation delete -propertyName dbdatasource}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation delete -propertyName dbdatasource]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'delete',
     '-propertyName', 'dbdatasource'])
```

Search for a specified property to view the property and its value:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation search -propertyName dbdatasource}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation search -propertyName dbdatasource]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'search',
     '-propertyName', 'dbdatasource'])
```

View all risk-based access properties and their values:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation search}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation search]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'search'])
```

Publish pages and plug-ins and reload the Tivoli Federated Identity Manager runtime environment:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation reload}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation reload]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'reload'])
```

Create a response file template for this command:

- Using Jacl:

```
$AdminTask manageRbaConfiguration {-operation createResponseFile
     -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaConfiguration('[-operation createResponseFile
     -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaConfiguration(['-operation', 'createResponseFile',
     '-fileId', '/tmp/resp.xml'])
```

Note: After you use the **manageRbaConfiguration** command to change configuration properties, you must run the **manageRbaConfiguration** command with the **reload** option for the changes to take effect.

Response file example for manageRbaConfiguration command

Use a response file to run batch operations for the **manageRbaConfiguration** command. The response file contains information that you would normally specify on the command line. Using a response file helps you automate your configuration process.

To create a response file template, use the **fileId** parameter with the **createResponseFile** operator of the **manageRbaConfiguration** command. Edit the response file template to specify the values that you require. For more information about the **fileId** parameter, see "manageRbaConfiguration command" on page 63.

The following response file example shows how to create a configuration property for the data source and specify the value of the configuration property:

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0" class="java.beans.XMLDecoder">
 <object class="java.util.Properties">
  <void method="put">
    <string>dbdatasource</string>
    <string>jdbc/rbadb</string>
  </void>
 </object>
</java>
```

manageRbaDevices command

Use the **manageRbaDevices** command to manage the registered devices and the device fingerprints. The device fingerprint consists of the user ID and key-value pairs of attributes, which are stored in the risk-based access database.

Purpose

Use the **manageRbaDevices** command to create, search, or delete device registration information.

Syntax

```
$AdminTask manageRbaDevices {-operation operator [-userId unique_user_ID]
     [-deviceId unique_device_ID] [-attributes attribute_list]}
```

Parameters

Use the following parameters with the **manageRbaDevices** command:

-operation *operator*
Specifies the operation for the attribute. This parameter is required.

The valid values are `create`, `delete`, and `search`.

Table 8. Required and optional parameters for operators of the **manageRbaDevices** *command*

Operator	Description	Required parameters	Optional parameters
`create`	Creates registration information for a device.	**-userId**, **-attributes**	**-delimiter**
`delete`	Deletes the registration information of a specified device or all devices of a specified user	Either **-userId** or **-deviceId** is required	**-userId**, **-deviceId**
`search`	Lists all the registered devices or searches for the devices that are registered for a specified user.	None	**-userId**

-userId *unique_user_ID*
An alphanumeric value that is the unique ID of the user.

If you do not specify the user ID for a `search` operation, all registered devices are listed.

-deviceId *unique_device_ID*
An alphanumeric value that is the unique ID of a device.

If you do not specify the device ID for a `delete` operation, all of the registration information for the specified user ID is deleted.

-attributes *attribute_list*
A list of key-value pairs of attributes.

The default delimiter is the percent symbol (%).

-delimiter *delimiter*
The character that delimits the key-value pairs in the list of attributes.

Use the **-delimiter** parameter to specify the delimiter when you register a device with the **-attributes** parameter.

If you do not specify the *delimiter* parameter, the percent symbol (%) is the default delimiter.

Examples

The following examples show the correct syntax for various operations of the **manageRbaDevices** command:

Create registration information for a device. The following example uses the default delimiter, which is the percent symbol (%):

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation create -userId user1
     -attributes attribute1=value1%attribute2=value2%attribute3=value3}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation create -userId user1
     -attributes attribute1=value1%attribute2=value2%attribute3=value3]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'create', '-userId', 'user1',
     '-attributes' 'attribute1=value1%attribute2=value2%attribute3=value3'])
```

Create registration for a device and specify a delimiter for the list of attributes. The following example uses a comma (,) as the delimiter:

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation create -userId user1 -delimiter ,
     -attributes attribute1=value1,attribute2=value2,attribute3=value3}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation create -userId user1 -delimiter ,
     -attributes attribute1=value1,attribute2=value2,attribute3=value3]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'create', '-userId', 'user1',
     '-delimiter' ','
     '-attributes' 'attribute1=value1,attribute2=value2,attribute3=value3'])
```

List all of the registered devices:

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation search}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation search]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'search'])
```

Search for the devices that are registered for a specific user:

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation search -userId user1}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation search -userId user1]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'search', '-userId', 'user1'])
```

Delete the registration information of all devices for a specified user:

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation delete -userId user1}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation delete -userId user1]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'delete', '-userId', 'user1'])
```

Delete the registration information of a specified device:

- Using Jacl:

```
$AdminTask manageRbaDevices {-operation delete -deviceId device1}
```

- Using Jython string:

```
$AdminTask manageRbaDevices('[-operation delete -deviceId device1]')
```

- Using Jython list:

```
$AdminTask manageRbaDevices(['-operation', 'delete', '-deviceId', 'device1'])
```

manageRbaPolicy command

After you write a script with policy rules, you must use the **manageRbaPolicy** command to create and configure an authorization policy for risk-based access.

Purpose

Use the **manageRbaPolicy** command to create, update, delete, and search for policies.

Before you create a policy, you must write a script that specifies the policy rules in the "Authorization policy generation language" on page 46.

Note: You cannot use the **manageRbaPolicy** command to manage risk-based access policies if the runtime security service is configured with IBM Tivoli Security Policy Manager as the policy administration point. For more information, see "Policy management with IBM Tivoli Security Policy Manager" on page 50.

Syntax

```
$AdminTask manageRbaPolicy {-operation operator [-policyId unique_ID]
     [-policyFile file_name] [-serviceName service_name] [-fileId response_file_path]}
```

Parameters

Use the following parameters with the **manageRbaPolicy** command:

-operation *operator*
Specifies the operation for the policy. This parameter is required.

The valid values are `create`, `update`, `delete`, `search`, `export`, and `createResponseFile`.

Table 9. Required and optional parameters for operators of the **manageRbaPolicy** *command*

Operator	Description	Required parameters	Optional parameters
`create`	Creates a policy.	**-policyFile**	**-policyId**, **-fileId**
`update`	Updates a policy.	**-policyFile**	**-policyId**, **-fileId**
`delete`	Deletes a policy.	**-serviceName**	**-policyId**, **-fileId**
`search`	Searches for policies.	None	**-policyId**, **-serviceName**
`export`	Exports a policy to a Java archive (JAR) file.	**-fileId**	None
`createResponseFile`	Creates a response file template or uses a response file to run batch operations.	**-fileId**	None

-policyId *unique_ID*
An alphanumeric value that is the unique ID of the policy.

Note: In an IBM Tivoli Access Manager for e-business environment that uses WebSEAL, the value of the **-policyId** parameter is typically the same as the **-serviceName** parameter. In such cases, the runtime security services stores the policy with _DEFAULT_ as the policy ID. Hence, when you search for a specified policy, you must search with the **-policyId** value as _DEFAULT_ along with the **-serviceName** parameter.

-policyFile *file_path*
The path and file name of the policy.

-serviceName
The service name that the policy uses.

In an IBM Tivoli Access Manager for e-business environment that uses WebSEAL, the service name is the same as the WebSEAL server name, for example, `localhost-default`.

You can use the service name to search for a specific policy.

-fileId *response_file_path*
The path of the input or output response file or exported policy file:

- To output an XML response file template for the **manageRbaPolicy** command, use the `createResponseFile` operator with the **fileId** parameter.
- To input an XML response file to configure policies, use the `create`, `update`, or `delete` operators with the **fileId** parameter. The specified response file must contain the information that you would typically specify on the command line. Edit the XML response file with a text editor to specify the values that you require. For more information, see "Response file example for manageRbaPolicy command" on page 73.
- To export all the policies in the runtime security service that is configured for risk-based access, use the `export` operator with the **fileId** parameter. The policies are exported in XML format to a JAR file.

Note: You can use the **fileId** parameter only with the **operator** parameter. Any other parameters that you specify are ignored.

Examples

The following examples show the correct syntax for various operations of the **manageRbaPolicy** command:

Create a policy:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation create -policyId RBA_policy
    -policyFile /tmp/policy001}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation create -policyId RBA_policy
    -policyFile /tmp/policy001]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'create', '-policyId', 'RBA_policy',
    '-policyFile' '/tmp/policy001'])
```

Modify a policy:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation update -policyFile /tmp/policy001}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation update -policyFile /tmp/policy001]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'update', '-policyFile', '/tmp/policy001'])
```

Search for a policy by its ID:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation search -policyId RBA_policy}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation search -policyId RBA_policy]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'search', '-policyId', 'RBA_policy'])
```

Search for all policies:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation search}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation search]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'search'])
```

Delete a policy

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation delete -serviceName localhost-default}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation delete -serviceName localhost-default]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'delete', '-serviceName', 'localhost-default'])
```

Export policies to a JAR file:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation export -fileId /tmp/policy.jar}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation export -fileId /tmp/policy.jar]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'export', '-fileId', '/tmp/policy.jar'])
```

Create a response file template for this command:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation createResponseFile -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation createResponseFile -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'createResponseFile',
     '-fileId', '/tmp/resp.xml'])
```

Create a policy by using a response file as input:

- Using Jacl:

```
$AdminTask manageRbaPolicy {-operation create -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaPolicy('[-operation create -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaPolicy(['-operation', 'create', '-fileId', '/tmp/resp.xml'])
```

Response file example for manageRbaPolicy command

Use a response file to run batch operations for the **manageRbaPolicy** command. The response file contains information that you would normally specify on the command line. Using a response file helps you automate your configuration process.

To create a response file template, use the **fileId** parameter with the **createResponseFile** operator of the **manageRbaPolicy** command. Edit the response file template to specify the values that you require. For more information about the **fileId** parameter, see "manageRbaPolicy command" on page 70.

The following response file example shows how to create a policy named rbapolicy and specify policy properties such as **policyId** and **policyFile**:

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0" class="java.beans.XMLDecoder">
 <object class="java.util.ArrayList">
  <void method="add">
   <object class="com.tivoli.am.rba.cli.CLIPolicyParameters">
    <void property="policyId">
     <string>rbapolicy</string>
    </void>
    <void property="policyFile">
     <string>/tmp/policy001.xml</string>
    </void>
   </object>
  </void>
 </object>
</java>
```

manageRbaRiskProfile command

Use the **manageRbaRiskProfile** command to manage and configure weights for the risk attributes. Risk score calculation is based on the weights that you specify for the attributes.

Purpose

Use the **manageRbaRiskProfile** command to create, update, delete, and search for attributes.

For a list of all risk-based access attributes, see "Attributes for risk profiles and policies" on page 85.

Syntax

```
$AdminTask manageRbaRiskProfile {-operation operator [-attributeId unique_ID]
     [-weight weight] [-device true|false] [-session true|false]
     [-behavior true|false] [-fileId response_file_path]}
```

Parameters

Use the following parameters with the **manageRbaRiskProfile** command:

-operation *operator*
Specifies the operation for the attribute. This parameter is required.

The valid values are `create`, `update`, `delete`, `search`, and `createResponseFile`.

Note: An error occurs if you attempt to delete an attribute that is stored in the database as part of a registered device fingerprint. A device fingerprint consists of a user ID and a set of key-value pairs of attributes that identify a particular user. You can delete only the attributes that are not registered as part of any device fingerprint in the database.

Table 10. Required and optional parameters for operators of the **manageRbaRiskProfile** *command*

Operator	Description	Required parameters	Optional parameters
`create`	Creates an attribute and specify its weight.	**-attributeId**, **-weight**	**-fileId**, **-device**, **-session**, **-behavior**
`update`	Updates an existing attribute.	**-attributeId**, **-weight**	**-fileId**, **-device**, **-session**, **-behavior**
`delete`	Deletes an attribute.	**-attributeId**	**-fileId**
`search`	Searches for a specific attribute or lists all attributes.	None	**-attributeId**, **-fileId**
`createResponseFile`	Creates a response file template or uses a response file to run batch operations.	**-fileId**	None

-attributeId *unique_ID*
An alphanumeric value that is the unique ID of the attribute.

If you do not specify the attribute ID for a `search` operation, the search is done on all of the attributes.

-weight *weight*
A numeric value that is the weight of the attribute.

Risk-based access uses the weight to calculate the risk score.

-device *true|false*
Indicates whether the attribute must be stored as part of the device fingerprint.

The default value of this parameter is `true`.

-session *true|false*
Indicates whether the attribute must be stored as a session attribute.

The default value of this parameter is `true`.

-behavior *true|false*
Indicates whether the attribute must be stored as a behavior attribute in the historical storage area.

The default value of this parameter is `false`.

-fileId *response_file_path*
The path of the input or output response file:

- To output an XML response file template for the **manageRbaRiskProfile** command, use the `createResponseFile` operator with the **fileId** parameter.
- To input an XML response file to configure attributes, use the `create`, `update`, `delete`, or `search` operators with the **fileId** parameter. The specified response file must contain the information that you would typically specify on the command line. Edit the XML response file with a text editor to specify the values that you require. For more information, see "Response file example for manageRbaRiskProfile command" on page 76.

Note: Use the **fileId** parameter only with the **operator** parameter. Any other parameters that you specify are ignored.

Examples

The following examples show the correct syntax for various operations of the **manageRbaRiskProfile** command:

Create an attribute:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation create -attributeId browser
     -weight 75}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation create -attributeId browser
     -weight 75]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'create', '-attributeId', 'browser',
     '-weight', '75'])
```

Search for a specific attribute:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation search -attributeId browser}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation search -attributeId browser]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'search', '-attributeId', 'browser'])
```

List all attributes:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation search}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation search]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'search'])
```

Modify the weight of an existing attribute:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation update -attributeId browser
     -weight 25}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation update -attributeId browser
     -weight 25]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'update', '-attributeId', 'browser',
     '-weight', '25'])
```

Delete an attribute:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation delete -attributeId browser}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation delete -attributeId browser]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'delete', '-attributeId', 'browser'])
```

Create a response file template for this command:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation createResponseFile -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation createResponseFile
     -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'createResponseFile',
     '-fileId', '/tmp/resp.xml'])
```

Create several attributes with a response file as input:

- Using Jacl:

```
$AdminTask manageRbaRiskProfile {-operation create -fileId /tmp/resp.xml}
```

- Using Jython string:

```
$AdminTask manageRbaRiskProfile('[-operation create -fileId /tmp/resp.xml]')
```

- Using Jython list:

```
$AdminTask manageRbaRiskProfile(['-operation', 'create', '-fileId', '/tmp/resp.xml'])
```

Note: After you use the **manageRbaRiskProfile** command to change attributes, you must run the **manageRbaConfiguration** command with the **reload** option for the changes to take effect.

Response file example for manageRbaRiskProfile command

Use a response file to run batch operations for the **manageRbaRiskProfile** command. The response file contains information that you would normally specify on the command line. Using a response file helps you automate your configuration process.

To create a response file template, use the **fileId** parameter with the **createResponseFile** operator of the **manageRbaRiskProfile** command. Edit the response file template to specify the values that you require. For more information about the **fileId** parameter, see "manageRbaRiskProfile command" on page 73.

The following response file example shows how to create an attribute called `browser` and specify its weight:

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0" class="java.beans.XMLDecoder">
 <object class="java.util.ArrayList">
  <void method="add">
   <object class="com.tivoli.am.rba.cli.CLIConfigParameters">
```

```
    <void property="attrId">
     <string>browser</string>
    </void>
    <void property="weight">
     <string>80</string>
    </void>
   </object>
  </void>
 </object>
</java>
```

manageRbaSessions command

Use the **manageRbaSessions** command to manage the session attributes that the attribute collection service collects. The session attributes of the user are stored temporarily until the session times out. However, if the device is registered, the session attributes are also stored as part of the device fingerprint.

Purpose

Use the **manageRbaSessions** command to list, search for, or delete session attributes.

Syntax

```
$AdminTask manageRbaSessions {-operation operator [-userId unique_user_ID]}
```

Parameters

Use the following parameters with the **manageRbaSessions** command:

-operation *operator*
Specifies the operation for the session attributes. This parameter is required.

The valid values are `search` and `delete`.

Table 11. Required and optional parameters for operators of the **manageRbaSessions** *command*

Operator	Required parameters	Optional parameters
`search`	Lists all session attributes or searches for the session attributes of a specified user.	**-userId**
`delete`	Deletes the session attributes of a specified user or all session attributes from the database.	**-userId**

-userId *unique_user_ID*
The unique ID of the user.

Examples

The following examples show the correct syntax for various operations of the **manageRbaSessions** command:

List all of the session attributes:

- Using Jacl:
  ```
  $AdminTask manageRbaSessions {-operation search}
  ```
- Using Jython string:
  ```
  $AdminTask manageRbaSessions('[-operation search]')
  ```

- Using Jython list:

```
$AdminTask manageRbaSessions(['-operation', 'search'])
```

Search for the session attributes of a specified user:

- Using Jacl:

```
$AdminTask manageRbaSessions {-operation search -userId myname}
```

- Using Jython string:

```
$AdminTask manageRbaSessions('[-operation search -userId myname]')
```

- Using Jython list:

```
$AdminTask manageRbaSessions(['-operation', 'search', '-userId', 'myname'])
```

Delete all of the session attributes:

- Using Jacl:

```
$AdminTask manageRbaSessions {-operation delete}
```

- Using Jython string:

```
$AdminTask manageRbaSessions('[-operation delete]')
```

- Using Jython list:

```
$AdminTask manageRbaSessions(['-operation', 'delete'])
```

Delete the session attributes of a specified user:

- Using Jacl:

```
$AdminTask manageRbaSessions {-operation delete -userId myname}
```

- Using Jython string:

```
$AdminTask manageRbaSessions('[-operation delete -userId myname]')
```

- Using Jython list:

```
$AdminTask manageRbaSessions(['-operation', 'delete', '-userId', 'myname'])
```

Configuration properties

Specify properties to configure various risk-based access components, such as the database, attribute collection service, runtime security service, and attribute matchers.

You can use the "manageRbaConfiguration command" on page 63 to create and configure the following risk-based access properties.

Important: After you use the **manageRbaConfiguration** command to change configuration properties, you must restart WebSphere Application Server for the changes to take effect.

Note: If you configure a property with a name that ends with `password` or `pwd`, the value of the property is obfuscated when it is stored in the database.

Properties for attribute collection service

ac.uuid

The name of the cookie that stores the correlation ID.

The correlation ID is a UUID that is stored in a cookie, which is used by the attribute collection service. This cookie contains the session data of the user. The session data can be captured even before the user passes the first level of authentication. As the session data cannot always be stored, the cookie stores session data.

The default value is `ac.uuid`.

ac.request.server
The server from which requests are received.

The value is the host name of the server where the `info.js` file is located. The `info.js` file is the JavaScript file of the attribute collection service that is served to the browser of the user.

The default value is `rbademo.ibm.com`.

ac.service.location
The location of the attribute collection service.

The value is the base URL of the IBM Tivoli Federated Identity Manager attribute collection service. It hosts the attribute collection service and the supporting JavaScript and SWF file to capture attributes.

The default value is `http://rbademo.ibm.com/FIM/sps/ac/`.

session.timeout
A numeric value that represents the number of seconds before a risk-based access session automatically expires unless the session is updated. If any attribute in the session is updated, the expiry time for the session is extended by the number of seconds specified in this property.

The default value is `3600`.

ac.get.attributes.enabled
A boolean property that indicates whether access to the `GET` method of the Representational State Transfer (REST) service is allowed.

The default value is `false`.

ac.get.attributes.allowed.clients
A comma-separated list of IP addresses or host names that can access the `GET` method of the REST service.

If this property is not set and the `ac.get.attributes.enabled` property is set to `true`, anyone can access the `GET` method.

If this property is set but the `ac.get.attributes.enabled` property is set to `false`, then this property is ignored.

Properties for attribute value storage

attributes.hash.algorithm
Determines the algorithm that is used to hash the attributes.

For example: `SHA256`

attributes.hash.enabled
A comma-separated list of values that must be hashed.

Properties for auditing

com.tivoli.am.rba.audit.file.name
Specifies how the audit log files are named.

The default prefix of the log file name is `rba_audit_log`.

com.tivoli.am.rba.audit.enableAuditing
Indicates whether auditing is enabled or disabled.

The default value is `true`.

com.tivoli.am.rba.audit.file.location
The location of the audit log file.

The default location is the /rba_audit directory in the WebSphere Application Server default logs directory.

The location that you specify for the com.tivoli.am.rba.audit.file.location property is relative to the default logs directory for WebSphere Application Server.

For example, on a Windows system, if you specify the value as rba, then the log files are stored in the C:\Program Files\IBM\WebSphere\AppServer\profiles*profile_name*\logs\rba directory.

com.tivoli.am.rba.audit.file.size
A numeric value that specifies the maximum audit file size in megabytes (MB).

The default value is 100000.

com.tivoli.am.rba.audit.number.of.files
A numeric value that specifies the maximum number of audit files that are created before the oldest file is overwritten.

The default value is 10.

Properties for database configuration

dbdatasource
The JNDI data source name for database access, which is configured in WebSphere Application Server.

The default value is jdbc/rba. Use this property to specify another JNDI name only if there is a problem with the default value.

dbpassword
A special property so that users can set the password of the database. This property works only if you are using the embedded solidDB database; otherwise it is ignored.

Properties for device registration

subject.id.attribute.name
The name of the attribute in the subject section of the incoming request from EAS, which provides the user ID to risk-based access.

When a device is registered, the attribute provides the ID of the user who is logged in.

The default value is AZN_CRED_PRINCIPAL_NAME.

max.no.of.registered.devices
A numerical value that specifies the maximum number of device fingerprints that can be registered for a user.

The default value is 10.

Valid values are 1 to 20. Values that exceed this range default to 20.

Properties for external authentication interface (EAI) and consent-based registration

Risk-based access provides a ready-to-use EAI implementation that supports consent-based registration. The following properties primarily apply to the consent-based registration function.

eai.header.auth.level
The EAI header for authentication level.

This value must match the value of the corresponding WebSEAL EAI configuration property.

For example: `am-eai-auth-level`.

eai.header.user.id
The EAI header for user ID.

This value must match the value of the corresponding WebSEAL EAI configuration property.

For example: `am-eai-user-id`.

user.consent.attribute.id
The session attribute ID that indicates the consent decision of the user for device registration.

The default value is `http://security.ibm.com/attributes/environment/userConsent`.

device.name
The name of the device as specified by the user.

This value is captured as part of consent-based device registration.

The default value is `device.name`.

consent.application.attribute.id
The application that captures the consent of the user.

The default value is `http://security.ibm.com/attributes/environment/consentApplication`.

Properties for logging and reporting

db.logging.enabled
Indicates whether fine-grained logging is enabled for database SQL statements that are fired by risk-based access.

The default value is `false`.

generate.risk.calculation.reports
Indicates whether risk calculation reports must be generated.

The default value is `false`.

If the value is `true`, the reports are stored in the temporary directory of the WebSphere Application Server Java virtual machine (JVM) in a folder named `rba`. The value of the `java.io.tmpdir` system property indicates the path of the JVM temporary directory. For example, on AIX or Linux systems the files might be stored in the `/tmp/rba` directory.

Properties for matchers

The following sections list the configuration properties that are required for the various ready-to-use matchers and the JavaScript matcher for writing custom matchers.

Properties for IP address matcher

The IP address matcher processes the `ipaddress` attribute.

ip.whitelist
A list of comma-separated IP addresses or subnets that you want to permit.

Include `X.X.X.X` as a value for `-propertyValue` to compare the incoming IP address with the IP address with which the device is registered.

ip.whitelist.netmask
A comma-separated list of netmasks for the IP addresses that are listed in the `ip.whitelist` property.

The list of netmasks must be specified in the same order as the list of IP addresses in the `ip.whitelist` property. The total number of netmasks must correspond exactly to the total number of IP addresses.

ip.blacklist
A comma-separated list of IP addresses that must not be permitted.

ip.blacklist.netmask
A comma-separated list of netmasks for the IP addresses that are listed in the `ip.blacklist` property.

The list of netmasks must be specified in the same order as the list of IP addresses in the `ip.blacklist` property. The total number of netmasks must correspond exactly to the total number of IP addresses.

Properties for the location matcher

The location matcher attribute processes the longitude, latitude, accuracy, and altitude attributes.

location.comparison
Indicates how you want the attribute matcher to calculate the accuracy range of the location coordinates.

Valid values are `closest`, `midpoint`, and `farthest`.

The default value is `midpoint`.

location.allowable.distance
The maximum distance between the new location and the historic locations.

The unit of the numeric value is kilometers.

The default value is `40`.

Properties for the login time matcher

The login time matcher is a behavior-based matcher that uses historical access times to determine the probability of a fraudulent user. The access times are recorded in the risk-based access database.

login.time.probability.threshold
Represents the probability that a user might log in at a particular time.

Valid values are `0` to `1` of double data type.

The default value is `.3`.

Properties for the JavaScript matcher

The JavaScript matcher processes all of the attributes that are identified by the value of the **javascript.attributes** property.

javascript.attributes
A comma-separated list of JavaScript attributes that are processed by the JavaScript attribute matcher.

For example: `userLanguage,fonts,screenHeight,plugins,userAgent`.

Properties for behavior matchers

`min.usage.history.required`
This property controls how many usage data records must be considered as sufficient for behavior matchers such as the login time matcher.

The default value is 8. When less than eight logins are detected in the usage data, then the login time matcher returns an `INDETERMINATE` result, which means that its weight is considered as zero.

`usage.data.cap.per.user`
The maximum number of usage data records that the system can store for each user.

The default value is 1000.

Properties for password obfuscation

Some runtime configuration properties are automatically obfuscated by risk-based access. For example, if you configure a property with a name that ends with `assword` or `pwd`, the value of the property is obfuscated by default when it is stored in the database.

`obfuscator.default`
Identifies the property that specifies the name of the default obfuscator class.

For example: `Password.ClassName.MyObfuscator`.

If the value of this property is not specified, the default obfuscator that is provided with risk-based access is used.

`obfuscator.filename`
The name of Java archive (JAR) file that contains one or more obfuscator classes.

If this file is not found in the WebSphere Application Server `lib` directory, custom obfuscators are loaded. The custom obfuscators are identified by properties with names that resemble `obfuscator.classname.*`. A maximum of 10 properties can be configured to specify various password obfuscators that risk-based access can use.

For example:

```
obfuscator.classname.MyObfuscator=com.xyz.MyObfuscator
obfuscator.classname.1=com.xyz.PasswordObfuscator1
obfuscator.classname.2=com.xyz.PasswordObfuscator2
obfuscator.classname.3=com.xyz.PasswordObfuscator3
```

Properties for policy information points (PIPs)

`pip.attributes.user.fingerprint.count`
Stores the attribute ID of the device count that is computed by the risk-based access `UserFingerprintCountPIP`. This PIP computes the count of devices that are registered for a specific user ID in the risk-based access database.

The default value is `http://security.ibm.com/attributes/subject/registeredDeviceCount`.

`pip.attributes.risk.score`
Stores the attribute ID of the risk score that is computed by the risk-based access `RiskCalculatorPIP`, which computes the risk that is associated with an incoming request.

The default value is `http://security.ibm.com/attributes/resource/riskScore`.

Properties for runtime security service

rtss.admin.ws.host
The host name where the runtime security service is located.

For example: `rbademo.ibm.com`.

rtss.admin.port
The administration port number of the runtime security service.

rtss.admin.wssecurity.enabled
Indicates whether security is enabled for the runtime security service.

rtss.admin.keystore.alias
The alias of the runtime security service keystore.

rtss.admin.keystore.file
The file name of the runtime security service keystore.

rtss.admin.keystore.file.pwd
The password for the runtime security service keystore file.

rtss.admin.keystore.key.pwd
The password to the runtime security service key in the keystore.

rtss.admin.keystore.type
The type of the keystore that is used by the runtime security service server.

rtss.admin.truststore.file
The location of the truststore file for the runtime security service server.

rtss.admin.truststore.file.pwd
The password to the truststore for the runtime security service server.

rtss.admin.truststore.file.type
The type of the truststore.

rtss.admin.basic.authn.username
The user name to access the runtime security service server with basic authentication.

rtss.admin.basic.authn.pwd
The password to access the runtime security service server with basic authentication.

policy.service.name
The policy service name in the runtime security service.

For example: `webseald-localhost-default`.

The service name must match the top-level context ID of all the risk-based access policies. All incoming requests from the runtime security services external authorization service (EAS) in WebSEAL contain the instance name of the WebSEAL server as the context ID. You can customize the service name in the WebSEAL configuration file. If this value does not match the context ID coming from EAS, then the policies are not applied to the incoming requests from the EAS.

Attributes for risk profiles and policies

Risk-based access stores user and device-related data in the form of attributes. These attributes are used to calculate risk. Use the following attributes to author or customize policies and policy information providers (PIPs):

The following tables show all the attributes for risk profiles and policies:

- Authorization credential attributes
- Device fingerprint attributes
- Risk score attributes
- Runtime security services audit attributes
- Session attributes
- User metadata attributes

Table 12. Use and characteristics of policy and context attributes: authorization credential.

Contains information about the identity and authorization capability of the subject of the credential.

Stored in the authorization credential of the user.

Attributes	Category	Data type	Description
AUTHENTICATION_LEVEL	Subject	String	Authentication level of a user.
AZN_CRED_AUTHNMECH_INFO	Subject	String	Authentication mechanism.
AZN_CRED_AUTHZN_ID	Subject	String	Registry name of the user.
AZN_CRED_AUTH_METHOD	Subject	String	Method of authentication used. For example: password.
AZN_CRED_BROWSER_INFO	Subject	String	User agent.
AZN_CRED_GROUPS	Subject	String	Groups that the user is a member of.
AZN_CRED_GROUP_REGISTRY_IDS	Subject	String	Registry IDs of the groups the user is a member of.
AZN_CRED_GROUP_UUIDS	Subject	String	Universally Unique Identifer (UUID) of the groups the user is a member of.
AZN_CRED_IP_FAMILY	Subject	String	IP family.
AZN_CRED_MECH_ID	Subject	String	Mechanism ID.
AZN_CRED_NETWORK_ADDRESS_BIN	Subject	String	Binary representation of the IP.
AZN_CRED_NETWORK_ADDRESS_STR	Subject	String	String representation of the IP
AZN_CRED_PRINCIPAL_DOMAIN	Subject	String	Domain that the user is authenticated in.
AZN_CRED_PRINCIPAL_NAME	Subject	String	User name.
AZN_CRED_PRINCIPAL_UUID	Subject	String	UUID of the user.
AZN_CRED_QOP_INFO	Subject	String	Quality of protection information.
AZN_CRED_REGISTRY_ID	Subject	String	ID of the user in the registry.
AZN_CRED_USER_INFO	Subject	String	Extra information about the user. For example: name.
AZN_CRED_VERSION	Subject	String	Credential version.
tagvalue_login_user_name	Subject	String	Credential attribute that records the user name. For example: sec_master.
tagvalue_session_index	Subject	String	Session index in the user credential.
tagvalue_user_session_id	Subject	String	Name and value of the user session ID in the user credential.
urn:oasis:names:tc:xacml:1.0:subject:group-id	Subject	String or X500Name	Groups that the user is a part of. For example: **cn=SecurityGroup,1.3.6.1.4.1.4228.1.3=Default1,dc=ibm,dc=com**

Table 12. Use and characteristics of policy and context attributes: authorization credential (continued).

Contains information about the identity and authorization capability of the subject of the credential.

Stored in the authorization credential of the user.

Attributes	Category	Data type	Description
`urn:oasis:names:tc:xacml:1.0:subject:subject-id`	Subject	String or X500Name	Identity of the requester. For example: **`cn=SecurityMaster,secAuthority=Default1,dc=ibm,dc=com`**

Table 13. Use and characteristics of policy and context attributes: device fingerprint.

Retrieved from standard HTTP headers.

Configure the attributes that you want to retrieve in the `[azn-decision-info]` stanza of the WebSEAL configuration file.

The configured attributes are provided when an access request is routed from the external authorization service (EAS) to the runtime security service. For more information, see Runtime security services external authorization service.

Attributes	Category	Data type	Description
`header:Accept`	Environment	String	Content-types that acceptable.
`header:Accept-Charset`	Environment	String	Character sets that are acceptable.
`header:Accept-Encoding`	Environment	String	Encoding types that are acceptable.
`header:Accept-Language`	Environment	String	Languages that are acceptable.
`header:Authorization`	Environment	String	Authentication credentials.
`header:Cache-Control`	Environment	String	Used to specify the directives that must be obeyed by all caching mechanisms along the request chain and response chain.
`header:Connection`	Environment	String	Type of connection that the user-agent prefers.
`header:Content-Type`	Environment	String	MIME type of the request body.
`header:Host`	Environment	String	Requested host.
`header:Pragma`	Environment	String	Implementation-specific headers that might have various effects anywhere along the request-response chain.
`header:rspcode`	Environment	String	Response code.
`header:Transfer-Encoding`	Environment	String	Encoding type of the response.
`header:User-Agent`	Environment	String	String that gives you information about the environment from which you access the resource. For example: browser information or version information.
`header:X-Requested-With`	Environment	String	Identifies Ajax requests.
`method`	Environment	String	Type of request that is made. For example: `GET` or `POST`.
`scheme`	Environment	String	Type of protocol that the request is using. For example: `HTTP` or `HTTPS`.
`uri`	Environment	String	Requested resource. The query string is included with the URI.

Table 14. Use and characteristics of policy and context attributes: risk score. Contains the calculated risk score.

Attributes	Category	Data type	Description
`http://security.ibm.com/attributes/resource/riskScore`	Resource	Integer	Calculated risk score.

Table 15. Use and characteristics of policy and context attributes: Runtime security services audit.

Used by the runtime security services policy decision point (PDP).

Not used for authoring policies.

Attributes	Category	Data type	Description
urn:oasis:names:tc:xacml:1.0:action:action-id	Action	String	Action that is performed on the resource. For example: read.
urn:oasis:names:tc:xacml:1.0:environment:current-date	Environment	Date	Date of the request.
urn:oasis:names:tc:xacml:1.0:environment:current-time	Environment	Time	Time of the request.
urn:oasis:names:tc:xacml:1.0:resource:resource-id	Resource	String	ID of the resource that is accessed.
urn:oasis:names:tc:xacml:1.0:subject:subject-id	Subject	String	Subject that accesses the resource.

Table 16. Use and characteristics of policy and context attributes: session. Uses the functions that are provided in the info.js file to collect session attributes. Complete the following steps to employ:

1. Embed JavaScript code that calls the functions in the info.js file in the landing page of your application.
2. Use the **manageRbaRiskProfile** command to configure the attributes and their weights for risk score calculation.

Attributes	Category	Data type	Description
accessTime	Environment	String	Time that a request is made.
accuracy	Environment	String	Radius around the latitude and longitude that contains the actual location of the device in meters.
altitude	Environment	String	Altitude of the device.
availableHeight	Environment	String	Indicates the height of the requesting device's screen without the Windows toolbar.
availableWidth	Environment	String	Indicates the width of the requesting device's screen without the Windows toolbar.
colorDepth	Environment	String	Indicates the bit depth of the device's color palette.
fonts	Environment	String	Determines all of the installed fonts in the browser.
ipaddress	Environment	String	Contains the IP address of the incoming request.
language	Environment	String	Determines the language that the device is using.
latitude	Environment	String	Latitude of the device that is based on the w3c geolocation standard.
longitude	Environment	String	Longitude of the device that is based on the w3c geolocation standard.
platform	Environment	String	Determines the operating system (OS) the device is running.
plugins	Environment	String	All of the installed plug-ins in the browser.
screenHeight	Environment	String	Indicates the height of the screen of the requesting device.
screenWidth	Environment	String	Indicates the width of the screen of the requesting device.
urn:oasis:names:tc:xacml:1.0:action:action-id	Action	String	Action that must be taken on the resource. Typically, the action is an HTTP method, such as GET, POST, or OPTIONS.
urn:oasis:names:tc:xacml:1.0:resource:resource-id	Resource	String	URI of the resource on which the action must be taken. Does not include the query string. Usually the same as the **uri** attribute.
userAgent	Environment	String	Indicates the user agent.

Table 17. Use and characteristics of policy and context attributes: user metadata. Contains information about registered attributes.

Attributes	Category	Data type	Description
`http://security.ibm.com/attributes/subject/registeredDeviceCount`	Subject	Integer	Number of registered user devices.

Chapter 8. Error message reference

Messages indicate events that occur during the operation of the system. Depending on their purpose, messages might be displayed on the screen. By default, all informational, warning, and error messages are written to the message logs. You can review the logs later to determine what events occurred, see what corrective actions were taken, and audit the actions taken.

The following types of messages are used:

Informational messages
Indicate conditions that are worthy of noting but that do not require you to take any precautions or actions.

Warning messages
Indicate that a condition is detected, about which you must be aware, but does not necessarily require any action.

Error messages
Indicates that a condition occurred, which requires your action.

Use the explanation of the error messages to troubleshoot problems that might occur.

DPWBA0300W A general error occurred: *exception message*.

Explanation: An error occurred when attempting to process the authorization decision request. The WebSEAL logs might contain more information.

Administrator response: Check the WebSEAL logs for more details.

DPWBA0303E A required configuration entry, *entry name*, **under the** *stanza name* **stanza is missing from the configuration file.**

Explanation: See message.

Administrator response: Add the specified missing entry to the configuration file.

DPWBA0304E Unable to determine whether the IBM Tivoli Access Manager policy needs to be applied: *error code*.

Explanation: An error occurred when parsing the IBM Tivoli Access Manager policy value as Boolean. The WebSEAL trace logs might contain more details.

Administrator response: Ensure that the configuration value for setting the IBM Tivoli Access Manager policy is set to either true or false.

DPWBA0305E Initialization of cluster manager failed: *error code*.

Explanation: This error might be due to a missing configuration entry for the runtime security services cluster definition.

Administrator response: Verify that all configuration entries are present and have correct values.

DPWBA0306E Failed to add cluster to cluster manager: *error code*.

Explanation: This error might be due to a missing configuration entry for the runtime security services cluster definition.

Administrator response: Verify that all configuration entries are present and have correct values.

DPWBA0307W The stanza *stanza name* **in file** *file name* **is not found.**

Explanation: An error occurred when looking up the specified stanza in the file.

Administrator response: Ensure that the specified stanza is present.

DPWBA0308W The header key name is missing for the app_context_data key: *key*.

Explanation: A header key name (the custom ones specified on the LHS by users) is missing in the configuration file, but is present in an IBM Tivoli Access Manager structure.

Administrator response: Check the WebSEAL logs for more details.

DPWBA0309W The authentication level is missing for the obligation: *obligation*.

Explanation: An entry is missing in the [obligations-levels-mapping] stanza in WebSEAL.

Administrator response: Add the missing entry that maps the obligation to its authentication level and restart WebSEAL.

DPWBA0310W An error occurred when trying to retrieve the obligation ID from the runtime security services response.

Explanation: The obligation ID sent by the runtime security services could not be parsed successfully. The error is due to an issue either with the runtime security services server or the runtime security services external authorization service (EAS).

Administrator response: If the problem persists, contact IBM support.

DPWBA0311W Unable to contact runtime security services.

Explanation: Unable to make a SOAP call to the runtime security service, which might be because the service is down or due to a malformed request.

Administrator response: Verify that the runtime security services server is up and running. Also check the WebSEAL logs for more details.

DPWBA0312W The *attribute name* attribute could not be extracted from a credential: API error: *error string* (API error code [*major error code*:*minor error code*]).

Explanation: The specified attribute could not be extracted from a credential. This error might be due to resource exhaustion, and as such be transient.

Administrator response: If the problem persists, check IBM Electronic Support for additional information at http://www.ibm.com/software/sysmgmt/products/support/index.html?ibmprd=tivman.

DPWBA0313E Cannot allocate memory.

Explanation: Memory allocation operation failed.

Administrator response: Check memory limits on your machine, and increase available memory if possible.

DPWBA0314E The URL is invalid.

Explanation: A client request contained a URL that does not conform to HTTP specifications.

Administrator response: Verify the request from the client and ensure that it conforms to HTTP specifications.

FBTRBA001E A database error occurred.

Explanation: An unrecoverable database error occurred.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA002E An error occurred when managing the policy.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA003E An error occurred during command execution.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA005E A required parameter *parameter name* is missing or invalid.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA007E The policy file does not exist.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Specify a policy file that exists and run the command again.

FBTRBA008E Creation of database connection failed. Check the database configuration and network connectivity to the database server.

Explanation: The database connection could not be created.

System action: Command execution is halted.

Administrator response: Ensure that the database is configured correctly. Also check that the network connectivity to the database server is available.

FBTRBA009E Unable to modify the application parameter *task or role name***.**

Explanation: An attempt to locate and modify a particular set of application parameters failed during deployment.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA010E The IBM Tivoli Federated Identity Manager runtime is not deployed. Deploy the IBM Tivoli Federated Identity Manager runtime before continuing.

Explanation: The risk-based access runtime requires that the IBM Tivoli Federated Identity Manager runtime be deployed first.

System action: Command execution is halted.

Administrator response: Deploy the IBM Tivoli Federated Identity Manager runtime before proceeding.

FBTRBA011E The risk-based access deployment failed.

Explanation: An error occurred during risk-based access deployment.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA012E The risk-based access deployment failed because it could not determine the directory in which IBM Tivoli Federated Identity Manager is installed.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA013E The risk-based access deployment failed because the runtime security services EAR is not found at the following location: *RTSS Ear path***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA014E The risk-based access deployment failed because the following configuration directory could not be created: *Configuration Directory***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA017E The risk-based access deployment failed because the configuration repository directory could not be determined.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA018E The risk-based access deployment failed because the runtime security services archive file is not found at the following location: *RTSS archive path***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA019E The risk-based access deployment task failed because the risk-based access runtime security services plugins directory is not found at the following location: *rba plugins path***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA020E The risk-based access deployment failed because a required file is not found at the following location: *path to file***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA021E An error occurred during the risk-based access redeployment. Check the application server logs for more details.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA028E Deserialization of the response file failed.

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the XML response file to verify that it is valid and try again.

FBTRBA029E The risk-based access deployment failed because the risk-based access JavaScript directory is not found at the following location: *rbajavascript path***.**

Explanation: See message.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA049E The runtime property ac.request.server is not configured.

Explanation: To make cross-domain AJAX requests, the runtime property ac.request.server must be configured.

System action: The CORS headers are not set in the HTTP response.

Administrator response: Configure the runtime property ac.request.server.

FBTRBA057E The attribute string is formatted incorrectly.

Explanation: Attributes must be formatted as key=value and separated by the specified delimiter or a percent symbol (%).

System action: Command execution is halted.

Administrator response: Ensure that the attribute string is formatted correctly.

FBTRBA058E The attribute name, *name***, is invalid and is not configured.**

Explanation: The attribute validation failed because the attribute is not configured.

System action: Command execution is halted.

Administrator response: Configure the attribute.

FBTRBA060E The policies are not exported to *location***.**

Explanation: An error occurred when exporting policies to the specified location.

System action: Command execution is halted.

Administrator response: Check the server logs for more details to trace the cause of the error.

FBTRBA061E An error occurred when parsing the XACML rules file.

Explanation: The XACML rules file could not be parsed successfully, probably due to improper syntax.

System action: Command execution is halted.

Administrator response: Check the syntax of the XACML rules file and try again. Also check the server logs for more details to trace the cause of the error.

FBTRBA062E An unknown operation *operation name* **is specified.**

Explanation: An invalid operation is specified by the user.

System action: Command execution is halted.

Administrator response: Run the command with a correct operation as specified in the documentation.

FBTRBA065E Reloading of the configuration failed.

Explanation: A subcomponent returned an error during the reload attempt.

System action: Command execution is halted.

Administrator response: Check the logs for more details to trace the cause of the error.

FBTRBA066E The device ID is invalid.

Explanation: The device ID is not formatted correctly. It must be an integer value.

System action: Command execution is halted.

Administrator response: Verify the device ID.

FBTRBA069E The type for the attribute *id* **is not specified.**

Explanation: An attribute and its type must be specified must be specified before referencing the attribute. Valid types are integer, double, string, time, or date.

System action: Command execution is halted.

Administrator response: Specify the type for the

attribute in the XACML rules file.

FBTRBA075E The *operation* operation is not allowed because runtime security service was installed by IBM Tivoli Security Policy Manager.

Explanation: The runtime security service was deployed by IBM Tivoli Security Policy Manager; so policy creation and deletion must be done using IBM Tivoli Security Policy Manager.

System action: Command execution is halted.

Administrator response: Use IBM Tivoli Security Policy Manager to create, delete, or update policies.

FBTRBA077E Service name missing. To specify the service name, use the -serviceName parameter, or add *serviceNameConfigPropertyName* to the risk-based access configuration.

Explanation: The default service name was not configured, and a value was not provided through the serviceName parameter.

System action: Command execution is halted.

Administrator response: Add a default service name to the risk-based access configuration, or specify one using the the serviceName parameter.

FBTRBA078E The risk-based access deployment task failed because the risk-based access matchers directory was not found at this location: *rba matchers path*

Explanation: The risk-based access deployment encountered an error and could not continue.

System action: The risk-based access deployment task is halted.

Administrator response: Check the system logs for more details and ensure that the risk-based access installation step has completed.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law :

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to

IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Glossary

This glossary includes terms and definitions for risk-based access.

The following cross-references are used in this glossary:

- See refers you from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
- See also refers you to a related or contrasting term.

To view glossaries for other IBM products, go to www.ibm.com/software/globalization/terminology (opens in new window).

A

access control
In computer security, the process of ensuring that users can access only those resources of a computer system for which they are authorized.

access control list (ACL)
In computer security, a list associated with an object that identifies all the subjects that can access the object and their access rights.

ACL See access control list.

attribute
A characteristic or trait of an entity that describes the entity. See also risk-related attribute.

attribute configuration
In risk-based access, the configuration of a risk attribute to specify its weight and indicate whether the attribute is required for risk calculation. See also risk-related attribute.

attribute list
A linked list that contains extended information that is used to make authorization decisions. Attribute lists consist of a set of key-value pairs.

attribute collection service
In risk-based access, a Representational State Transfer (REST) service that collects web browser and location information from the user.

attribute matcher
In risk-based access, a feature that compares the attribute values in the incoming device fingerprint with the existing device fingerprint of the user. The ready-to-use attribute matchers that are provided with risk-based access are the location matcher, IP address matcher, and login time matcher.

audit event
A record of an operation in the audit log or change history; for example, an audit entry is created when an attribute or property is modified.

audit log
A log file that contains a record of system events and responses. The audit log maintains the history of significant operations that modify metadata or configuration data, including operations that fail to complete successfully.

authentication
In computer security, the process that verifies identity and provides proof that a user of a computer system is genuinely who that person claims to be. Common mechanisms for implementing this service are passwords and digital signatures. Authentication is distinct from authorization; authorization is concerned with granting and denying access to resources.

authorization
The process of granting a user, system, or process either complete or restricted access to an object, resource, or function.

authorization policy generation language
In risk-based access, a simplified format that specifies the policy rules in a script, which can be used to generate an authorization policy for risk-based access.

authorization service
A dynamic or shared library that can be loaded by the authorization API runtime client at initialization time. It is used for

operations that extend a service interface in the Authorization API.

automatic device registration
See silent device registration.

B

batch operation
A predefined group of processing actions that are submitted to the system. The actions are taken with little or no interaction between the user and the system.

behavioral pattern
In risk-based access, the pattern derived from the historical access data of a user. For example, a behavioral pattern can indicate that a specified user regularly accesses a protected resource only during business hours on weekdays.

C

challenge
A request for certain information to a system. The information, which is sent back in response to this request, is necessary for authentication.

class path
A list of directories and JAR files that contain resource files or Java classes that a program can load dynamically at run time.

configuration property
In risk-based access, a property that specifies the configuration for various components of risk-based access. For example, properties are used to configure the database, attribute collection service, runtime security services, and attribute matchers.

consent-based device registration
In risk-based access, the process of registering the device fingerprint only if the user consents to have the device registered. The risk-based access policy can be configured to present an HTML form that requests the consent of the user before the device is registered.

cookie Information that a server stores on a client system and accesses during subsequent sessions. Cookies allow servers to remember specific information about clients.

correlation ID
A UUID that is stored in a cookie.

credentials
Detailed information that is acquired during authentication, which describes the user, any group associations, and other security-related identity attributes. For example, a user ID and password are credentials that allow access to network and system resources.

D

data source
The means by which an application accesses data from a database.

DB2 A family of IBM licensed programs for relational database management.

demilitarized zone (DMZ)
A configuration that includes multiple firewalls to add layers of protection between a corporate intranet and a public network, such as the Internet.

deploy
To place files or install software into an operational environment. In Java Platform, Enterprise Edition (Java EE), deploying involves creating a deployment descriptor suitable to the type of application that is being deployed.

device fingerprint
In risk-based access, a set of attributes that is part of the request that comes from a device. A device fingerprint consists of a user ID and a set of key-value pairs of attributes that identify a particular user.

device registration
In risk-based access, the process of storing the incoming device fingerprint in the database. See also device fingerprint.

DMZ See demilitarized zone.

domain
1. An object, icon, or container that contains other objects, which represent the resources of a domain. The domain object can be used to manage those resources.

2. A logical grouping of resources in a network for common management and administration. For example, a deployment of the Tivoli Federated Identity Manager runtime component on WebSphere Application Server.

E

EAS See external authorization service.

element
In markup languages, a basic unit that consists of a start tag, end tag, attributes and their values, and any text between the start and end tags.

event Any significant change in the state of a system resource, network resource, or network application. An event can be generated for a problem, for the resolution to a problem, or for the successful completion of a task.

eXtensible Access Control Markup Language (XACML)
A standard for access control policy language. It provides a syntax that is defined in XML for managing access to resources.

Extensible Markup Language (XML)
A standard meta-language for defining markup languages that is based on Standard Generalized Markup Language (SGML).

external authorization service (EAS)
An authorization API runtime plug-in that can be used to make application- or environment-specific authorization decisions as part of the authorization decision chain. Risk-based access uses the runtime security services EAS plug-in to enforce policy decisions.

F

failover
An automatic operation that switches to a redundant or standby system in the event of a software, hardware, or network interruption.

fix pack
A cumulative collection of fixes that is made available between scheduled refresh packs, manufacturing refreshes, or releases. It is intended to allow customers to move to a specific maintenance level.

H

hash In computer security, a number generated from a string of text that is used to ensure that transmitted messages arrived intact.

hashing
The process of encoding a character string as a fixed-length bit string for comparison.

host name
In Internet communication, the name that is given to a computer. The host name might be a fully qualified domain name, such as mycomputer.city.company.com, or it might be a specific subname, such as mycomputer.

HTTP See hypertext transfer protocol.

hypertext transfer protocol (HTTP)
In the Internet suite of protocols, the protocol that is used to transfer and display documents.

I

identifier
The name of an item in a program that is written in the Java language.

Internet protocol (IP)
In the Internet suite of protocols, a connectionless protocol that routes data through a network or interconnected networks. IP acts as an intermediary between the higher protocol layers and the physical network.

IP See internet protocol.

J

Java Database Connectivity (JDBC)
An industry standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC interface provides a call level interface for SQL-based and XQuery-based database access.

Java Naming and Directory Interface (JNDI)
An extension to the Java platform that

provides a standard interface for heterogeneous naming and directory services.

JDBC See Java Database Connectivity.

JNDI See Java Naming and Directory Interface.

junction
A logical connection that is created to establish a path from one server to another.

K

key-value pair
Information that is expressed as a paired set.

N

naming service
An implementation of the Java Naming and Directory Interface (JNDI) standard.

O

object In object-oriented design or programming, a concrete realization (instance) of a class that consists of data and the operations associated with that data. An object contains the instance data that is defined by the class, but the class owns the operations that are associated with the data.

one-time password (OTP)
A password that is valid for a single login session or transaction.

OTP See one-time password.

P

PDP See policy decision point.

PEP See policy enforcement point.

permission
Authorization for activities, such as reading and writing local files, creating network connections, and loading native code.

persist
To be maintained across session boundaries, typically in nonvolatile storage, such as a database system or a directory.

persistence
A characteristic of data that is maintained across session boundaries. Data persistence is typically available in nonvolatile storage, such as a database system.

PIP See policy information point.

plug-in
A separately installable software module that adds function to an existing program, application, or interface.

policy

1. A set of considerations that influence the behavior of a managed resource or a user.
2. A set of conditions that, after they are evaluated, determine access decisions.

policy decision point (PDP)
A policy decision component that evaluates and determines the results an access request.

policy enforcement point (PEP)
A policy decision component that receives a request, notifies the policy decision point of the request, receives a decision, and enforces the decision.

policy generation language
See authorization policy generation language.

policy information point (PIP)
A policy decision component that provides additional information about a request.

POP See protected object policy.

port As defined in a Web Services Description Language (WSDL) document, a single endpoint that is defined as a combination of a binding and a network address.

port number
In Internet communications, the identifier for a logical connector between an application entity and the transport service.

profile
Data that describes the characteristics of a user, group, resource, program, device, or remote location.

property
A characteristic of an object that describes

the object. A property can be changed or modified. Properties can describe an object name, type, value, or behavior, among other things.

protected object policy (POP)
A security policy that imposes additional conditions on the operation permitted by the ACL policy to access a protected object. It is the responsibility of the resource manager to enforce the POP conditions.

pull A network operation that initiates an action by requesting the action from a resource.

push A network operation that sends information to resources.

R

Repository
A persistent storage area for data and other application resources.

Representational State Transfer (REST)
A software architectural style for distributed hypermedia systems like the World Wide Web. REST is a simple interface that uses XML (or YAML, JSON, plain text) over HTTP without an additional messaging layer, such as SOAP. See also RESTful.

response file
A file that contains predefined values, such as parameters and values used to control the actions of a component in a predetermined manner. For example, in risk-based access, the XML response file for a command contains the information that you would otherwise specify on the command line.

REST See Representational State Transfer.

RESTful
Pertaining to applications and services that conform to Representational State Transfer (REST) constraints. See also Representational State Transfer.

risk-based access
A pluggable and configurable component for IBM Tivoli Access Manager for e-business. Risk-based access provides access decision and enforcement that is based on a dynamic risk assessment or confidence level of a transaction.

Risk-related attribute
In risk-based access, an attribute that is used to calculate the risk score. Risk-related attributes are collected from HTTP headers, captured by attribute collection service, or with custom JavaScript.

risk score
In risk-based access, a numerical value that represents the amount of risk that is associated with a request to access a protected resource. The risk score indicates the confidence level that the user who is using the device is actually the authenticated user. The risk score is computed on a scale of 1 to 100. The risk score determines whether access to a protected resource is permitted, denied, or permitted with further authentication proof.

run time
The time period during which a computer program is running.

runtime environment
A subset of an application development kit (ADK) that contains the executable files and other supporting files that comprise the operational environment of the platform.

runtime security services
A software component that, depending on its configuration, can do the functions of a policy decision point, policy enforcement point, and policy distribution target.

runtime security services EAS
In risk-based access, a plug-in that provides the policy enforcement point (PEP) functionality.

rule A condition that is used in the evaluation of a policy.

S

schema
The set of statements, expressed in a data definition language, that completely describes the structure of data that is stored in a database, directory, or file. A schema provides a logical classification of database objects.

script A series of commands, combined in a file,

that carry out a particular function when the file is run. Scripts are interpreted as they are run.

secondary challenge
In risk-based access, the additional challenge that is presented to the user for authentication when the risk score is high. See also challenge.

security
The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

security policy
A written document that defines the security controls that you institute for your computer systems. A security policy describes the risks that you intend these controls to minimize and the actions that must be taken if someone breaches your security controls.

service
A resource, such as a web service, to which access requests are made and that can be protected by policies.

session
In Java EE, an object used by a servlet to track user interaction with a web application across multiple HTTP requests. A session consists of a series of requests to a server or application that originate from the same user at the same browser.

session attribute
In risk-based access, an attribute that the attribute collection service captures during a user session. The session attributes are stored in the risk-based access database when the device of the user is registered.

shared library file
A file that consists of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries. Applications that are deployed on the same node as this file can access this information.

silent device registration
In risk-based access, the process of registering the device fingerprint after the user successfully authenticates with a secondary challenge. Silent registration does not require any further interaction or consent from the user. See also secondary challenge, device registration, and consent-based device registration.

SOAP A lightweight, XML-based protocol for exchanging information in a decentralized, distributed environment. SOAP can be used to query and return information and invoke services across the Internet.

stanza A group of lines in a file that together have a common function or define a part of the system. Stanzas are separated by blank lines or colons, and each stanza has a name.

step-up authentication
A protected object policy (POP) that relies on a preconfigured hierarchy of authentication levels. It enforces a specific level of authentication according to the policy set on a resource. The step-up authentication POP does not force the user to authenticate with multiple levels of authentication to access a resource. It requires the user to authenticate at a level at least as high as the level required by the policy that protects a resource. See also protected object policy.

string In programming languages, the form of data used for storing and manipulating text.

syntax The rules for the construction of a command or statement.

T

target The destination for an action or operation.

throughput
The measure of the amount of work done by a device over a time period, for example, the number of jobs per day.

token A particular message or bit pattern that signifies permission or temporary control to transmit over a network.

U

Uniform Resource Identifier (URI)

1. A compact string of characters for identifying an abstract or physical resource.
2. A unique address that is used to identify content on the web, such as a

page of text, video, sound clip, image, or program. The most common form of URI is the web page address, which is a particular form or subset of URI called a Uniform Resource Locator (URL). A URI typically describes how to access the resource, the computer that contains the resource, and the name of the resource.

Uniform Resource Locator (URL)
The unique address of an information resource that is accessible in a network, such as the Internet. The URL includes the abbreviated name of the protocol that is used to access the information resource. The URL also includes the information used by the protocol to locate the information resource.

Uniform Resource Name (URN)
A name that uniquely identifies a web service to a client.

Universally Unique Identifier (UUID)
The 128-bit numeric identifier that is used to ensure that two components do not have the same identifier.

URI See Uniform Resource Identifier.

URL See Uniform Resource Locator.

URN See Uniform Resource Name.

UUID See Universally Unique Identifier.

V

variable
A representation of a changeable value.

W

web application
An application that is accessible by a web browser and that provides some function beyond static display of information. Common components of a web application include HTML pages, JSP pages, and servlets.

web server
A software program that can service Hypertext Transfer Protocol (HTTP) requests.

web service
A self-contained, self-describing modular application that can be published, discovered, and invoked over a network that uses standard network protocols. Typically, XML is used to tag the data, and SOAP is used to transfer the data. A WSDL is used for describing the services available, and UDDI is used for listing what services are available.

WebSEAL
A high performance, multi-threaded Web server that applies a security policy to a protected object space. WebSEAL can provide single sign-on solutions and incorporate back-end Web application server resources into its security policy.

weight
In risk-based access, a numerical value that is assigned to an attribute. The weight of an attribute indicates its importance from a risk perspective and is used for calculating the risk score.

X

XACML
See eXtensible Access Control Markup Language.

XML See Extensible Markup Language.

Index

T

U

W

Printed in USA

SC27-4445-02