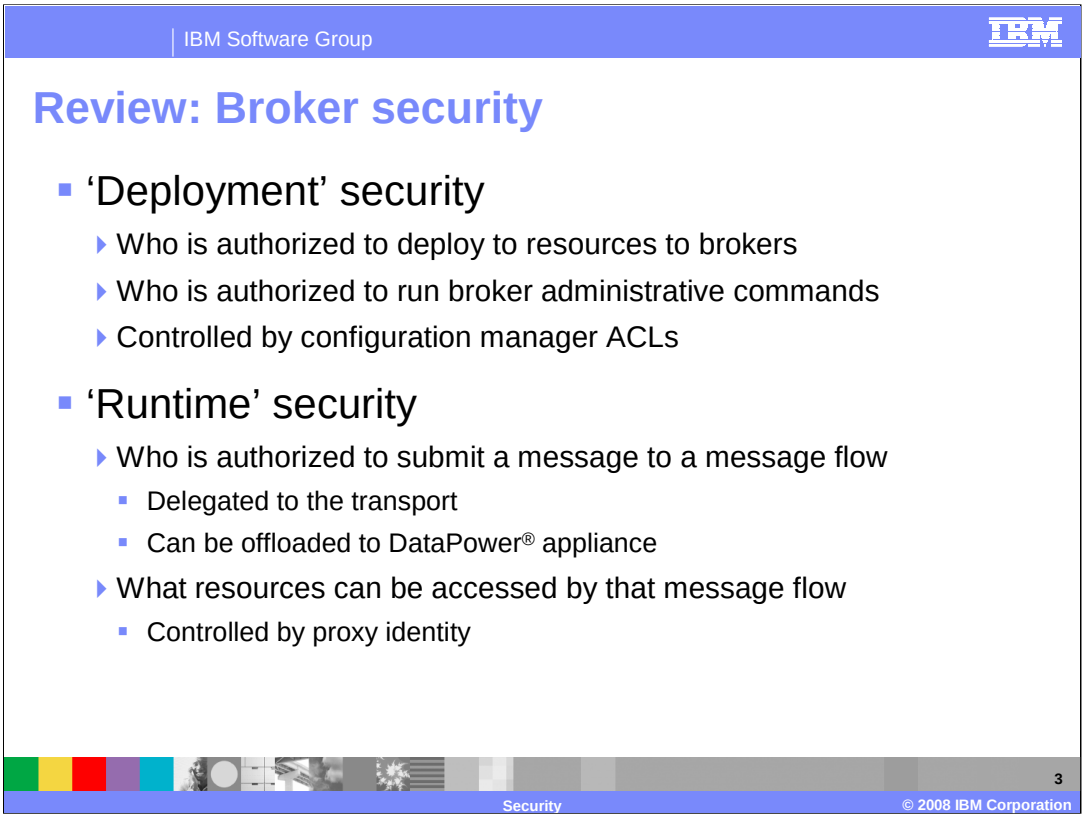
This presentation will give an overall view of the new security functions in WebSphere Message Broker Version 6.1

Agenda

- Review of security in Message Broker Version 6.0
- New runtime security manager in Message Broker Version 6.1

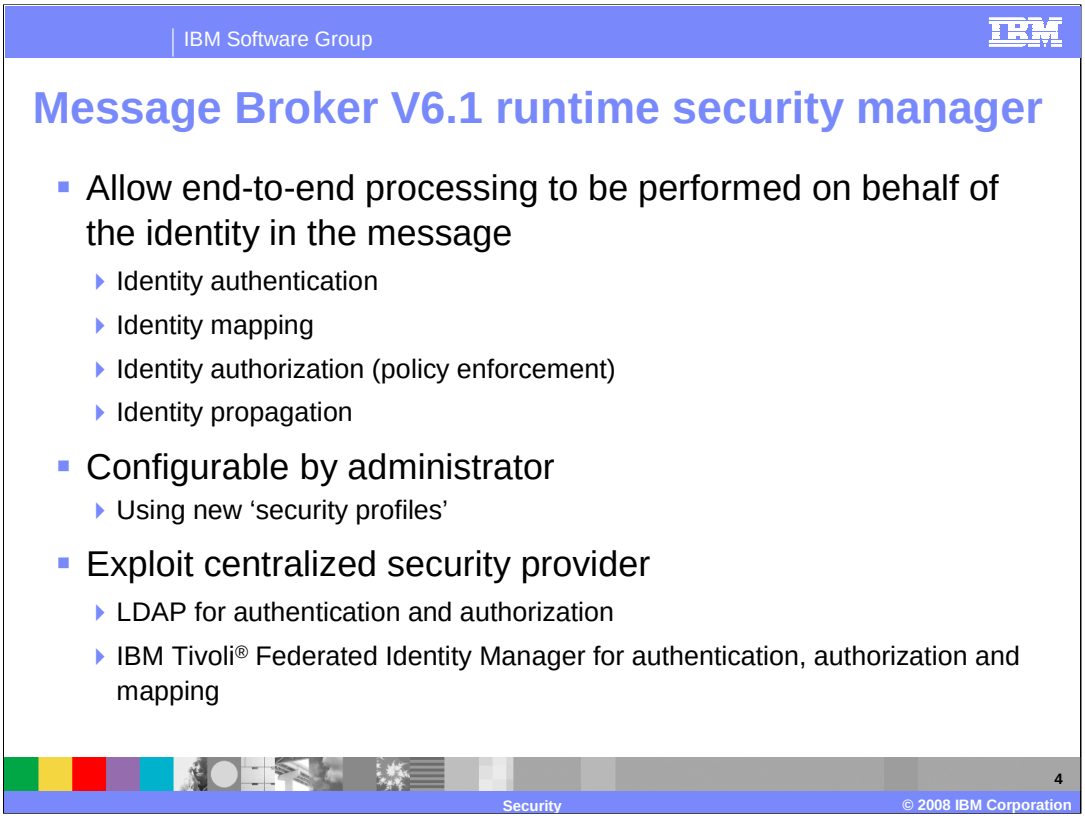
This session will provide a brief reminder of the security functions available in Message Broker Version 6.0, and then go on to cover the new security functions provided in Version 6.1.



Security in WebSphere Message Broker falls into two main areas.

The first is Deployment Security. This concerns the administration of the Message Broker environment, and controls who is allowed to perform operations on the runtime components of the Broker. It concerns the Broker administrative commands, and this access is managed by the configuration manager access controls lists, or ACLs. This area has not changed in Message Broker Version 6.1.

The second area is the Runtime Security. This concerns the Broker runtime itself, and which clients are allowed to send a message to the runtime for processing. In many cases, the control of this security will be delegated to the underlying transport, for example, WebSphere MQ. In some cases, for example with DataPower, this control can be performed outside of the Broker environment. This area has some significant changes in Version 6.1.



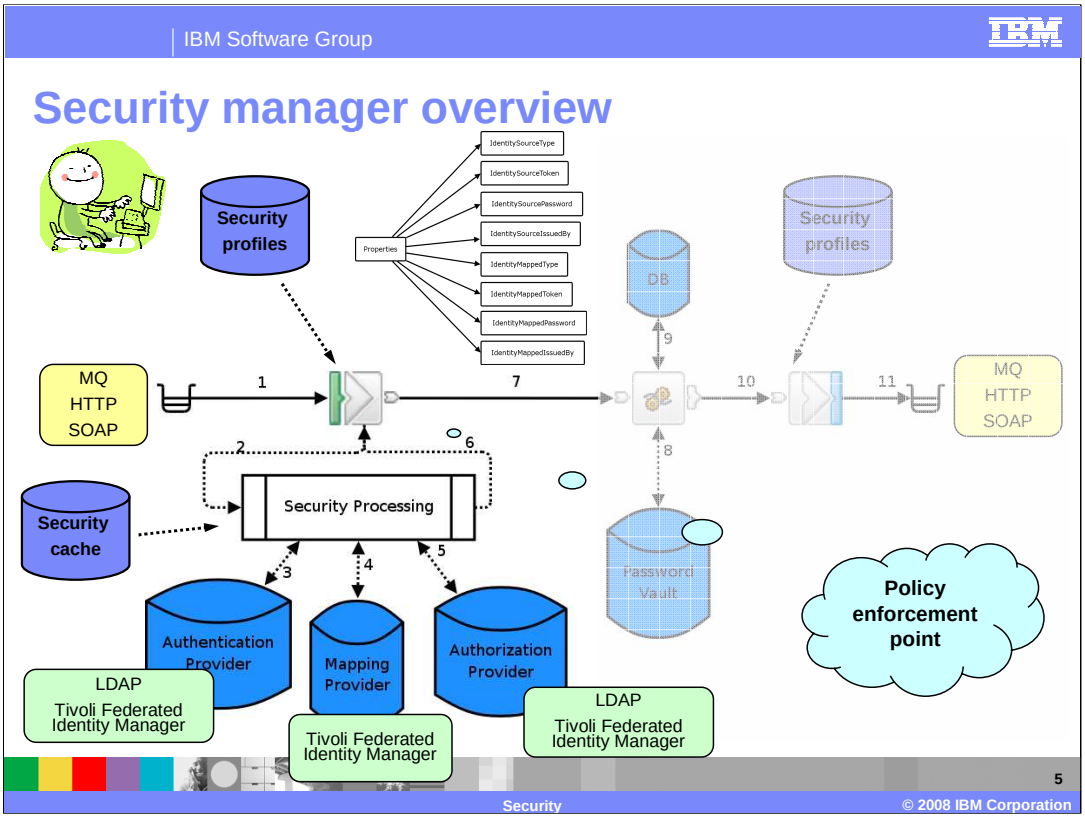
The significant change to security in Version 6.1 is for Runtime security. A new *security manager* is provided that enables access to message flows to be controlled on a *per message basis* using the *identity* of the message.

Instead of delegating this authority to the transport, or to an external security manager such as a DataPower appliance, the broker itself can perform several tasks. It can extract the identity from an inbound message. Using an external security provider, it can authenticate an inbound message, map the identity to an alternate identity, and check that the alternate identity or the original identity is authorized to access the message flow. It can also Propagate the alternate identity or the original identity with an outbound message.

The actions to take for a given message flow are controlled using new *security profiles*. These are created by the broker administrator and are accessed by the security manager at runtime.

In Version 6.1, two external security providers are supported, so that the broker can participate in a centralized security framework.

These are *LDAP* for authentication and authorization, and Tivoli Federated Identity Manager for authentication, mapping and authorization.



The first step in configuring the security manager is to create security profiles. This is done using either the Broker Administration perspective of the toolkit or the *mqsicreateconfigurableservice* command. It is also in the IS02 SupportPac.

When a message arrives at an input node - step 1 in this picture - a security profile is used to indicate whether runtime security is configured. The input nodes that support runtime security in 6.1 are *MQInput, HTTPInput* and *SOAPInput*. At step 2 to 5, the broker's security manager is called to read the security profile. This specifies which combination of authentication, authorization and mapping is to be performed with the identity of the message, and by what external security provider.

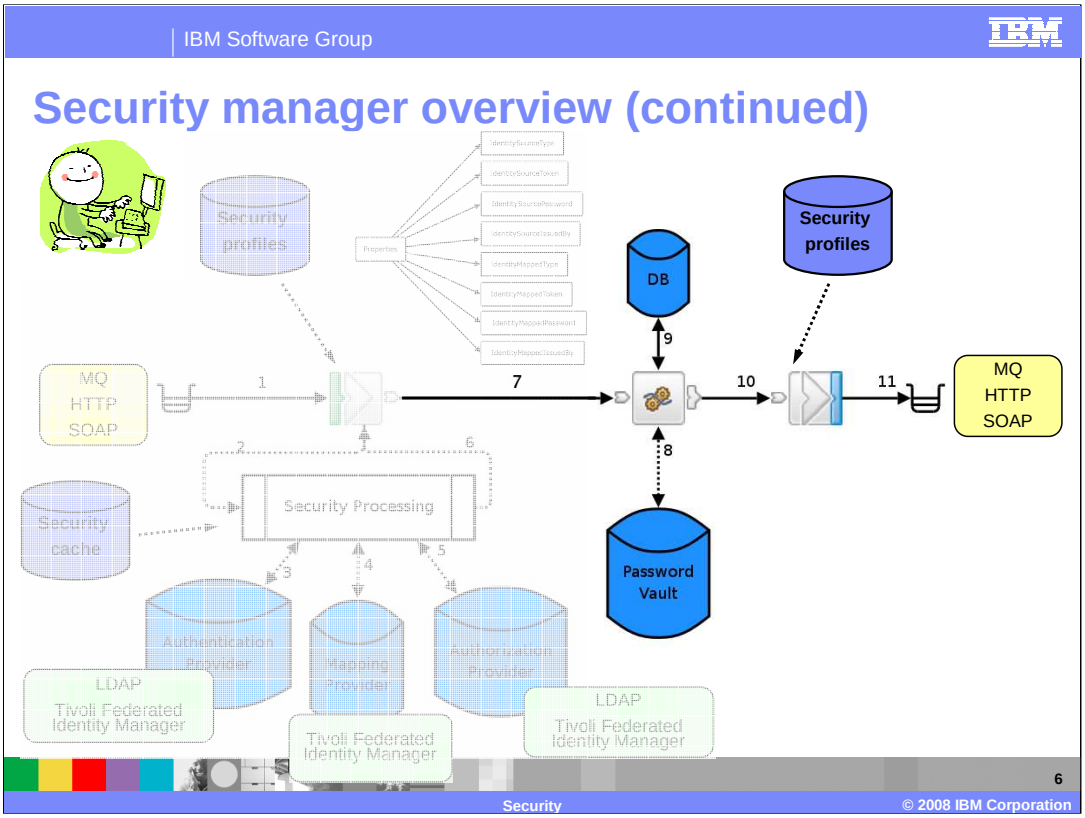
The security manager extracts the identity information from the input message and sets it in a group of new elements in the Properties folder. This 'source' identity information could be in a message header or in the message body itself, or a mixture of the two.

If authentication was specified in the security profile, the security manager calls the provider to authenticate the identity. A failure results in a SecurityException being thrown. Supported providers in 6.1 are *LDAP* and *Tivoli Federated Identity Manager*.

If identity mapping was specified in the security profile, the security manager calls the provider to map the identity to an alternate identity. A failure results in a SecurityException being thrown. Otherwise the 'mapped' identity information is set in a group of new elements in the Properties folder. Supported provider in 6.1 is *Tivoli Federated Identity Manager*.

If authorization was specified in the security profile, the security manager calls the provider to authorize that the identity has access to this message flow. A failure results in a SecurityException being thrown. Supported providers in 6.1 are *LDAP* and *Tivoli Federated Identity Manager*.

When all security processing is complete, at step 6, control returns to the input node. The input node is a *Policy Enforcement Point*. In Version 6.1, only input nodes can be policy enforcement points.



The message, including the properties folder and its source and mapped identity information, is propagated down the flow, shown as step 7.

At subsequent nodes in the flow, an identity may need to be used to access a resource such as a database. The identity used to access such a resource continues to be a proxy identity, either the broker's identity or an identity configured using the *mqsi-set-db-parms* command.

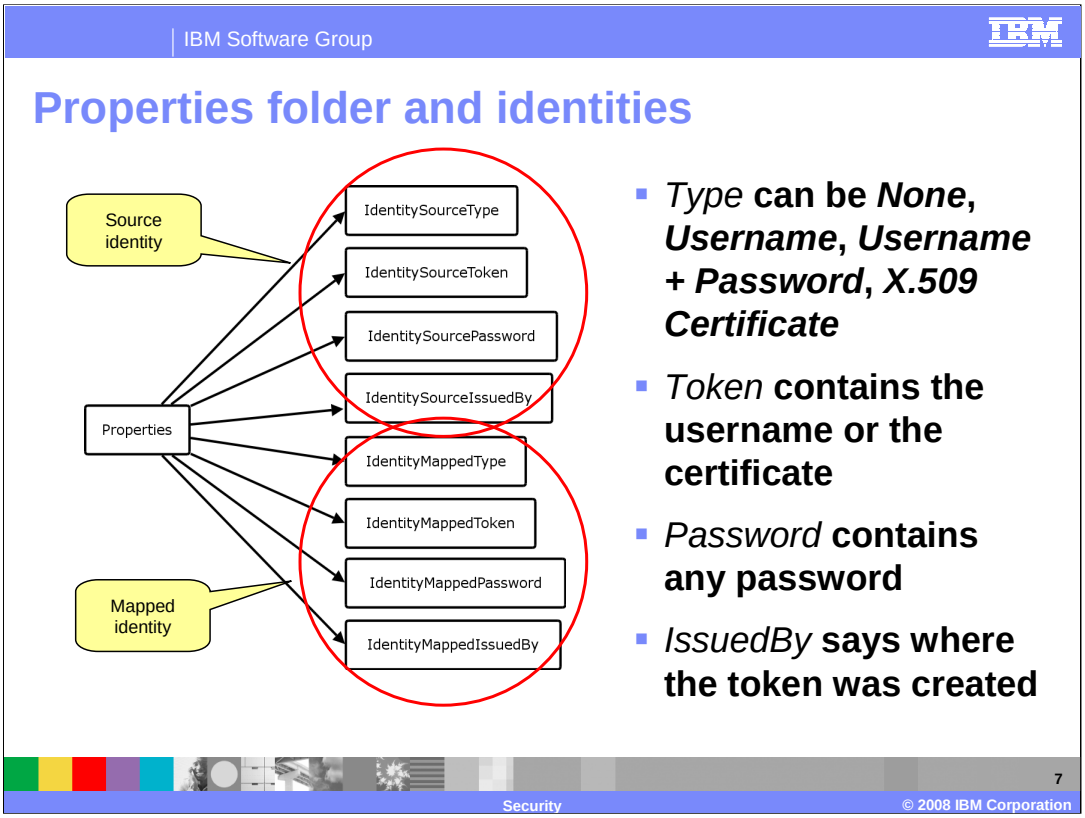
The resource is accessed using the appropriate proxy identity, shown at steps 8 and 9.

When the message reaches an output node, in step 10, a security profile is used to indicate whether an identity is to be propagated when the message is sent. The output nodes that support identity propagation in version 6.1 are *MQOutput, HTTPRequest*, *SOAPRequest* and *SOAPAsyncRequest*.

If the security profile indicates that propagation is required, the mapped identity is used, or if that is not set, the source identity is used. If no identity is set a RecoverableException is thrown.

In step 11, any propagated identity is included in the appropriate message header when it is sent.

To improve performance, authentication, authorization and mapping information from the providers is cached for re-use. The operation of the cache is automatic, but it can be tuned if needed using the *mqsichangeproperties* and new *mqsireloadsecurity* commands.



An identity is a piece of information which can uniquely identify an individual or object. Within the Broker, identity is held in the Properties folder of the broker message tree.

There are eight new additions to the Properties folder, between them defining two identities; 'source' and 'mapped'. For each of these identities, Type, Token, Password and IssuedBy fields are held.

The *Type* field defines the format of the Token, either *Username*, *Username + Password* or *X.509 Certificate*.

The *Token* field holds the actual token, either the username or certificate.

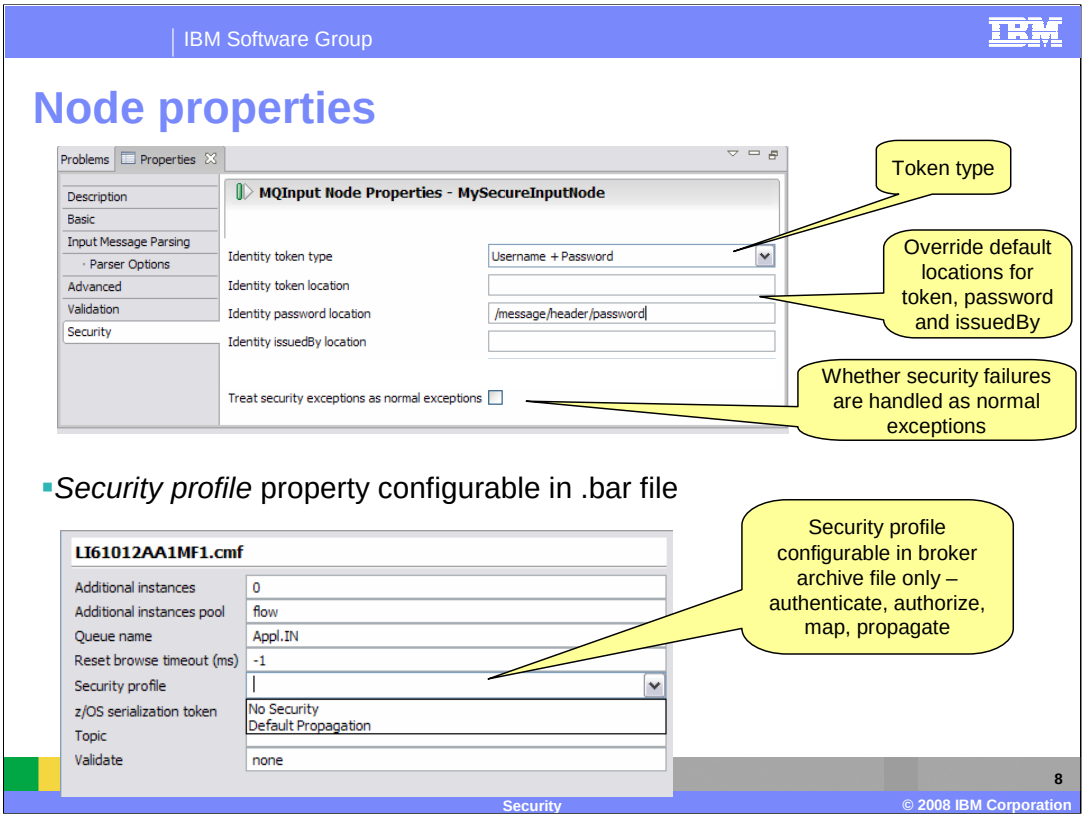
In the case of a *Username + Password* token, the *Password* field will additionally contain the associated password.

The *IssuedBy* field defines where the Token was created. For example, for a X.509 certificate this could be "IBM" (the Common Name of the Certifying Authority). For a *Username* and *Username + Password,* this is transport specific.

The source identity is always set by the input node.

If multiple identities are available (for example through aggregation), the first identity is used.

The values in the properties are writeable, for example from ESQL, though it is not recommended to write to the IdentitySource values.



Security properties are carried on two kinds of nodes, input nodes and output/request nodes.

For input nodes, whether or not runtime security is configured for the node is determined by the *Security profile* property. If no security profile is specified then security is not configured and the flow will behave the same as in version 6.0. Otherwise it is the security profile that says which combination of authentication, authorization and mapping is to be performed with the identity in the message.

The *Identity token type* property specifies how the identity appears in the message. It can be one of *Username, Username + Password*, or *X.509 Certificate* then security is configured.

The default location in the message of the token, password and issuer is transport dependent (see later slides). However the location can be overridden using the *Identity token location, Identity password location* and *Identity issuedBy location* properties.

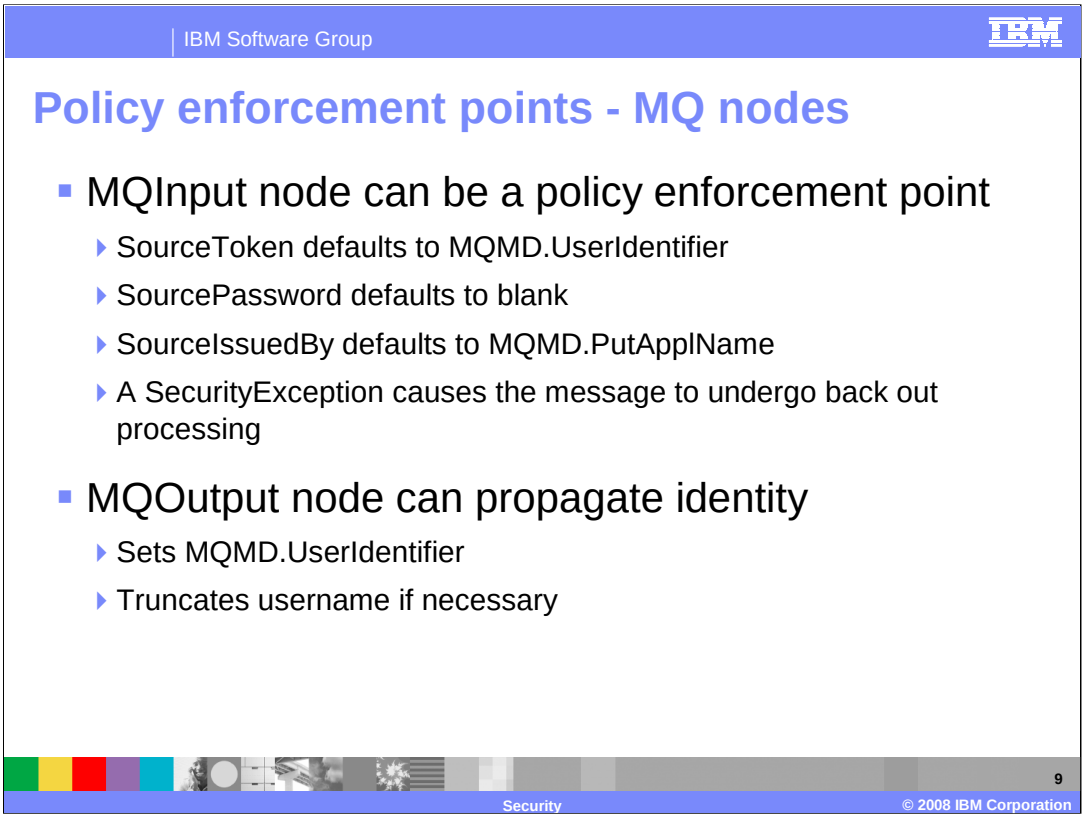
If a SecurityException is thrown as a result of an authentication, authorization, or mapping failures, the default behavior is that it can not be caught by exception handlers, such as wired Catch terminals. Instead the exception is always returned to the input node, where the behavior is transport dependent. This can be overridden by the *Treat Security exceptions as normal exceptions* property, which if checked allows security failures to be handled using the usual exception handlers.

Note that the Identity fields in the Properties folder are only set if a security profile is present for the input node.

For output and request nodes, whether the identity is propagated with the outbound message is determined by the security profile given by the *Security profile* property. A pre-configured profile for use by output and request nodes is shipped with the broker which specifies propagation.

Note that the *Security profile* property is ‘hidden’ but ‘configurable’, meaning that it can only be set in the broker archive (bar) file at deploy time by an administrator. It is *not* visible on the node itself. There is also a *Security profile* property on the message flow itself, which acts as a default for all nodes in the message flow that do not specify a security profile explicitly.

When the flow-level property is set a node can still be configured to not have a profile by choosing "No Security" on it. In this case, it will not use the flow-default value.



The provision of the security manager means that an input node can now act as a Policy Enforcement point.

The default locations from where to obtain the token, password and issuedBy information are transport dependent and are shown on the slides. To override the default locations, use the node location properties to specify an ESQL path or XPath to the actual location in the message header or body.

The behavior when handling a Security Exception is transport dependent. This slide shows the behavior when using an MQ Input node.

If no source token is available, then the default values will be based on normal MQ security values. Note that the password will be blank, since the MQMD has no password field. In this case, if a security exception occurs, then the message flow will undergo normal back out processing.

An MQ Output node can propagate this security context, and sets the User Identifier in the MQMD. Since this field has a maximum length, it is possible that this value may be truncated to fit this field.

Policy enforcement points - HTTP nodes

- HTTPInput node can be a policy enforcement point
  - Provides “HTTP Basic Auth” capability
  - SourceToken defaults to HTTP Authorization header user
  - SourcePassword defaults to HTTP Authorization header password
  - SourceIssuedBy defaults to HTTP User-Agent header or “HTTP”
  - A SecurityException causes an HTTP 40x status code to be returned to the client
- HTTPRequest node can propagate identity
  - Sets HTTP Authorization header

When using the HTTP input node, security is equivalent to “HTTP Basic Auth”. This means that security is based around user ID and password authentication.

Hence, the default values for the source token and password fields are the HTTP header user and password fields. The “IssuedBy” field is set to the User-Agent header or HTTP.

If a security exception occurs, the message flow returns a standard HTTP 40x status code to the client.

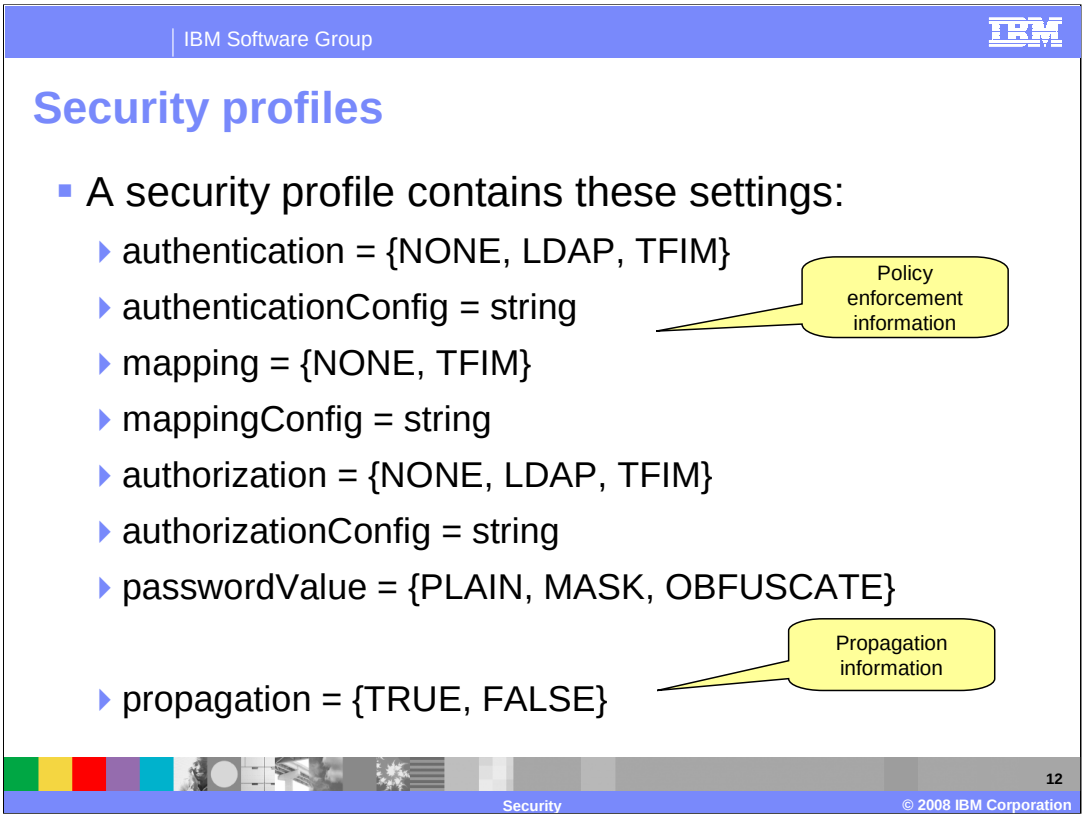
For outbound security, the HTTP Request node can propagate the security context. It does this by setting the same values in the HTTP Authorization header.

Policy enforcement points - SOAP nodes

- SOAP Input node can be a policy enforcement point
  - Behavior depends on whether WS-Security in use
  - If no WS-Security then behaves like HTTPInput node
  - If WS-Security then uses SOAP Header <wsse:security> element
  - A SecurityException causes a SOAP fault to be returned
- SOAP Request and SOAP AsyncRequest nodes can propagate identity
  - Behavior depends on whether WS-Security in use
  - If no WS-Security then behaves like HTTPRequest node
  - If WS-Security then sets <wsse:security> element

The SOAP nodes behave in two different ways depending on whether the WS-Security protocol is being used by the message. WS-Security is covered in a separate session.

If WS-Security is not being used, then security behavior is exactly the same as in the HTTP nodes, described on the previous slide.



In Message Broker version 6.1, a security profile consists of two kinds of information:

First, policy enforcement information. This covers whether to authenticate, authorize or map an identity along with the provider to use an associated configuration string.

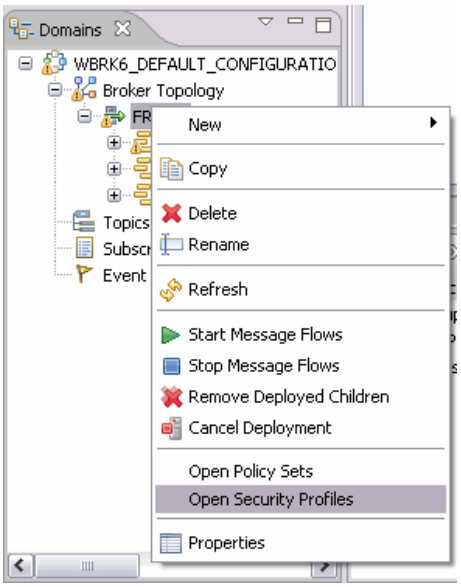
Second, propagation information. This covers whether to propagate the identity with an output message.

Security profiles can be created, deleted, viewed and edited using a security profile editor, part of the broker toolkit administration perspective. This assists with the building of the sometimes complex configuration strings needed by the providers. Clicking on the Finish button of the editor sends the updates direct to the broker. Security profiles are *not* deployed in the .bar file.

Alternatively security profiles may be created, deleted, and viewed using the broker *mqsicreateconfigurableservice, mqsideleteconfigurableservice, mqsichangeproperties* and *mqsireportproperties* commands, or their Configuration Manager API equivalent.

Creating security profiles using the editor

- The existing broker administration 'Domains' view has a new option when right-clicking on a broker called '*Open Security Profiles*'
- Selecting '*Open Security Profiles*' will open the security profiles editor
- Also in IS02 SupportPac™

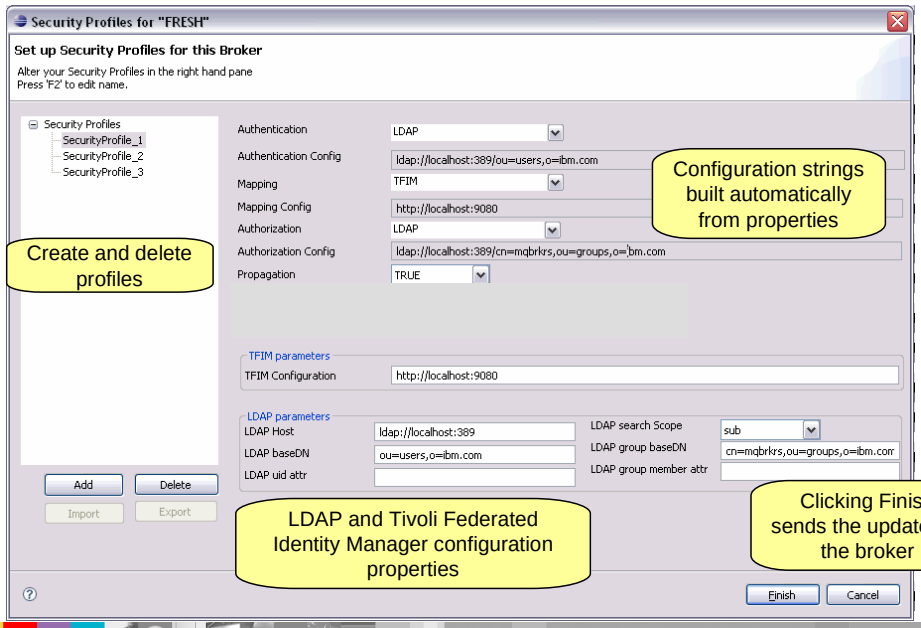


This slide shows how to open the Security Profiles editor from the Administration perspective in the Broker Toolkit.

Right-click the Broker that you want to update, and select the Profile editor as shown.

It is also possible to do this using the IS02 SupportPac, which can be used to manage the Broker runtime.

Creating security profiles using the editor



This slide shows the Security Profiles editor. Using this editor, you can create and delete particular profiles. In each of these profiles, you can specify the LDAP or Tivoli parameters that are required. Once complete, clicking Finish sends these updates directly to the Broker.

You are recommended to review the Info Center to fully understand these options, and how to specify the various parameters.

Creating security profiles using commands

- To create a new security profile
  - mqsicreateconfigurableservice <broker> -c SecurityProfiles –o <profile-name> -n <property-name-list> -v <property-value-list>
- To delete a security profile
  - mqsideleteconfigurableservice <broker> -c SecurityProfiles –o <profile-name>
- To change the values in a security profile
  - mqsichangeproperties <broker> -c SecurityProfiles –o <profile-name> -n <property-name-list> -v <property-value-list>
- To report the values in a security profile
  - mqsireportproperties <broker> -c SecurityProfiles –o <profile-name> -r
  - mqsireportproperties <broker> -c SecurityProfiles –o allReportableEntityNames -r

The security profiles can be created and maintained using configurable services. The commands shown on this slide show the commands and options that would be required to achieve this. As with the security profile editor, you should review the product documentation before starting this process.

LDAP support

- LDAP Version 3-compatible server required
  - IBM Tivoli Directory Server
  - Microsoft® Active Directory®
  - OpenLDAP
- If anonymous login not permitted
  - mqsisetdbparms –n ldap::LDAP –u <username> –p <password>
  - mqsisetdbparms –n ldap::<servername> –u <username> –p <password>
- Supported token types
  - *Username*
  - *Username + password*
- Use of security profile editor recommended

If you are going to manage security using an LDAP repository, then a server compatible with LDAP Version 3 is required. This can be provided with IBM Tivoli Directory Server, Microsoft Active Directory or OpenLDAP.

If your LDAP server does not permit anonymous login, you need to use the "*mqsi-set-db-parms"* command to set up the fully qualified username and password to be used.

LDAP support is for token types of *username* and *username + password*.

Building LDAP configuration strings can be complex. The next slide shows some examples of how to do this with the command interface.

LDAP support - examples

- Support available for three combinations
  - Authentication only
  - Authorization only
  - Authentication and authorization
- Authentication and authorization example

```
mqsicreateconfigurableservice WBRK_BROKER -c SecurityProfiles -o LDAP
-n authentication,authenticationConfig,authorization,authorizationConfig
-v "LDAP,\"ldap://ldap.acme.com:389/ou=sales,o=acme.com\",LDAP,
\"ldap://ldap.acme.com:389/cn=All Sales,ou=acmegroups,o=acme.com\""
```

- Note that any commas within baseDN and uid attribute need to be replaced with “%2c”

Building LDAP configuration strings is quite complicated. The security configuration can be used for authentication, authorization, or a combination of the two.

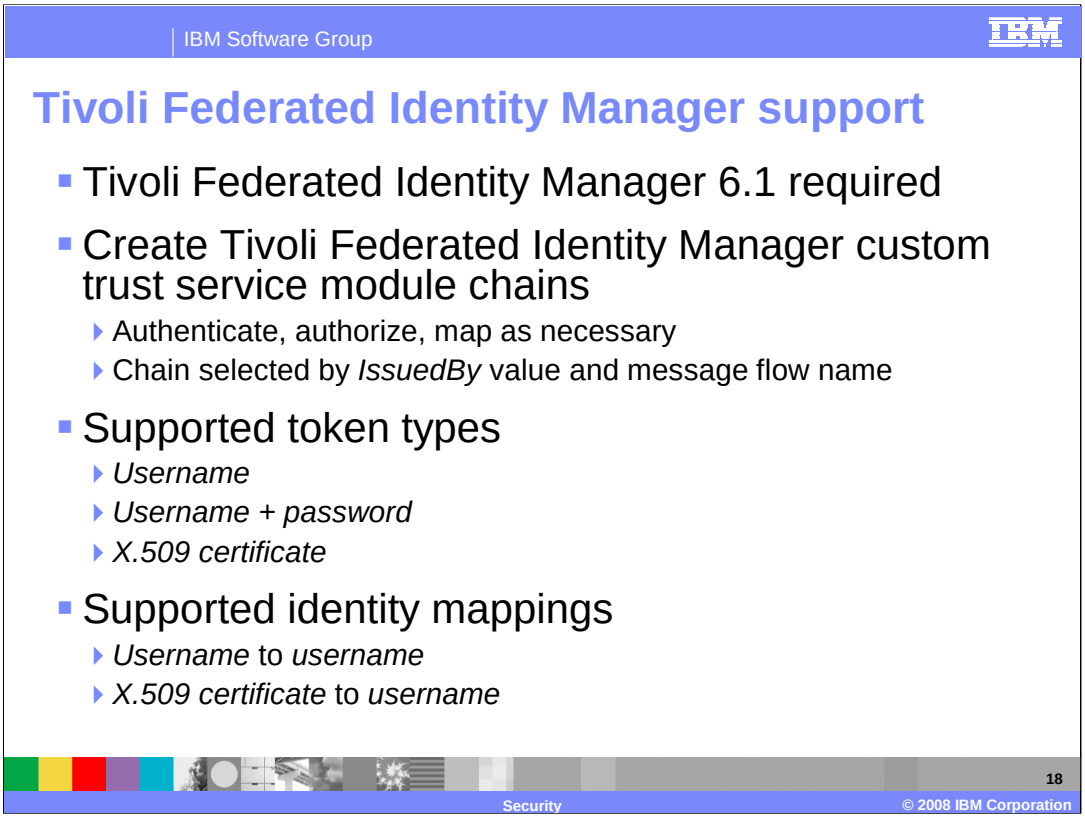
The example shown on this slide shows the command that would be required to configure a combined authentication and authorization scenario.

The key observations in this example are:

The values on the “-n” parameter are separated by a Comma; this example has values specifying with authentication and authorization.

There are two values:

In this example above, you must enclose the LDAP URL, which contains commas, with escaped quotation marks, so that the URL commas are not confused with the comma separator of the value parameter of *“mqsi-create-configurable-service”*. These are shown in red on this slide.



If you are going to use Tivoli Federated Identity Manager, then you will need to plan for Tivoli Federated Identity Manager Version 6.1.

You should customize Tivoli Federated Identity Manager to perform the required action against the identity. This is performed using Trust Service module chains to authenticate or authorize or map the identity.

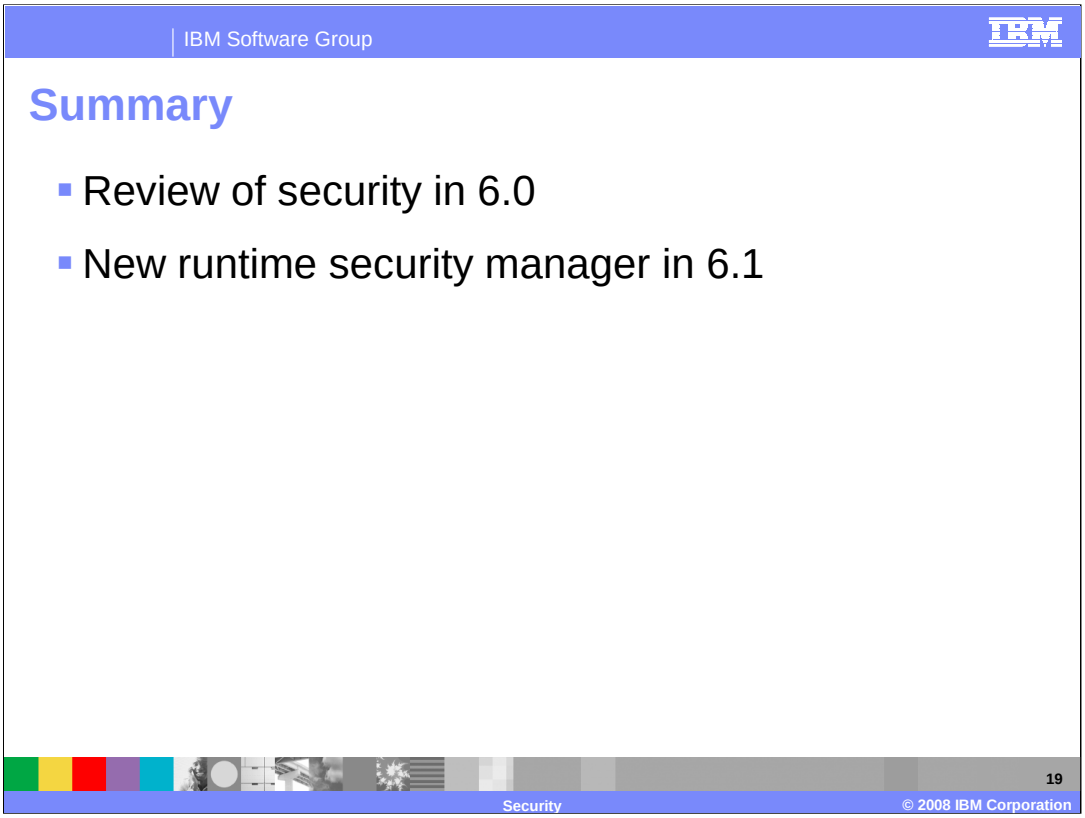
The chain to use is determined by a combination of the source identity *issuedBy* value and the name of the message flow, expressed as:

<broker-name>.<execution-group-name>.<message-flow-name>.

Tivoli Federated Identity Manager support is for token types of *Username, Username + Password* and *X.509 Certificate*.

As far as identity mapping is concerned, it is possible to map a username to another username, and an X.509 certificate to a username. But it is not possible to map a username to an X.509 certificate.

When mapping from an X.509 certificate, Tivoli Federated Identity Manager can validate the certificate, but can not be used to verify the identity of the original sender. This would have to be done elsewhere, for example, using WS-Security support for digital signatures using a SOAPInput node.



To summarize, this presentation covered a brief review of the security mechanisms available in Message Broker Version 6.0. The presentation then discussed the new security features in Version 6.1, with some examples of how to use the command line interface to configure this.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WMB61_IEA_Security.ppt

This module is also available in PDF format at: ../WMB61_IEA_Security.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

DataPower IBM Tivoli WebSphere

Active Directory, Microsoft, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.