Related Patterns

Three types of patterns are related to this pattern: child patterns, alternate patterns, and additional patterns.

The following child patterns are related to the Data Authentication pattern:

- **Implementing Direct Authentication with UsernameToken in WSE 3.0**. This implementation pattern focuses on using direct authentication at the message layer.
- **Implementing Transport Layer Security Using HTTP Basic over HTTPS**. This reference provides information about implementing direct authentication using Internet Information Services (IIS) with X.509 certificates at the transport layer.

The following alternate pattern is related to the Direct Authentication pattern:

- **Brokered Authentication**. This pattern is an alternative to direct authentication that describes how to prove a client's identity to an authentication broker for issuance of a security token, and then use the issued security token to authenticate with a service.

The following pattern may use the Direct Authentication pattern:

- **Brokered Authentication**. This pattern may use variations of direct authentication to prove a client's identity to an authentication broker for issuance of a security token.

Brokered Authentication

Context

A client needs to access a Web service. The Web service requires the application to present credentials for authentication so that additional controls such as authorization and auditing can be implemented.

Problem

How does the Web service verify the credentials that are presented by the client?

Forces

Any of the following conditions justifies using the solution described in this pattern:

- **The client accesses additional services, which results in the need for a single sign on (SSO) solution**. Without a single sign on solution, the client may be forced to authenticate prior to every Web service call or cache the user's credentials within the application. If the user's credentials include a password, caching the password is not recommended because it may pose a security risk.
- **The client and the Web service do not trust each other directly**. The client and the Web service may not trust one another to manage or exchange shared secrets securely. Establishing trust directly between a client and Web service often requires out of band interactions that can hinder clients and services from interacting dynamically.
- **The Web service and the identity store do not trust each other directly**. The Web service may be unable to communicate with the identity store directly, because of access control restrictions, network restrictions, or organizational policy.

The following condition is an additional reason to use the solution.

- **The client and Web service share a standard access control infrastructure**. You can simplify the development of new Web services by standardizing and centralizing the issuance and verification of credentials. You can also centralize the management of data associated with credentials; this reduces the costs associated with identity management.

Solution

Use brokered authentication where the Web service validates the credentials presented by the client, without the need for a direct relationship between the two parties. An authentication broker that both parties trust independently issues a security token to the client. The client can then present credentials, including the security token, to the Web service.

Participants

Brokered authentication involves the following participants:

- **Client**. The client accesses the Web service. The client provides the credentials for authentication during the request to the Web service.
- **Service**. The service is the Web service that requires authentication of a client prior to authorizing the client.
- **Authentication broker**. The authentication broker authenticates clients and maintains authoritative control over security tokens. It also vouches for the client by issuing it a security token.
- **Identity store**. The entity that stores a client's credentials for a particular identity domain.

Process

Figure 1.6 depicts the interactions that are performed during brokered authentication.

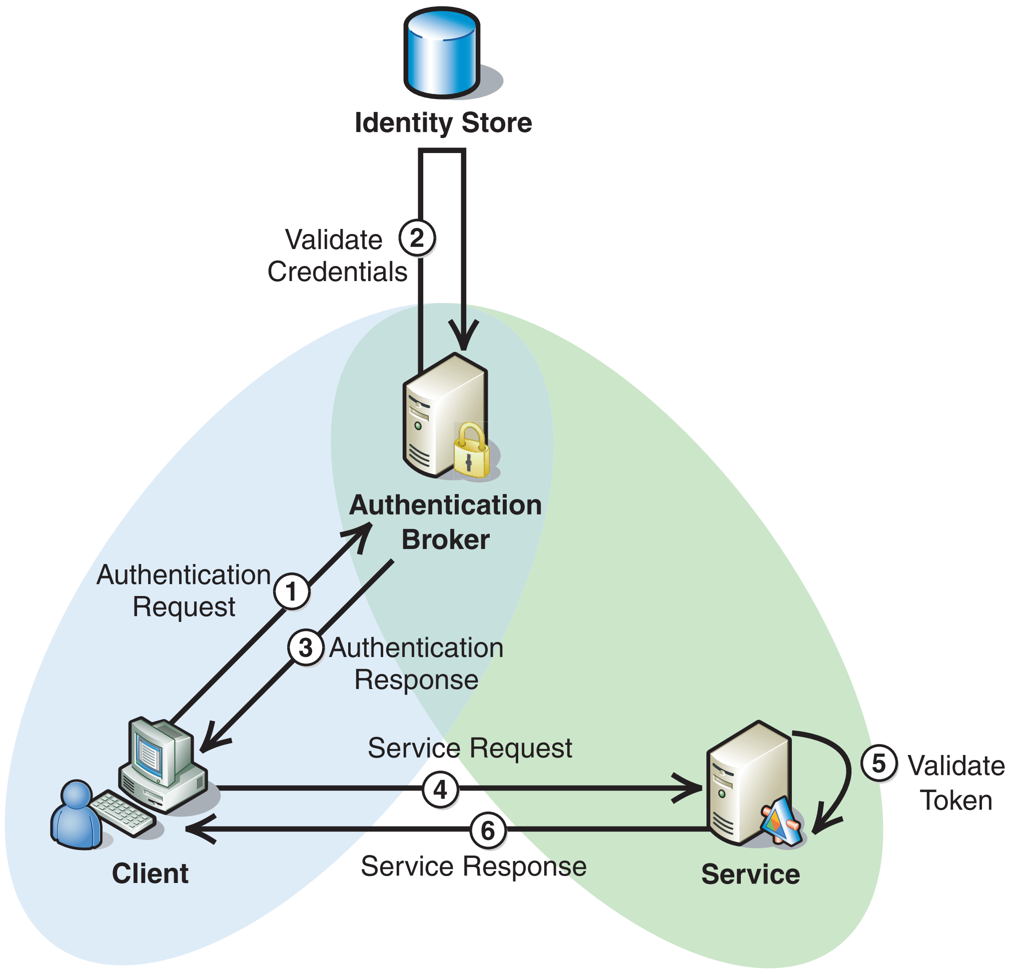

Figure 1.6
Brokered authentication process

As illustrated in Figure 1.6, the following steps describe the brokered authentication process:

1. The client submits an authentication request to the authentication broker.
2. The authentication broker contacts the identity store to validate the client's credentials.
3. The authentication broker responds to the client, and if authentication is successful, it issues a security token. The client can use the security token to authenticate with the service. The security token can be used by the client for a period of time that is defined by the authentication broker. The client can then use the issued security token to authenticate requests to the service throughout the lifetime of the token.
4. A request message is sent to the service; it contains the security token that is issued by the authentication broker.
5. The service authenticates the request by validating the security token that was sent with the message.
6. The service returns the response to the client.

There are different types of authentication brokers. Each type uses different mechanisms to broker authentication between a client and a service. Common examples of an authentication broker include the following:

- X.509 PKI
- Kerberos protocol
- Web Service Security Token Service (STS)

Resulting Context

This section describes some of the more significant benefits, liabilities, and security considerations of using this pattern.

Note: The information in this section is not intended to be comprehensive. However, it does discuss many of the issues that are most commonly encountered for this pattern.

Benefits

The benefits of using the Brokered Authentication pattern include the following:

- The authentication broker manages trust centrally. This eliminates the need for each client and service to independently manage their own trust relationships, as shown in Figure 1.7.

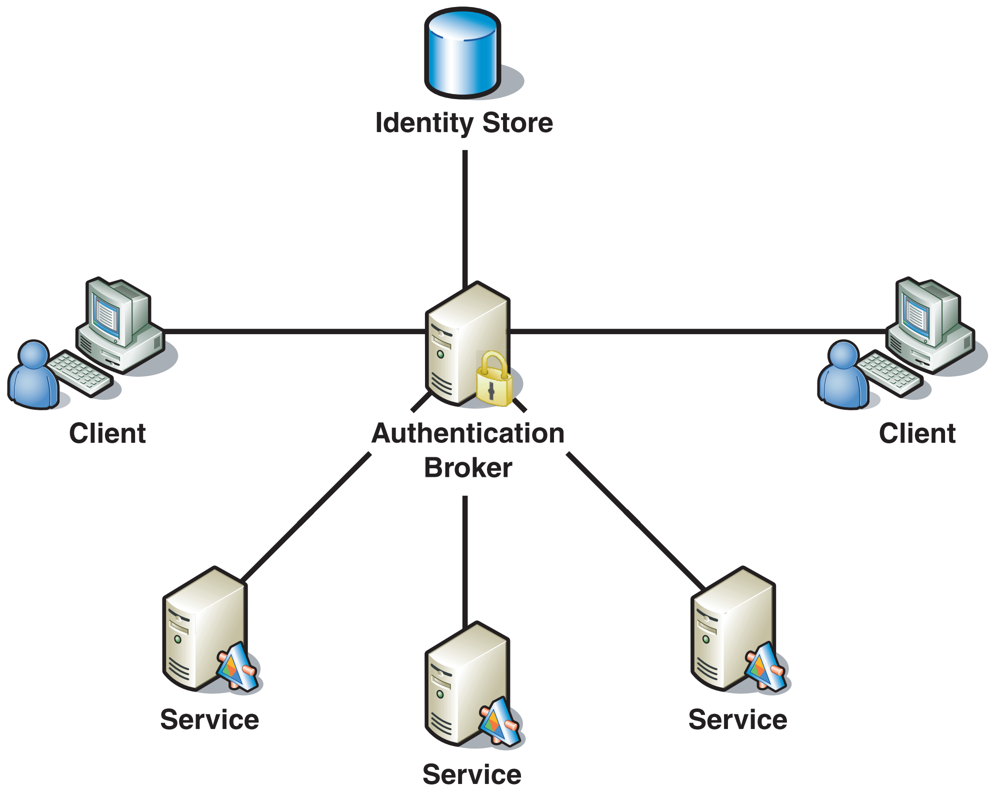

Figure 1.7
An authentication broker centrally managing trust

- Solutions built around brokered authentication with a centralized identity provider are often easier to maintain than direct authentication solutions. When new users who require access to any of the clients or Web services are added to the identity store, their credentials are maintained in one central point.
- Two parties participating in brokered authentication do not require prior knowledge of one another to communicate. If a client is modified to call a Web service it has never used before, the Web service requires no changes to its configuration or data to authenticate credentials presented by the client.
- Trust relationships can be established between different authentication brokers. This means that an authentication broker can issue security tokens that are used across organizational boundaries and autonomous security domains.

Liabilities

The liabilities associated with the Brokered Authentication pattern include the following:

- The centralized trust model that is used by Brokered authentication can sometimes create a single point of failure. Some types of authentication brokers, such as the Kerberos Key Distribution Center (KDC), must be online and available to issue a security token to a client. If the authentication broker somehow becomes unavailable, none of the parties that rely on the authentication broker to issue security tokens can communicate with each other. This problem of a single point of failure can be mitigated by implementing redundant or back-up authentication brokers, although this increases the complexity of the solution.
- Any compromise of an authentication broker results in the integrity of the trust that is provided by the broker also being compromised. If an attacker does successfully compromise the authentication broker, it can use the authentication broker to issue security tokens, and conduct malicious activity against parties that trust the authentication broker.

Security Considerations

Security considerations associated with the Brokered Authentication pattern include the following:

- Claims held in security tokens often contain sensitive data, and must be protected in transit, either by using message layer security, or transport level security.
- Security tokens must be signed by the issuing authentication broker. If they are not, their integrity cannot be verified. This could result in attackers trying to issue false tokens.
- A Time of Change/Time of Use vulnerability may exist if the client's account status, identity attributes, or authorization attributes are modified by an account administrator. If these changes are not reflected in the security token, it creates a vulnerability that may lead to invalid clients interacting with the service with elevated privileges.

Related Patterns

Three types of patterns are related this pattern: child patterns, alternate patterns, and additional patterns.

The following child patterns are related to the Brokered Authentication pattern:

- **Brokered Authentication: X.509 PKI**. This pattern describes a specialized authentication broker based on the X.509 PKI standard.
- **Brokered Authentication: Kerberos**. This pattern describes a specialized authentication broker based on the Kerberos authentication protocol.
- **Brokered Authentication: Security Token Service (STS)**. This pattern describes a specialized authentication broker in the form of a Security Token Service.

The following alternate pattern is related to the Brokered Authentication pattern:

- **Direct Authentication**. This pattern is an alternative to brokered authentication where authentication is based on an identifier and a shared secret, such as username and password.

One additional pattern is related to the Brokered Authentication pattern:

- **Broker**. This pattern is in *Enterprise Solution Patterns Using Microsoft .NET* on the MSDN Web site. This pattern shows how to hide the implementation details of remote service invocation.

Brokered Authentication: Kerberos

Context

Web services must authenticate clients so that additional controls, such as authorization and auditing, can be implemented. The organization has decided to use an *authentication broker* to provide a common access control infrastructure for a group of applications. The authentication broker negotiates trust between client applications and Web services, which removes the need for a direct relationship. The authentication broker should issue signed security tokens that can be used for authentication.

Problem

How does the Web service verify the credentials presented by the client application?

Forces

Any of the following conditions justifies using the solution described in this pattern:

- **Users access multiple clients that call Web services, resulting in the need for single sign on (SSO) capabilities**. To ensure a good user experience, users should only have to enter a username and password when they logon to their workstations. They should not need to re-enter them multiple times to access multiple clients.
- **Centralized authentication of clients is required**. Management of user and computer credentials must be centralized to minimize security risks associated with persisting credentials and to reduce maintenance overhead.
- **Clients that require authentication are implemented on a variety of platforms within the organization, and the organization has identified a need for interoperability between those platforms**. The easiest way to attain interoperability between different platforms is to use a standards-based mechanism for authentication.