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.

- **Clients may exist in an untrusted network environment**. You may not be able to guarantee the security of the computers on the network or of the network itself. User credentials must be protected from malicious attackers that may have gained access to the network inappropriately.

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

- **Applications require some of the extended capabilities associated with a particular implementation of the Kerberos protocol**. For example, the Windows Server 2003 implementation of the Kerberos protocol provides capabilities, such as protocol transition, constrained delegation, and integration with Active Directory.

Solution

Use the Kerberos protocol to broker authentication between clients and Web services.

The client requests a ticket from an authentication broker, which returns a service ticket and session key used to create a Kerberos security token. The security token includes the service ticket and a data structure called an *authenticator*, which is encrypted by using the session key retrieved from the broker. Then the Kerberos security token is sent with a request message to the Web service.

When the Web service receives a Kerberos security token, it extracts the service ticket and uses a long-term service key to decrypt the service ticket. The Web service uses the session key from the service ticket to decrypt the authenticator and authenticate the client.

Note: The term *Kerberos security token* is used to represent a data structure that contains a service ticket and authenticator. For more information on Kerberos tickets and authenticators, see Kerberos Technical Supplement for Windows in Chapter 7, "Technical Supplements."

Participants

Brokered authentication using the Kerberos protocol 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.
- **Key Distribution Center (KDC)**. The KDC is the authentication broker that is responsible for authenticating clients and issuing service tickets. On the Windows platform, the KDC is implemented in Active Directory.

The Kerberos protocol is an authentication protocol that requires the following additional components.

- **Account database**. This is an identity store that the Kerberos KDC uses to check client credentials presented for authentication. Master keys for the client and service are also stored in this database. If the Kerberos protocol is implemented on a Windows Server 2003 domain controller or on a Windows 2000 domain controller, then Active Directory provides this function.

> **Note:** The term *master key* refers to a long-term key, which is described in the Kerberos Technical Supplement for Windows in Chapter 7, "Technical Supplements."

- **Kerberos policy**. This is a security policy that defines behavior for a Kerberos realm, which is also an Active Directory domain. Policy settings include user logon restrictions, service ticket lifetime, user ticket lifetime, and clock synchronization.

> **Note:** There is some inconsistency in how service tickets are described in Kerberos documents. The names *service ticket* and *session ticket* are used interchangeably. When you encounter the phrase *session ticket*, remember that this is a *service ticket*.

The relationship between participants is shown in Figure 1.8.

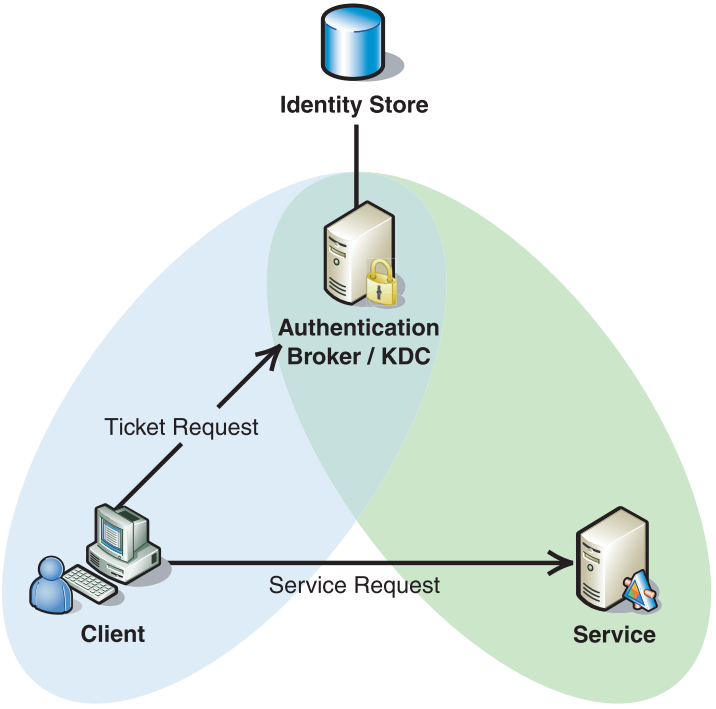

Figure 1.8
Relationship between participants

Process

Brokered authentication with the Kerberos protocol consists of the following high-level tasks:

1. **The client authenticates with the broker (KDC)**. The client is authenticated with the broker, and given access to a ticket-granting ticket (TGT) that can be used to request access to a service.
2. **The client authenticates with the service**. The client uses the TGT to request access to a particular service, and then it receives a service ticket. The service uses the service ticket to validate credentials.

Note: The Windows implementation of the Kerberos protocol uses many components and interfaces that are beyond the scope of this pattern. The process described in this pattern focuses on the interaction of primary components that the Kerberos protocol uses to authenticate with a Web service, and not on the low-level implementation of the Kerberos protocol on the Windows platform.

Client Authenticates with Broker (KDC)

Clients can be authenticated through a wide variety of techniques, including:

- Workstation user login using the secure attention sequence (CTRL+ALT+DELETE).
- Windows integrated authentication used to access a Web application.
- IIS process identity authentication used when the process starts.
- Protocol transition used to transition clients authenticated using a non-Windows protocol into a Kerberos security context.

The actual process of authenticating a client is beyond the scope of this pattern. However, it's important to understand that the client must first be authenticated with the broker and have access to a TGT before it can request access to a service.

For detailed information about the process of authenticating clients and issuing ticket-granting tickets on the Windows platform, see Kerberos Technical Supplement for Windows in Chapter 7, "Technical Supplements."

For information about protocol transition, see Protocol Transition with Constrained Delegation Technical Supplement in Chapter 4, "Resource Access Patterns."

Client Authenticates with Service

The process of authenticating with a service is shown in Figure 1.9.

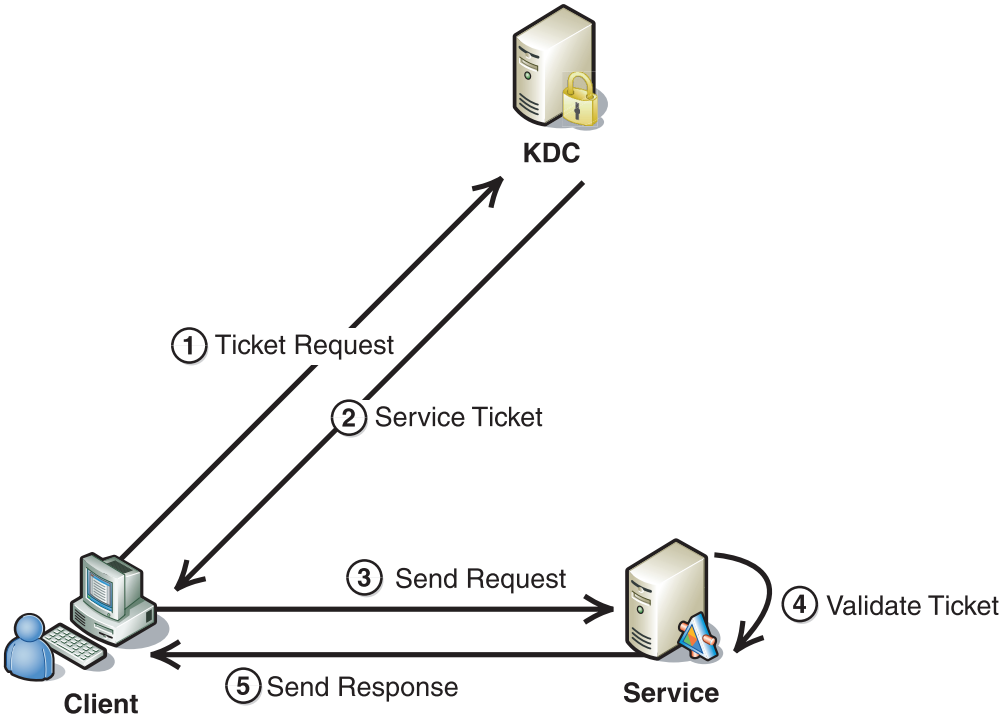

Figure 1.9
Service Authentication

The following steps describe the process of service authentication depicted in Figure 1.9:

1. The client sends a TGT in a message to the KDC to request a service ticket for communication with a specific Web service.
2. The KDC creates a new session key and service ticket that will be used for communication with the requested service. The service ticket contains the client's authorization data and the new session key. The KDC encrypts the service ticket with the Web service's master key. The service ticket and encrypted session key are returned to the client.

   Both the new session key and service ticket represent credentials used to create a security token that allows access to the Web service. The client decrypts the session key and then uses the key to encrypt an authenticator, which contains a timestamp and other information. The authenticator and session ticket are included in the new Kerberos security token. The session key is not included in the token; however, it is included in the service ticket, which is what the service uses to validate the token.

3. A request message, which contains the Kerberos security token created in the previous step, is sent to the service.
4. The service uses its master key to decrypt the service ticket found in the security token and to retrieve the session key. The session key is used to decrypt the authenticator and validate the security token. When it is validated, the service accepts the security token and uses it to initialize a security context based on the client information contained in the service ticket.
5. (optional)The service returns a response to the client. To provide mutual authentication, the response should contain unique information that is encrypted with the session key to prove to the client that the service knows the session key.

The Kerberos protocol follows the basic pattern of brokered authentication, but it has properties that differentiate it from other types of brokered authentication, including the following:

- The Kerberos protocol supports the notion of ticket renewal, but it does not automatically revoke tickets. By default, Kerberos tickets have a fixed lifetime of 8 hours; however, the Windows implementation uses a fixed lifetime of 10 hours.
- The KDC does not terminate a service ticket when an authenticated client is finished communicating with a service. Instead, it lets the ticket expire at the end of its normal lifetime. Because tickets are used for authentication, if the ticket expires during communication with a service, the expiration will not affect current operations. Clients are not notified when a ticket is about to expire.

For more information on Kerberos tickets and ticket lifetime, see Kerberos Technical Supplement for Windows in Chapter 7, "Technical Supplements."

The Kerberos protocol can be used for brokering authentication at either the transport layer or message layer. Some implementations of Kerberos authentication include the following:

- Transport-layer Kerberos authentication, which includes:
  - Windows Integrated Security
  - IP Security Protocol with Internet Key Exchange (IPSec/IKE)
- Message-layer Kerberos authentication, which includes:
  - Web Service Enhancements (WSE) 2.0 KerberosToken2
  - Web Service Enhancements (WSE) 3.0 KerberosToken

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: Kerberos pattern include:

- The Kerberos protocol provides SSO capabilities, which allow a client to authenticate only once per logon session.
- The Kerberos protocol has broad acceptance as a brokered authentication protocol and is in use in the majority of large organizations that have a centralized authentication infrastructure.
- The Kerberos protocol is closely integrated with the Windows operating system (Windows 2000 and later). This enables the operating system to provide additional capabilities, such as user impersonation/delegation, authorization, and auditing.
- Kerberos supports mutual authentication when a service sends a response that contains data encrypted with the shared session key.

Liabilities

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

- The centralized nature of the Kerberos protocol requires a KDC, which acts as an authentication broker, to be available at all times. If the KDC fails, clients will not be able to establish new trust relationships with a service. You should consider using redundant KDCs or providing an alternative mechanism, such as X.509 certificates, for authentication. With Active Directory, KDC availability can be improved by establishing secondary domain controllers. This creates a redundant set of Kerberos KDCs.
- The Kerberos protocol is only useful for online authentication and secure communication. Kerberos is not useful for long-term persistence because of the limited lifetime of tickets and session keys used for encryption and signing.
- The Kerberos protocol cannot establish proof of authentication for a client outside of its security realm (Active Directory domain) unless a trust relationship has been established with the other security realm.

Security Considerations

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

- Clients must keep their master keys secret. If an intruder somehow compromises a client's key, it will be able to masquerade as that client or impersonate any server to the legitimate client.
- With the Kerberos protocol, password guessing attacks can occur against messages encrypted with a password equivalent derived from the client's password. (For client authentication, this is the client's password. For service authentication, this is the password of the service account.) The Kerberos protocol uses this derived key to encrypt data in the authentication request. To discover the password, an attacker could mount an offline dictionary attack by repeatedly attempting to decrypt the data in the authentication request sent to the KDC.

- The Kerberos protocol does not implement authorization, although it is typically coupled with an identity store that may store authorization information for a client. Resources may control access based on the client's authorization information, which is contained in the service ticket.
- The Kerberos protocol cannot be used for non-repudiation because the client's identity secret is shared with the KDC.
- Each host on the network must have a clock that is loosely synchronized to the time of the other hosts. This synchronization reduces the bookkeeping needs of application servers when they do replay detection. You can configure the degree of looseness on a per-server basis. If the clocks are synchronized over the network, the clock synchronization protocol itself must be secured from network attackers.

Related Patterns

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

The following parent pattern is related to the Brokered Authentication: Kerberos pattern:

- **Brokered Authentication**. This pattern describes how to prove a client's identity to an authentication broker so that the broker can issue a security token.

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

- **Implementing Brokered Authentication Using Windows Integrated Security on IIS**. This reference provides a concise reference on how to use Windows Integrated Security on IIS.
- **Implementing Message Layer Security with Kerberos in WSE 3.0**. This pattern provides implementation guidelines for using the Kerberos protocol in WSE 3.0 to implement brokered authentication, authorization, data integrity, and data origin authentication.
- **Protocol Transition with Constrained Delegation Technical Supplement**. This technical supplement describes different scenarios for using protocol transition, and then provides step-by-step details for implementing protocol transition. In addition, this pattern describes how a protocol transition can be used with constrained delegation to access downstream resources.

The following alternate patterns are related to the Brokered Authentication: Kerberos pattern:

- **Brokered Authentication: X.509 PKI**. This pattern describes a specialized authentication broker based on the X.509 PKI standard.
- **Brokered Authentication: Security Token Service (STS)**. This pattern describes a specialized authentication broker based on using a security token service.