Erstellen sicherer ASP.NET-Anwendungen

Authentifizierung, Autorisierung und sichere Kommunikation

Kapitel 5 - Intranetsicherheit

J.D. Meier, Alex Mackman, Michael Dunner und Srinath Vasireddy

Microsoft Corporation

Oktober 2002

Auf der Orientierungsseite finden Sie einen Ausgangspunkt und eine vollständige Übersicht zum Erstellen sicherer ASP.NET-Anwendungen.

Zusammenfassung

In diesem Kapitel wird beschrieben, wie häufige Intranetanwendungsszenarien gesichert werden. Es enthält eine Darstellung der Merkmale der einzelnen Szenarien und eine Beschreibung der erforderlichen Schritte, um das Szenario zu sichern. Weitere Informationen finden Sie in den entsprechenden Analyseabschnitten.

Inhalt

Der Zugriff auf Intranetanwendungen ist auf eine begrenzte Gruppe autorisierter Benutzer beschränkt (wie z. B. Mitarbeiter, die zu einer Domäne gehören). Obwohl eine Intraneteinstellung die Offenlegung der Anwendung begrenzt, müssen Sie bei der Entwicklung von Strategien für Authentifizierung, Autorisierung und sichere Kommunikation mehrere Anforderungen berücksichtigen. Eventuell sind nicht-vertrauenswürdige Domänen vorhanden, die die Übermittlung des Sicherheitskontexts und der Identität eines Aufrufers an die Back-End-Ressource im System erschweren. Möglicherweise arbeiten Sie auch in einer heterogenen Umgebung mit verschiedenen Browsertypen, wodurch die Verwendung eines gemeinsamen Authentifizierungsmechanismus weiter erschwert wird.

In einem homogenen Intranet, in dem auf allen Computern das Betriebssystem Microsoft® Windows® 2000 oder höher ausgeführt wird, und bei Vorhandensein einer Domäne, in denen den Benutzern im Hinblick auf eine Delegierung vertraut wird, bildet die Delegierung des Sicherheitskontexts des ursprünglichen Aufrufers an das Back-End eine Möglichkeit.

Sie müssen weiterhin die sichere Kommunikation berücksichtigen. Trotz der Tatsache, dass die Anwendung in einer Intranetumgebung ausgeführt wird, können Sie die über das Netzwerk gesendeten Daten nicht als sicher betrachten. Sie müssen die Daten, die zwischen den Browsern und dem Webserver gesendet werden, sowie die Daten, die zwischen den Anwendungsservern und den Datenbanken gesendet werden, wahrscheinlich sichern.

Anhand der folgenden häufigen Intranetszenarien werden in diesem Kapitel die wesentlichen Techniken für Authentifizierung, Autorisierung und sichere Kommunikation veranschaulicht:

- ASP.NET zu SQL Server
- ASP.NET über Enterprise Services zu SQL Server
- ASP.NET über Webdienste zu SQL Server
- ASP.NET über Remoting zu SQL Server

Zusätzlich wird in diesem Kapitel ein Windows 2000-Delegierungsszenario beschrieben (Übermitteln des ursprünglichen Aufrufers an die Datenbank), in dem der Sicherheitskontext und die Identität des ursprünglichen Aufrufers auf Betriebssystemebene vom Browser über die dazwischen liegenden Web- und Anwendungsserver an die Datenbank gesendet werden.

Hinweis: Bei mehreren Szenarien, die in diesem Kapitel beschrieben werden, wird entweder das Standardkonto ASPNET, mit dem ASP.NET-Anwendungen ausgeführt werden, ersetzt oder dessen Kennwort geändert, damit auf den Remotecomputern duplizierte Konten erstellt werden können. Bei diesen Szenarien wird das <**processModel**>-Element von **Machine.config** geändert. Dies führt dazu, dass die Anmeldeinformationen unverschlüsselt in **machine.config** gespeichert werden. Eine detaillierte Behandlung dieses Themas finden Sie unter "Zugreifen auf Netzwerkressourcen" in Kapitel 8, "ASP.NET-Sicherheit".

ASP.NET zu SQL Server

In diesem Szenario stellt eine Personaldatenbank in einem homogenen Intranet benutzerbezogene Daten sicher bereit. Die Anwendung verwendet ein Modell mit vertrauenswürdigen Subsystemen und führt Aufrufe im Namen der ursprünglichen Aufrufer aus. Die Anwendung authentifiziert die Aufrufer mithilfe der integrierten Windows-Authentifizierung und ruft die Datenbank unter Verwendung der ASP.NET-Prozessidentität aus. Aufgrund der Vertraulichkeit der Daten wird zwischen dem Webserver und den Clients SSL verwendet.

Das grundlegende Modell für dieses Anwendungsszenario ist in Abbildung 5.1 dargestellt.

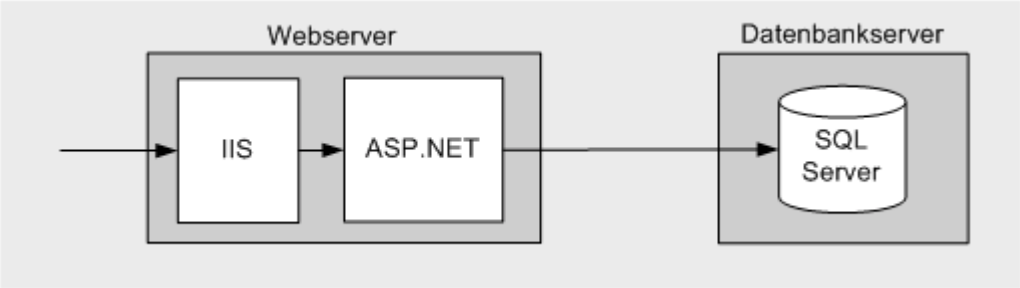

Abbildung 5.1
ASP.NET zu SQL Server

Merkmale

Dieses Szenario weist die folgenden Merkmale auf:

- Die Clients verfügen über Internet Explorer.
- Die Benutzerkonten befinden sich im Verzeichnisdienst Microsoft Active Directory®.
- Die Anwendung stellt vertrauliche benutzerbezogene Daten bereit.
- Auf die Anwendung dürfen nur authentifizierte Clients zugreifen.
- Die Datenbank vertraut der Anwendung, dass die Benutzer ordnungsgemäß authentifiziert werden (d. h. die Anwendung führt die Datenbankaufrufe im Namen der Benutzer durch).
- Microsoft SQL Server™ verwendet für die Autorisierung eine einzige Datenbankbenutzerrolle.

Sichern des Szenarios

In diesem Szenario authentifiziert der Webserver den Aufrufer. Der Zugriff wird anhand der Identität des Aufrufers auf lokale Ressourcen beschränkt. In der Webanwendung muss kein Identitätswechsel durchgeführt werden, um den Ressourcenzugriff für den ursprünglichen Aufrufer einzuschränken. Die Datenbank führt die Authentifizierung anhand der ASP.NET-Standardprozessidentität durch. Dabei handelt es sich um ein Konto mit wenigen Rechten (d. h. die Datenbank vertraut der ASP.NET-Anwendung).

Tabelle 5.1: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	● Bereitstellen einer starken Authentifizierung im Webserver, um die ursprünglichen Aufrufer mit der integrierten Windows-Authentifizierung in IIS zu authentifizieren. ● Verwenden der Windows-Authentifizierung in ASP.NET (kein Identitätswechsel). ● Sichern der Datenbankverbindungen durch Konfiguration von SQL Server für die Windows-Authentifizierung. ● Die Datenbank vertraut dem ASP.NET-Workerprozess bei Aufrufen. Authentifizieren der ASP.NET-Prozessidentität in der Datenbank.
Autorisierung	● Konfigurieren der Ressourcen im Webserver mit ACLs, die an die ursprünglichen Aufrufer gebunden sind. Für eine einfachere Verwaltung werden die Benutzer zu Windows-Gruppen hinzugefügt, die in den ACLs verwendet werden. ● Die Webanwendung führt für den ursprünglichen Aufrufer .NET-Rollenprüfungen durch, um den Zugriff auf Seiten zu beschränken.
Sichere Kommunikation	● Sichern der vertraulichen Daten, die zwischen dem Webserver und der Datenbank gesendet werden. ● Sichern der vertraulichen Daten, die zwischen den ursprünglichen Aufrufern und der Webanwendung gesendet werden.

Das Ergebnis

Abbildung 5.2 zeigt die empfohlene Sicherheitskonfiguration für dieses Szenario.

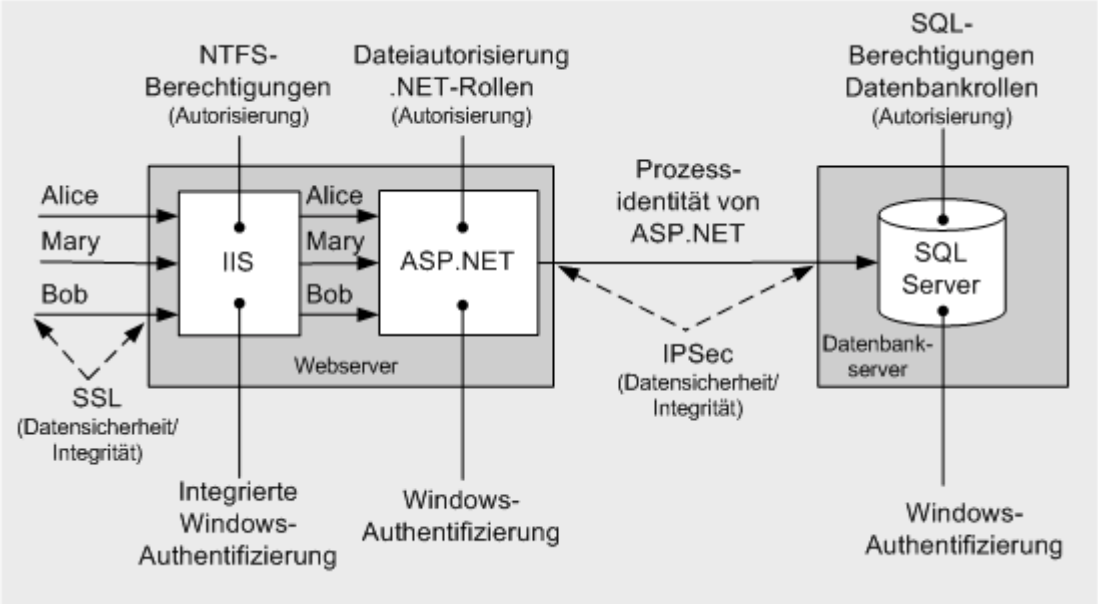

Abbildung 5.2
Intranetszenario: ASP.NET zu SQL Server - Empfohlene Sicherheitskonfiguration

Konfigurationsschritte für die Sicherheit

Bevor Sie beginnen, sollten Sie sich über folgende Themen informieren:

- Erstellen benutzerdefinierter ASP.NET-Konten (siehe "Vorgehensweise: Erstellen eines benutzerdefinierten Kontos zum Ausführen von ASP.NET" im Abschnitt "Referenz" dieses Handbuchs)
- Erstellen eines Datenbankkontos mit möglichst geringen Rechten (siehe Kapitel 12, "Datenzugriffssicherheit")
- Konfigurieren von SSL auf einem Webserver (siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs)
- Konfigurieren von IPSec (siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs)

Konfigurieren von IIS

Schritt	Weitere Informationen
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung Aktivieren der integrierten Windows-Authentifizierung	Die IIS-Authentifizierungseinstellungen können Sie mit dem IIS-MMC-Snap-In bearbeiten. Klicken Sie mit der rechten Maustaste auf das virtuelle Verzeichnis der Anwendung, und klicken Sie dann auf **Eigenschaften**. Klicken Sie auf die Registerkarte **Verzeichnissicherheit**, und klicken Sie dann im Gruppenfeld **Authentifizierung und Zugriffssteuerung** auf **Bearbeiten**.

Konfigurieren von ASP.NET

Schritt	Weitere Informationen

Ändern des ASPNET-Kennworts in einen bekannten starken Kennwortwert	Bei ASPNET handelt es sich um ein lokales Konto mit möglichst geringen Rechten, das standardmäßig für die Ausführung von ASP.NET-Webanwendungen verwendet wird.
	Legen Sie das Kennwort des Kontos ASPNET mit **Lokale Benutzer und Gruppen** auf einen bekannten Wert fest.
	Bearbeiten Sie **Machine.config** in **%windir%\Microsoft.NET\Framework\v1.0.3705\CONFIG,** und ändern Sie die Konfiguration des **password**-Attributs im <**processModel**>-Element.
	Die Standardeinstellung
	
	`<!-- userName="machine" password="AutoGenerate" -->`
	
	wird geändert in
	
	`<!-- userName="machine" password="YourNewStrongPassword" --> `

Konfigurieren der ASP.NET-Webanwendung für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Stammverzeichnis der Anwendung. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`

Sicherstellen, dass Identitätswechsel deaktiviert sind	Identitätswechsel sind standardmäßig deaktiviert. Überprüfen Sie in **Web.config** folgendermaßen, ob sie nicht doch aktiviert sind: `<identity impersonate="false" />` Der gleiche Effekt kann erzielt werden, indem das <**identity**>-Element entfernt wird.

Konfigurieren von SQL Server

Schritt	Weitere Informationen
Erstellen eines Windows-Kontos auf dem SQL Server-Computer, das mit dem ASP.NET-Prozesskonto (ASPNET) übereinstimmt	Benutzername und Kennwort müssen mit dem Konto ASPNET übereinstimmen. Erteilen Sie dem Konto die folgenden Rechte: - Auf diesen Computer vom Netzwerk aus zugreifen - Lokale Anmeldung verweigern - Anmelden als Stapelverarbeitungsauftrag
Konfigurieren von SQL Server für die Windows-Authentifizierung	
Erstellen eines SQL Server-Benutzernamens für das lokale Konto ASPNET	Dadurch wird der Zugriff auf SQL Server gewährt.
Erstellen eines neuen Datenbankbenutzers und Zuordnen des Benutzernamens zum Datenbankbenutzer	Dadurch wird der Zugriff auf die angegebene Datenbank gewährt.
Erstellen einer neuen benutzerdefinierten Datenbankrolle und Hinzufügen des Datenbankbenutzers zur Rolle	
Einrichten der Datenbankberechtigungen für die Datenbankrolle	Gewähren Sie minimale Rechte. Weitere Informationen finden Sie in Kapitel 12, "Datenzugriffssicherheit".

Konfigurieren der sicheren Kommunikation

Schritt	Weitere Informationen
Konfigurieren der Website für SSL	Siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren von IPSec zwischen Webserver und Datenbankserver	Siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs.

Analyse

- In diesem Szenario ist die integrierte Windows-Authentifizierung ideal, da alle Benutzer über Windows-Konten verfügen und Microsoft Internet Explorer verwenden. Der Vorteil der integrierten Windows-Authentifizierung besteht darin, dass das Kennwort des Benutzers niemals über das Netzwerk gesendet wird. Die Anmeldung ist darüber hinaus für den Benutzer transparent, da Windows die Anmeldesitzung des aktuellen interaktiven Benutzers verwendet.
- ASP.NET wird als Konto mit möglichst geringen Rechten ausgeführt, sodass ein mögliches Risiko einer Offenlegung gemindert wird.
- In ASP.NET muss für den ursprünglichen Aufrufer kein Identitätswechsel durchgeführt werden, um .NET-Rollenprüfungen durchzuführen oder die Ressourcen in Windows-ACLs zu sichern. Um für den ursprünglichen Aufrufer .NET-Rollenprüfungen durchzuführen, wird das **WindowsPrincipal**-Objekt, das den ursprünglichen Benutzer darstellt, folgendermaßen aus dem HTTP-Kontext abgerufen:

```
WindowsPrincipal wp = (HttpContext.Current.User as WindowsPrincipal);
if ( wp.IsInRole("Manager") )
{
  // User is authorized to perform manager-specific functionality
}
```

  Das **FileAuthorizationModule** von ASP.NET ermöglicht bei ASP.NET-Dateitypen, die in IIS **aspnet_isapi.dll** zugeordnet sind, ACL-Prüfungen für den ursprünglichen Aufrufer. Bei statischen Dateitypen, wie z. B. JPG-, GIF- und HTM-Dateien, fungiert IIS als Gatekeeper. Die Zugriffsprüfungen werden dann unter Verwendung der Identität des ursprünglichen Aufrufers gegenüber den der Datei zugeordneten NTFS-Berechtigungen durchgeführt.
- Durch Verwendung der Windows-Authentifizierung bei SQL Server vermeiden Sie das Speichern der Anmeldeinformationen in Dateien und deren Übertragung über das Netzwerk an den Datenbankserver.
- Die Verwendung eines duplizierten Windows-Kontos im Datenbankserver (das mit dem lokalen Konto ASPNET übereinstimmt) führt zu einem höheren Verwaltungsaufwand. Wenn das Kennwort auf einem Computer geändert wird, muss es synchronisiert und auf dem anderen Computer aktualisiert werden. In einigen Szenarien können Sie eventuell ein Domänenkonto mit möglichst geringen Rechten verwenden, um die Verwaltung zu vereinfachen.
- Das Verfahren mit einem duplizierten lokalen Konto funktioniert auch bei Vorhandensein eines Firewalls, bei der die für die Windows-Authentifizierung erforderlichen Ports u. U. nicht offen sind. Die Verwendung der Windows-Authentifizierung und von Domänenkonten funktioniert in diesem Szenario eventuell nicht.
- Sie müssen sicherstellen, dass die Windows-Gruppen so feinstufig sind wie die Sicherheitsanforderungen. Da die .NET-rollenbasierte Sicherheit auf der Windows-Gruppenmitgliedschaft basiert, wird bei dieser Lösung vorausgesetzt, dass die Windows-Gruppen entsprechend den Kategorien der Benutzer (mit gleichen Sicherheitsrechten), die auf die Anwendung zugreifen, feinstufig genug eingerichtet sind. Die hier für die Verwaltung von Rollen verwendeten Windows-Gruppen können auf dem betreffenden Computer lokale Gruppen oder Domänengruppen sein
- Benutzerrollen der SQL Server-Datenbank sind SQL Server-Anwendungsrollen vorzuziehen, um die damit verbundenen Probleme bei der Kennwortverwaltung und beim Verbindungspooling zu vermeiden, die bei der Verwendung von SQL-Anwendungsrollen entstehen.

Die Anwendungen aktivieren die SQL-Anwendungsrollen durch Aufrufen einer integrierten gespeicherten Prozedur mit einem Rollennamen und einem Kennwort. Das Kennwort muss daher sicher gespeichert werden. Das Datenbankverbindungspooling muss ebenfalls deaktiviert werden, wenn SQL-Anwendungsrollen verwendet werden, sodass die Skalierbarkeit der Anwendung erheblich beeinträchtigt wird.

Weitere Informationen zu SQL Server-Datenbankbenutzerrollen und SQL Server-Anwendungsrollen finden Sie in Kapitel 12, "Datenzugriffssicherheit".

- Der Datenbankbenutzer wird zu einer Datenbankbenutzerrolle hinzugefügt. Der Rolle werden Berechtigungen zugewiesen, sodass bei Änderungen des Datenbankkontos nicht die Berechtigungen aller Datenbankobjekte geändert werden müssen.

Fragen und Antworten

- **Warum kann für die Webanwendung kein Identitätswechsel aktiviert werden, damit die Ressourcen, auf die die Webanwendung zugreift, mit ACLs gesichert werden, die für den ursprünglichen Aufrufer konfiguriert sind?**

  Wenn Sie Identitätswechsel aktivieren, enthält der Sicherheitskontext, dessen Identität gewechselt wurde, keine Netzwerkanmeldeinformationen (unter der Annahme, dass keine Delegierung aktiviert ist und die integrierte Windows-Authentifizierung verwendet wird). Der Remoteaufruf an SQL Server verwendet daher eine NULL-Sitzung, sodass der Aufruf fehlschlägt. Bei deaktiviertem Identitätswechsel verwendet die Remoteanforderung die ASP.NET-Prozessidentität.

  Das obige Szenario verwendet **FileAuthorizationModule** von ASP.NET, das unter Verwendung der Windows-ACLs eine Autorisierung für die Identität des ursprünglichen Aufrufers durchführt und keinen Identitätswechsel erfordert.

  Wenn Sie die Standardauthentifizierung anstelle der integrierten Windows-Authentifizierung (NTLM) verwenden und keine Identitätswechsel aktivieren, wird bei jedem Aufruf der Datenbank der Sicherheitskontext des ursprünglichen Aufrufers verwendet. Jedes Benutzerkonto (bzw. die Windows-Gruppen, zu denen der Benutzer gehört) würde SQL Server-Benutzernamen benötigen. Die Berechtigungen für Datenbankobjekte müssten gegenüber der Windows-Gruppen (bzw. dem ursprünglichen Aufrufer) gesichert werden.

- **Die Datenbank kennt den ursprünglichen Aufrufer nicht. Wie kann eine Überwachungsliste erstellt werden?**

  Überwachen Sie die Aktivität des Endbenutzers in der Webanwendung, oder übergeben Sie die Identität des Benutzers explizit als Parameter des Aufrufs für den Datenzugriff.

Verwandte Szenarien

Andere Browser als Internet Explorer

Die integrierte Windows-Authentifizierung bei IIS erfordert Internet Explorer. In einer gemischten Browserumgebung stehen normalerweise die folgenden Optionen zur Verfügung:

- **Standardauthentifizierung und SSL** - Die Standardauthentifizierung wird von den meisten Browsern unterstützt. Da die Anmeldeinformationen des Benutzers über das Netzwerk übertragen werden, muss das Szenario mit SSL gesichert werden.
- **Clientzertifikate** - Die einzelnen Clientzertifikate können einem eindeutigen Windows-Konto zugeordnet werden, oder alle Clients können mit einem einzigen Windows-Konto dargestellt werden. Die Verwendung von Clientzertifikaten erfordert ebenfalls SSL.
- **Formularauthentifizierung** - Bei der Formularauthentifizierung können die Anmeldeinformationen anhand eines benutzerdefinierten Datenspeichers, z. B. einer Datenbank oder Active Directory, überprüft werden.

  Bei einer Authentifizierung anhand Active Directory müssen Sie sicherstellen, dass nur die erforderlichen Gruppen abgerufen werden, die zu der Anwendung gehören. Genauso, wie Sie Abfragen an eine Datenbank nicht mit SELECT *-Klauseln vornehmen sollten, sollten Sie ebenfalls nicht blind alle Gruppen von Active Directory abrufen.

Bei der Authentifizierung anhand einer Datenbank müssen Sie die Eingabe, die in SQL-Befehlen verwendet wird, zum Schutz vor SQL Injection-Angriffen sorgfältig analysieren. Anstelle von unverschlüsselten oder verschlüsselten Kennwörtern sollten Sie Kennworthashes (mit Salt-Werten) in der Datenbank speichern.

Weitere Informationen zur Verwendung von SQL Server als Speicher für Anmeldeinformationen und zum Speichern von Kennwörtern in der Datenbank finden Sie in Kapitel 12, "Datenzugriffssicherheit".

Beachten Sie, dass in allen Fällen letztlich SSL verwendet werden muss, wenn Sie nicht die integrierte Windows-Authentifizierung verwenden, bei der die Plattform die Anmeldeinformationen verwaltet. Dieser Vorteil betrifft allerdings nur den Authentifizierungsvorgang. Wenn Sie sicherheitssensitive Daten über das Netzwerk übertragen, müssen Sie weiterhin IPSec oder SSL verwenden.

SQL-Authentifizierung bei der Datenbank

In einigen Szenarien sind Sie u. U. gezwungen, die SQL-Authentifizierung anstelle der bevorzugten Windows-Authentifizierung zu verwenden. Möglicherweise befindet sich zwischen der Webanwendung und der Datenbank ein Firewall, oder der Webserver ist aus Sicherheitsgründen nicht Mitglied der Domäne. Dadurch wird ebenfalls eine Windows-Authentifizierung verhindert. In diesem Fall können Sie die SQL-Authentifizierung zwischen der Datenbank und dem Webserver verwenden. Um dieses Szenario zu sichern, sollten Sie folgendermaßen vorgehen:

- Verwenden der DPAPI (Data Protection API), um Datenbankverbindungszeichenfolgen zu sichern, die Benutzernamen und Kennwörter enthalten. Weitere Informationen finden Sie in den folgenden Ressourcen:
  - "Sicheres Speichern von Verbindungszeichenfolgen" in Kapitel 12, "Datenzugriffssicherheit"
  - "Vorgehensweise: Verwenden von DPAPI (Computerspeicher) von ASP.NET aus" im Abschnitt "Referenz" dieses Handbuchs
  - "Vorgehensweise: Verwenden von DPAPI (Benutzerspeicher) von ASP.NET aus mit Enterprise Services" im Abschnitt "Referenz" dieses Handbuchs
  - "Vorgehensweise: Erstellen einer DPAPI-Bibliothek" im Abschnitt "Referenz" dieses Handbuchs
- Verwenden von IPSec oder SSL zwischen Webserver und Datenbankserver, um die unverschlüsselten Anmeldeinformationen zu schützen, die über das Netzwerk übertragen werden.

Übermitteln des ursprünglichen Aufrufers an die Datenbank

In diesem Szenario werden die Aufrufe von der Webanwendung an die Datenbank mit dem Sicherheitskontext des ursprünglichen Aufrufers durchgeführt. Bei diesem Verfahren müssen Sie Folgendes beachten:

- Bei Auswahl dieses Verfahrens müssen Sie entweder die Kerberos-Authentifizierung (mit für eine Delegierung konfigurierten Konten) oder die Standardauthentifizierung verwenden.

  Ein Delegierungsszenario wird weiter unten in diesem Kapitel unter "Übermitteln des ursprünglichen Aufrufers an die Datenbank" behandelt.
- Darüber hinaus müssen in ASP.NET Identitätswechsel aktiviert werden, sodass der Zugriff auf lokale Systemressourcen mit dem Sicherheitskontext des ursprünglichen Aufrufers erfolgt. Die ACLs für lokale Ressourcen, wie z. B. Registrierung und Ereignisprotokoll, müssen entsprechend konfiguriert werden.
- Das Datenbankverbindungspooling ist begrenzt, da die ursprünglichen Aufrufer Verbindungen nicht gemeinsam nutzen können. Jeder Verbindung wird der Sicherheitskontext des Aufrufers zugeordnet.

- Eine alternative Möglichkeit, den Sicherheitskontext des Benutzers zu übermitteln, besteht darin, die Identität des ursprünglichen Aufrufers auf Anwendungsebene zu übertragen (z. B. über Parameter von Methoden und gespeicherten Prozeduren).

ASP.NET über Enterprise Services zu SQL Server

In diesem Szenario rufen ASP.NET-Seiten Unternehmenskomponenten auf, die in einer Enterprise Services-Anwendung enthalten sind, die wiederum Verbindungen zu einer Datenbank herstellt. Betrachten Sie als Beispiel ein internes Bestellsystem, das Transaktionen über das Intranet verwendet und den internen Abteilungen das Durchführen von Bestellungen ermöglicht. Dieses Szenario ist in Abbildung 5.3 dargestellt.

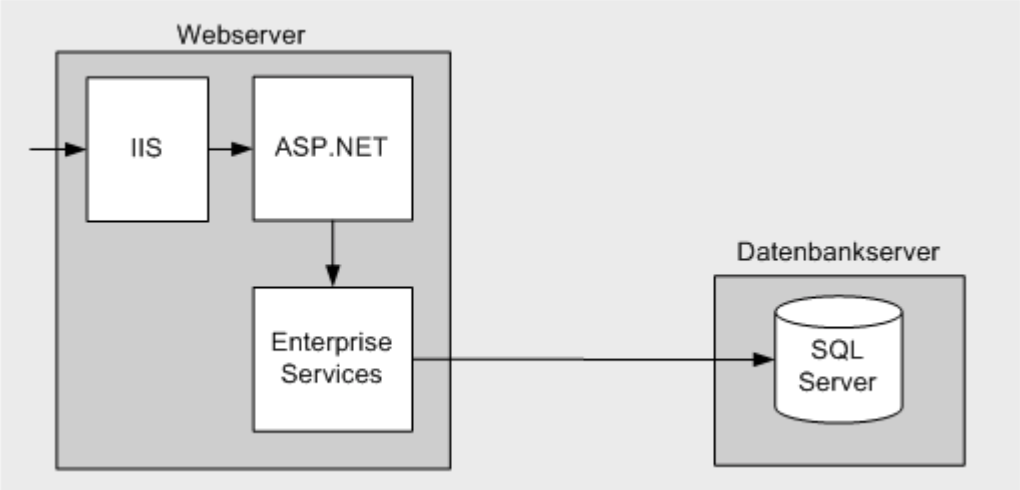

Abbildung 5.3
ASP.NET ruft eine Komponente in Enterprise Services auf, die die Datenbank aufruft

Merkmale

Dieses Szenario weist die folgenden Merkmale auf:

- Die Benutzer verfügen über Internet Explorer.
- Die Komponenten werden auf dem Webserver bereitgestellt.
- Die Anwendung verarbeitet vertrauliche Daten, die bei der Übertragung gesichert werden müssen.
- Die Unternehmenskomponenten stellen unter Verwendung der Windows-Authentifizierung die Verbindung zu SQL Server her.
- Die Unternehmensfunktionalität in diesen Komponenten wird anhand der Identität des Aufrufers beschränkt.
- Die Serviced Components sind als Serveranwendung (prozessextern) konfiguriert.
- Die Komponenten stellen die Verbindung zur Datenbank unter Verwendung der Prozessidentität der Serveranwendung her.
- In ASP.NET sind Identitätswechsel aktiviert (um die rollenbasierte Sicherheit der Enterprise Services zu vereinfachen).

Sichern des Szenarios

In diesem Szenario authentifiziert der Webserver den ursprünglichen Aufrufer. Anschließend wird der Sicherheitskontext des Aufrufers an die Serviced Component übermittelt. Die Serviced Component autorisiert den Zugriff auf die Unternehmensfunktionalität basierend auf der Identität des ursprünglichen Aufrufers. Die Datenbankauthentifizierung erfolgt anhand der Prozessidentität der Enterprise Services-Anwendung (d. h., die Datenbank vertraut den Serviced Components in der Enterprise Services-Anwendung). Wenn die Serviced Component die Datenbank aufruft, wird die Identität des Benutzers auf Anwendungsebene (unter Verwendung vertrauenswürdiger Abfrageparameter) übergeben.

Tabelle 5.2: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	● Stellen Sie auf dem Webserver mithilfe der integrierten Windows-Authentifizierung eine starke Authentifizierung bereit. ● Übermitteln Sie den Sicherheitskontext des ursprünglichen Aufrufers an die Serviced Component, damit Prüfungen der Enterprise Services (COM+)-Rollen unterstützt werden. ● Sichern Sie die Verbindungen an die Datenbank mit der Windows-Authentifizierung. ● Die Datenbank vertraut der Identität der Serviced Component bei Datenbankaufrufen. Die Datenbank authentifiziert die Prozessidentität der Enterprise Services-Anwendung.
Autorisierung	● Autorisieren Sie den Zugriff auf die Geschäftslogik mithilfe von Enterprise Services (COM+)-Rollen.
Sichere Kommunikation	● Die vertraulichen Daten, die zwischen den Benutzern und der Webanwendung gesendet werden, werden mit SSL gesichert. ● Die vertraulichen Daten, die zwischen dem Webserver und der Datenbank gesendet werden, werden mit IPSec gesichert.

Das Ergebnis

Abbildung 5.4 zeigt die empfohlene Sicherheitskonfiguration für dieses Szenario.

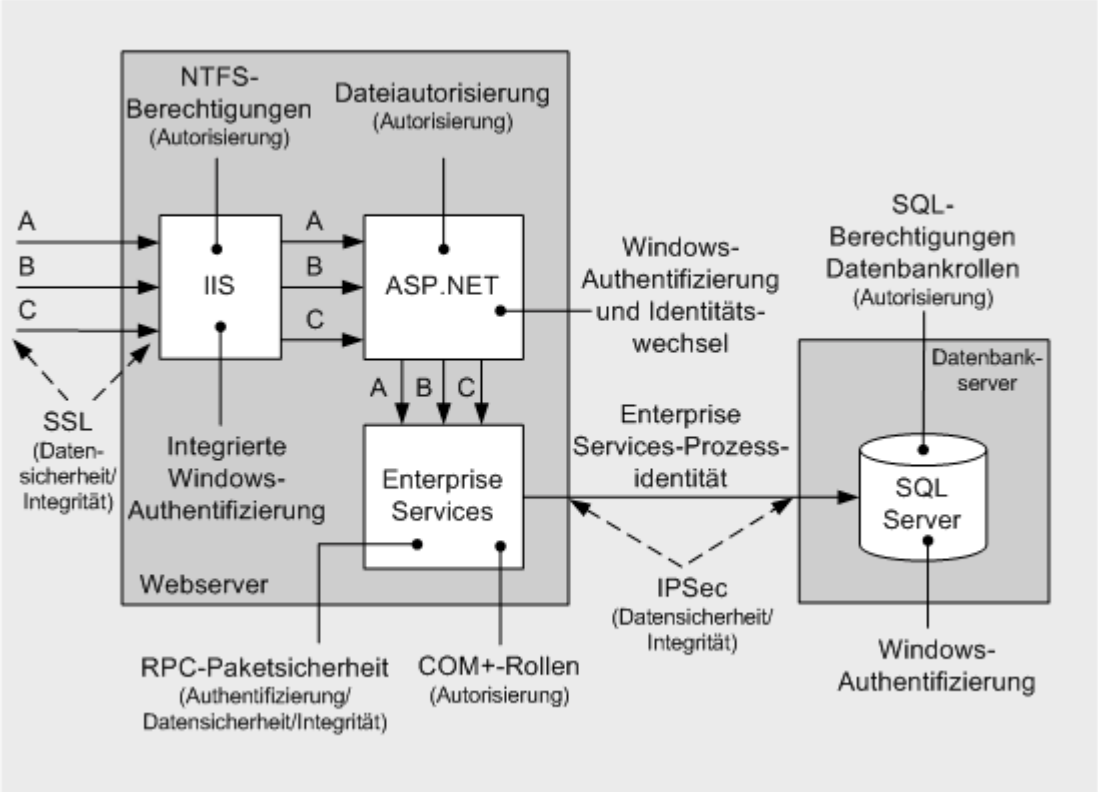

Abbildung 5.4
Intranetszenario: ASP.NET über lokale Enterprise Services zu SQL Server - Empfohlene Sicherheitskonfiguration

Konfigurationsschritte für die Sicherheit

Bevor Sie beginnen, sollten Sie sich über folgende Themen informieren:

- Erstellen eines Datenbankkontos mit möglichst geringen Rechten (siehe Kapitel 12, "Datenzugriffssicherheit")
- Konfigurieren von SSL auf einem Webserver (siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs)
- Konfigurieren von IPSec (siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs)
- Konfigurieren der Enterprise Services-Sicherheit (siehe "Vorgehensweise: Verwenden von rollenbasierter Sicherheit mit Enterprise Services" im Abschnitt "Referenz" dieses Handbuchs)

Konfigurieren von IIS

Schritt	Weitere Informationen
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung	
Aktivieren der integrierten Windows-Authentifizierung	

Konfigurieren von ASP.NET

Schritt	Weitere Informationen
Konfigurieren der ASP.NET-Webanwendung für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Stammverzeichnis der Anwendung. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`
Konfigurieren der ASP.NET-Webanwendung für Identitätswechsel	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis der Webanwendung. Legen Sie das <**identity**>-Element fest auf: `<identity impersonate="true" />`

Konfigurieren der DCOM-Sicherheit von ASP.NET, um sicherzustellen, dass bei Aufrufen der Enterprise Services Identitätswechsel der Aufrufer unterstützt werden	Bearbeiten Sie **Machine.config**, und suchen Sie das <**processModel**>-Element. Vergewissern Sie sich, dass das **comImpersonationLevel**-Attribute auf **Impersonate** eingestellt ist (Standardeinstellung)
	
	`<processModel`
	`  comImpersonationLevel="Impersonate"`

Konfigurieren der Enterprise Services

Schritt	Weitere Informationen
Erstellen eines benutzerdefinierten Kontos für die Ausführung von Enterprise Services	**HINWEIS:** Wenn Sie ein lokales Konto verwenden, müssen Sie auf dem SQL Server-Computer ebenfalls ein dupliziertes Konto erstellen.
Konfigurieren der Enterprise Services-Anwendung als Serveranwendung	Diese Konfiguration kann im Tool für Komponentendienste oder durch Einfügen des folgenden .NET-Attributs in die Serviced Component-Assembly erfolgen. `[assembly: ApplicationActivation(ActivationOption.Server)]`
Konfigurieren der Enterprise Services (COM+)-Rollen	Verwenden Sie das Tool für Komponentendienste oder ein Skript, um Windows-Benutzer und/oder –Gruppen zu Rollen hinzuzufügen. Die Rollen können mithilfe von .NET-Attributen in der Serviced Component-Assembly definiert werden.
Konfigurieren der Ausführung von Enterprise Services mit dem benutzerdefinierten Konto	Diese Konfiguration muss mit dem Tool für Komponentendienste oder einem Skript durchgeführt werden. In der Serviced Component-Assembly können keine .NET-Attribute verwendet werden.

Konfigurieren von SQL Server

Schritt	Weitere Informationen
Erstellen eines Windows-Kontos auf dem SQL Server-Computer, das mit dem Enterprise Services-Prozesskonto übereinstimmt	Benutzername und Kennwort müssen mit dem benutzerdefinierten Enterprise Services-Konto übereinstimmen. Erteilen Sie dem Konto die folgenden Rechte: - Auf diesen Computer vom Netzwerk aus zugreifen - Lokale Anmeldung verweigern - Anmelden als Stapelverarbeitungsauftrag
Konfigurieren von SQL Server für die Windows-Authentifizierung	
Erstellen eines SQL Server-Benutzernamens für das Enterprise Services-Konto	Dadurch wird der Zugriff auf SQL Server gewährt.
Erstellen eines neuen Datenbankbenutzers und Zuordnen des Benutzernamens zum Datenbankbenutzer	Dadurch wird der Zugriff auf die angegebene Datenbank gewährt.
Erstellen einer neuen Datenbankbenutzerrolle und Hinzufügen des Datenbankbenutzers zur Rolle	
Einrichten der Datenbankberechtigungen für die Datenbankbenutzerrolle	Gewähren Sie minimale Rechte. Weitere Informationen finden Sie in Kapitel 12, "Datenzugriffssicherheit".

Konfigurieren der sicheren Kommunikation

Schritt	Weitere Informationen
Konfigurieren der Website für SSL	Siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren von IPSec zwischen Webserver und Datenbankserver	Siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs.

Analyse

- ASP.NET und Enterprise Services werden als Konten mit möglichst geringen Rechten ausgeführt, sodass ein mögliches Risiko einer Offenlegung gemindert wird. Wenn eine der beiden Prozessidentitäten offengelegt wird, wird der Schaden durch die begrenzten Rechte des Kontos reduziert. Wenn im Fall von ASP.NET ein böswilliges Skript eingeschleust wird, ist der mögliche Schaden ebenfalls eingeschränkt.
- Die ASP.NET-Anwendung muss für Identitätswechsel konfiguriert werden, damit der Sicherheitskontext des ursprünglichen Anrufers an die Enterprise Services-Komponenten übermittelt wird (zur Unterstützung der Enterprise Services (COM+)-rollenbasierten Autorisierung). Wenn kein Identitätswechsel erfolgt, werden die Rollen anhand der Prozessidentität (des ASP.NET-Workerprozesses) überprüft. Der Identitätswechsel wirkt sich darauf aus, welche Identität für die Autorisierung des Ressourcenzugriffs verwendet wird.
    - Ohne Identitätswechsel werden die Prüfungen für die Systemressourcen anhand der ASP.NET-Prozessidentität durchgeführt. Mit Identitätswechsel werden die Prüfungen für die Systemressourcen anhand des ursprünglichen Aufrufers durchgeführt. Weitere Informationen zum Zugriff auf Systemressourcen von ASP.NET finden Sie unter "Zugreifen auf Systemressourcen" in Kapitel 8, "ASP.NET-Sicherheit".
- Aufgrund der Verwendung von Enterprise Services (COM+)-Rollen werden die Zugriffsprüfungen in die mittlere Ebene verlagert, in der sich die Geschäftslogik befindet. In diesem Fall werden die Aufrufer am Gate überprüft und den Rollen zugeordnet. Aufrufe der Geschäftslogik basieren dann auf den Rollen. Dadurch werden unnötige Aufrufe an das Back-End vermieden. Ein weiterer Vorteil von Enterprise Services (COM+)-Rollen liegt darin, dass Sie die Rollen bei der Entwicklung mit der Komponentendienstverwaltung erstellen und verwalten können.
- Durch die Windows-Authentifizierung bei SQL vermeiden Sie das Speichern der Anmeldeinformationen in Dateien und deren Übertragung über das Netzwerk.
- Die Verwendung eines lokalen Kontos für die Ausführung der Enterprise Services-Anwendung zusammen mit einem duplizierten Konto auf dem Datenbankserver funktioniert auch bei Vorhandensein eines Firewalls, wenn die für die Windows-Authentifizierung erforderlichen Ports u. U. nicht offen sind. Die Verwendung der Windows-Authentifizierung und von Domänenkonten funktioniert in diesem Szenario eventuell nicht.

Fallstricke

- Die Verwendung eines duplizierten Windows-Kontos im Datenbankserver (das mit dem Enterprise Services-Prozesskonto übereinstimmt) führt zu einem höheren Verwaltungsaufwand. Die Kennwörter sollten regelmäßig von Hand aktualisiert und synchronisiert werden.
- Da die .NET-rollenbasierte Sicherheit auf der Windows-Gruppenmitgliedschaft basiert, wird bei dieser Lösung vorausgesetzt, dass die Windows-Gruppen entsprechend den Kategorien der Benutzer (mit gleichen Sicherheitsrechten), die auf die Anwendung zugreifen, feinstufig genug eingerichtet sind.

ASP.NET über Webdienste zu SQL Server

In diesem Szenario stellt ein Webserver, auf dem ASP.NET-Seiten ausgeführt werden, Verbindungen zu einem Webdienst auf einem Remoteserver her. Dieser Server wiederum stellt Verbindungen zu einem Remotedatenbankserver her. Nehmen Sie als Beispiel eine Personalwebanwendung an, die vertrauliche benutzerbezogene Daten bereitstellt. Die Anwendung benötigt den Webdienst für den Abruf der Daten. Das grundlegende Modell für dieses Anwendungsszenario ist in Abbildung 5.5 dargestellt.

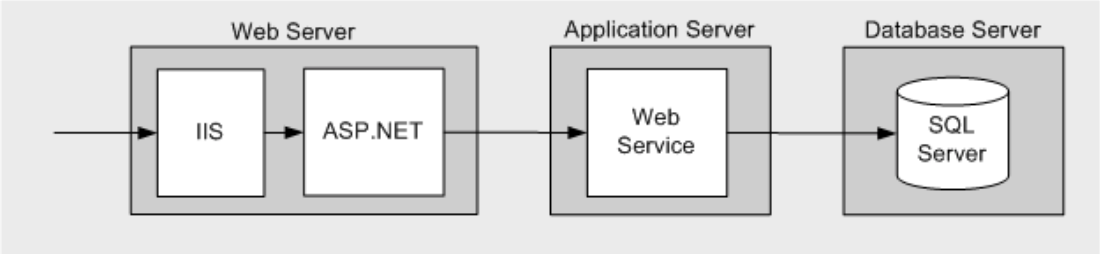

Abbildung 5.5
ASP.NET über Remotewebdienst zu SQL Server

Der Webdienst stellt eine Methode bereit, die es einem einzelnen Mitarbeiter ermöglicht, die eigenen persönlichen Details abzurufen. Die Details dürfen über die Webanwendung nur authentifizierten Personen bereitgestellt werden. Der Webdienst stellt außerdem eine Methode bereit, die den Abruf von beliebigen Mitarbeiterdetails unterstützt. Diese Funktionalität darf nur den Mitgliedern der Personal- oder Gehaltsabteilung zur Verfügung stehen. In diesem Szenario werden die Mitarbeiter in drei Windows-Gruppen kategorisiert:

- **HRDept** (Mitglieder der Personalabteilung)
  Die Mitglieder dieser Gruppe können Details zu allen Mitarbeitern abrufen.
- **PayrollDept** (Mitglieder der Gehaltsabteilung)
  Die Mitglieder dieser Gruppe können Details zu allen Mitarbeitern abrufen.
- **Employees** (alle Mitarbeiter)
  Die Mitglieder dieser Gruppe können lediglich die eigenen Details abrufen.

Aufgrund der Vertraulichkeit der Daten muss der Datenverkehr zwischen allen Knoten gesichert werden.

Merkmale

- Die Benutzer verfügen über Internet Explorer 5.x oder höher.
- Auf allen Computern wird Windows 2000 oder höher ausgeführt.
- Die Benutzerkonten befinden sich in Active Directory in einer einzigen Gesamtstruktur.
- Die Anwendung überträgt den Sicherheitskontext des ursprünglichen Aufrufers bis an die Datenbank.
- In allen Ebenen wird die Windows-Authentifizierung verwendet.
- Die Domänenbenutzerkonten sind für eine Delegierung konfiguriert.
- Die Datenbank unterstützt keine Delegierung.

Sichern des Szenarios

In diesem Szenario authentifiziert der Webserver, auf dem sich die ASP.NET-Webanwendung befindet, die Identität des ursprünglichen Aufrufers. Anschließend wird der Sicherheitskontext an den Remoteserver übermittelt, auf dem sich der Webdienst befindet. Dadurch können Autorisierungsprüfungen für Webmethoden durchgeführt werden, um dem ursprünglichen Aufrufer den Zugriff zu gestatten oder zu verweigern. Die Datenbankauthentifizierung erfolgt anhand der Prozessidentität des Webdienstes (die Datenbank vertraut dem Webdienst). Der Webdienst ruft wiederum die Datenbank auf und übergibt die Identität des Benutzers auf Anwendungsebene mithilfe von Parametern von gespeicherten Prozeduren.

Tabelle 5.3: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	● Die Webanwendung authentifiziert die Benutzer mithilfe der integrierten Windows-Authentifizierung von IIS. ● Der Webdienst verwendet die integrierten Windows-Authentifizierung von IIS und authentifiziert den Sicherheitskontext des ursprünglichen Aufrufers, der von der Webanwendung delegiert wurde. ● Mit dem Kerberos-Authentifizierungsprotokoll wird der Sicherheitskontext des ursprünglichen Aufrufers von der Webanwendung mit Delegierung an den Webdienst übermittelt. ● Für die Verbindung zur Datenbank wird die Windows-Authentifizierung mit dem ASP.NET-Prozesskonto verwendet.
Autorisierung	● Die Webanwendung führt für den ursprünglichen Aufrufer Rollenprüfungen durch, um den Zugriff auf Seiten zu beschränken. ● Der Zugriff auf die Methoden des Webdienstes wird mit .NET-Rollen gesteuert, die auf der Windows-Gruppenmitgliedschaft des ursprünglichen Aufrufers basieren.
Sichere Kommunikation	● Die vertraulichen Daten, die zwischen den ursprünglichen Aufrufern, der Webanwendung und dem Webdienst gesendet werden, werden mit SSL gesichert. ● Die vertraulichen Daten, die zwischen dem Webdienst und der Datenbank gesendet werden, werden mit IPSec gesichert.

Das Ergebnis

Abbildung 5.6 zeigt die empfohlene Sicherheitskonfiguration für dieses Szenario.

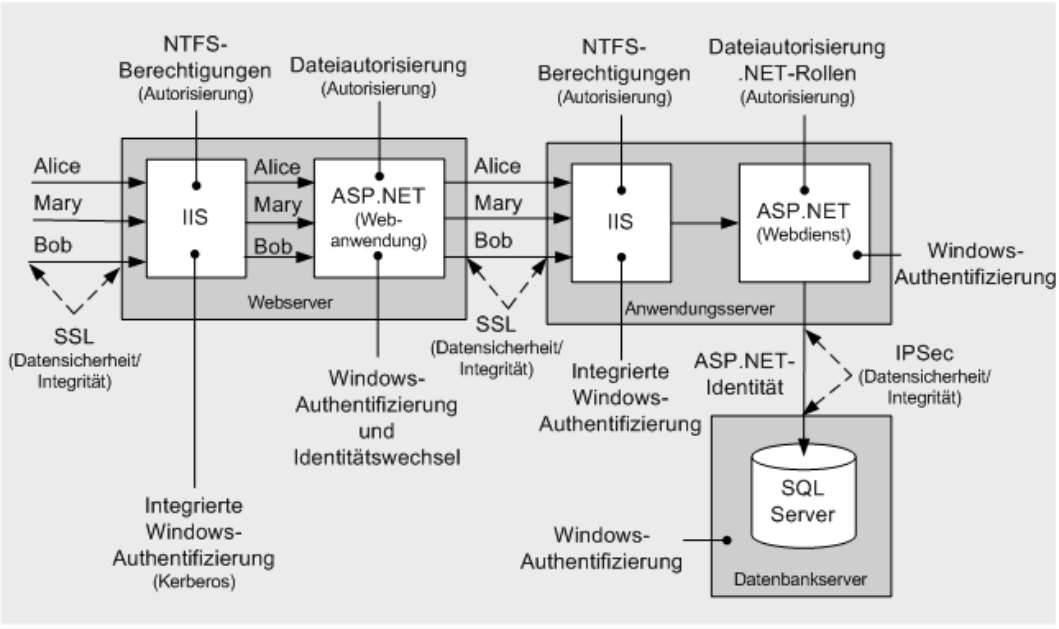

Abbildung 5.6

Intranetszenario: ASP.NET über Webdienst zu SQL Server - Empfohlene Sicherheitskonfiguration

Konfigurationsschritte für die Sicherheit

Bevor Sie beginnen, sollten Sie sich über folgende Themen informieren:

- Konfigurieren von SSL auf einem Webserver (siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs)
- Konfigurieren von IPSec (siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs)

Konfigurieren des Webservers (auf dem sich die Webanwendung befindet)

Konfigurieren von IIS	
Schritt	Weitere Informationen
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung Aktivieren der integrierten Windows-Authentifizierung für das virtuelle Stammverzeichnis der Webanwendung	
Konfigurieren von ASP.NET	
Schritt	Weitere Informationen
Konfigurieren der ASP.NET-Webanwendung für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis der Webanwendung. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`
Konfigurieren der ASP.NET-Webanwendung für Identitätswechsel	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis der Webanwendung. Legen Sie das <**identity**>-Element fest auf: `<identity impersonate="true" />`

Konfigurieren des Anwendungsservers (auf dem sich der Webdienst befindet)

Konfigurieren von IIS	
Schritt	**Weitere Informationen**
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis des Webdienstes Aktivieren der integrierten Windows-Authentifizierung für das virtuelle Stammverzeichnis des Webdienstes	

Konfigurieren von ASP.NET	
Schritt	**Weitere Informationen**

Ändern des ASPNET-Kennworts in einen bekannten Wert	Bei ASPNET handelt es sich um ein lokales Konto mit möglichst geringen Rechten, das standardmäßig für die Ausführung von ASP.NET-Webanwendung verwendet wird.
	Legen Sie das Kennwort des Kontos ASPNET mit **Lokale Benutzer und Gruppen** auf einen bekannten Wert fest.
	Bearbeiten Sie **Machine.config** in **%windir%\Microsoft.NET\Framework\v1.0.3705\CONFIG**, und ändern Sie die Konfiguration des **password**-Attributs im <**processModel**>-Element.
	Die Standardeinstellung
	
	`<!-- userName="machine" password="AutoGenerate" -->`
	
	wird geändert in
	
	`<!-- userName="machine" password="YourNewStrongPassword" --> `

Konfigurieren des ASP.NET-Webdienstes für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis des Webdienstes. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`
Sicherstellen, dass Identitätswechsel deaktiviert sind	Identitätswechsel sind standardmäßig deaktiviert. Überprüfen Sie in **Web.config** folgendermaßen, ob sie nicht doch aktiviert sind: `<identity impersonate="false" />` Da Identitätswechsel standardmäßig deaktiviert sind, kann der gleiche Effekt erzielt werden, indem das <**identity**>-Element entfernt wird.

Konfigurieren von SQL Server

Schritt	Weitere Informationen
Erstellen eines Windows-Kontos auf dem SQL Server-Computer, das dem ASP.NET-Prozesskonto entspricht, mit dem der Webdienst ausgeführt wird	Benutzername und Kennwort müssen mit dem benutzerdefinierten ASP.NET-Konto übereinstimmen. Erteilen Sie dem Konto die folgenden Rechte: - Auf diesen Computer vom Netzwerk aus zugreifen - Lokale Anmeldung verweigern - Anmelden als Stapelverarbeitungsauftrag
Konfigurieren von SQL Server für die Windows-Authentifizierung	
Erstellen eines SQL Server-Benutzernamens für das benutzerdefinierte ASP.NET-Konto	Dadurch wird der Zugriff auf SQL Server gewährt.
Erstellen eines neuen Datenbankbenutzers und Zuordnen des Benutzernamens zum Datenbankbenutzer	Dadurch wird der Zugriff auf die angegebene Datenbank gewährt.
Erstellen einer neuen Datenbankbenutzerrolle und Hinzufügen des Datenbankbenutzers zur Rolle	
Einrichten der Datenbankberechtigungen für die Datenbankbenutzerrolle	Gewähren Sie minimale Rechte.

Konfigurieren der sicheren Kommunikation

Schritt	Weitere Informationen
Konfigurieren der Website auf dem Webserver für SSL	Siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren von IPSec zwischen Webserver und Datenbankserver	Siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs.

Analyse

- Die integrierte Windows-Authentifizierung in IIS ist in diesem Szenario ideal, da alle Benutzer Windows 2000 oder höher und Internet Explorer 5.x oder höher verwenden und die Konten in Active Directory enthalten sind, sodass das Kerberos-Authentifizierungsprotokoll (das die Delegierung unterstützt) verwendet werden kann. Der Sicherheitskontext des Benutzers kann so über Computergrenzen hinweg übertragen werden.
- Die Konten der Endbenutzer dürfen in Active Directory NICHT als **Konto kann nicht delegiert werden** gekennzeichnet werden. Das Konto für den Webservercomputer muss in Active Directory als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet werden. Weitere Informationen finden Sie unter "Vorgehensweise: Implementieren der Kerberos-Delegierung unter Windows 2000" im Abschnitt "Referenz" dieses Handbuchs.
- ASP.NET wird auf dem Webserver und dem Anwendungsserver als lokales Konto mit möglichst geringen Rechten (dem lokalen Konto ASPNET) ausgeführt, sodass ein mögliches Risiko einer Offenlegung gemindert wird.
- Webdienst und Webanwendung sind für die Windows-Authentifizierung konfiguriert. IIS ist auf beiden Computern für die integrierte Windows-Authentifizierung konfiguriert.
- Wenn der Webdienst von der Webanwendung aufgerufen wird, werden standardmäßig keine Anmeldeinformationen übergeben. Sie werden jedoch benötigt, um auf dem nachgeschalteten Webserver auf die Netzwerkauthentifizierungsherausforderung zu reagieren, die von IIS ausgegeben wird. Dies muss explizit angegeben werden, indem die **Credentials**-Eigenschaft des Webdienstproxys folgendermaßen eingestellt wird:

  ```
  wsproxy.Credentials = CredentialCache.DefaultCredentials;
  ```

  Weitere Informationen zum Aufrufen von Webdiensten mit Anmeldeinformationen finden Sie in Kapitel 10, "Webdienstsicherheit".
- Die Webanwendung ist für Identitätswechsel konfiguriert. Daher wird bei Aufrufen der Webanwendung an den Webdienst der Sicherheitskontext des ursprünglichen Aufrufers übergeben, sodass der Webdienst den ursprünglichen Aufrufer authentifizieren (und autorisieren) kann.
- Die Benutzer werden im Webdienst mit .NET-Rollen autorisiert, die auf der Windows-Gruppe basieren, zu der die Benutzer gehören (**HRDept**, **PayrollDept** und **Employees**). Die Mitglieder von **HRDept** und **PayrollDept** können Mitarbeiterdetails für alle Mitglieder abrufen, während die Mitglieder der Gruppe **Employees** lediglich ihre eigenen Details abrufen können.
  Die Webmethoden können mit der **PrincipalPermissionAttribute**-Klasse erweitert werden, um eine bestimmte Rollenmitgliedschaft zu fordern (siehe folgendes Codebeispiel). Beachten Sie, dass **PrincipalPermission** anstelle von **PrincipalPermissionAttribute** verwendet werden kann. Hierbei handelt es sich um ein allgemeines Feature aller .NET-Attributtypen.

  ```
  [WebMethod]
  [PrincipalPermission(SecurityAction.Demand,
                       Role=@"DomainName\HRDept)]
  public DataSet RetrieveEmployeeDetails()
  {
  }
  ```

  Das in dem obigen Code gezeigte Attribut besagt, dass nur Mitglieder der Windows-Gruppe **Domänenname\HRDept** die **RetrieveEmployeeDetails**-Methode aufrufen dürfen. Wenn ein Nichtmitglied versucht, die Methode aufzurufen, wird eine Sicherheitsausnahme ausgelöst.

- Die ASP.NET-Dateiautorisierung (in der Webanwendung und im Webdienst) prüft für den Aufrufer die ACL für alle Dateitypen, für die in der IIS-Metabasis eine Zuordnung vorhanden ist, die den Dateityp **Aspnet_isapi.dll** zuordnet. Statische Dateitypen (wie JPG, GIF, HTM usw.), für die keine ISAPI-Zuordnung vorhanden ist, werden von IIS (anhand der ACL für die jeweilige Datei) geprüft.
- Da die Webanwendung für Identitätswechsel konfiguriert ist, müssen die Ressourcen, auf die die Anwendung zugreift, mit einer ACL konfiguriert sein, die zumindest einen Lesezugriff des ursprünglichen Aufrufers zulässt.
- Der Webdienst führt keinen Identitätswechsel und keine Delegierung durch. Der Zugriff auf die lokalen Systemressourcen und die Datenbank erfolgt daher mit der ASP.NET-Prozessidentität. Alle Aufrufe werden folglich nur mit dem Prozesskonto durchgeführt. Dadurch wird die Verwendung des Datenbankverbindungspoolings ermöglicht. Wenn die Datenbank keine Delegierung unterstützt (wie z. B. SQL Server 7.0 oder früher), bildet dieses Szenario eine gute Möglichkeit.
- Durch die Windows-Authentifizierung gegenüber SQL Server wird das Speichern der Anmeldeinformationen auf dem Webserver sowie das Senden der Anmeldeinformationen über das Netzwerk an den SQL Server-Computer vermieden.
- Zwischen dem ursprünglichen Aufrufer und dem Webserver schützt SSL die Daten, die zu und von der Webanwendung übertragen werden.
- IPSec zwischen dem nachgeschalteten Webserver und der Datenbank schützt die von und zu der Datenbank übertragenen Daten.

Fallstricke

- Die Verwendung eines duplizierten Windows-Kontos im Datenbankserver (das mit dem ASP.NET-Prozesskonto übereinstimmt) führt zu einem höheren Verwaltungsaufwand. Die Kennwörter sollten regelmäßig von Hand aktualisiert und synchronisiert werden.

  Als Alternative sollten Sie die Verwendung von Domänenkonten mit möglichst geringen Rechten erwägen. Weitere Informationen zum Auswählen einer ASP.NET-Prozessidentität finden Sie in Kapitel 9, "ASP.NET Sicherheit“. Da die .NET-rollenbasierte Sicherheit auf der Windows-Gruppenmitgliedschaft basiert, wird bei dieser Lösung vorausgesetzt, dass die Windows-Gruppen entsprechend den Kategorien der Benutzer (mit gleichen Sicherheitsrechten), die auf die Anwendung zugreifen, feinstufig genug eingerichtet sind.
- Die Kerberos-Delegierung unterliegt keinen Einschränkungen. Daher muss sorgfältig gesteuert werden, welche Anwendungsidentitäten auf dem Webserver ausgeführt werden. Um die Sicherheit weiter zu erhöhen, können Sie den Gültigkeitsbereich des Domänenkontos begrenzen, indem das Konto aus der Gruppe **Domänenbenutzer** entfernt und der Zugriff nur von den entsprechenden Computern zugelassen wird. Weitere Informationen finden Sie in dem englischsprachigen Whitepaper "Default Access Control Settings".

Fragen und Antworten

Die Datenbank kennt den ursprünglichen Aufrufer nicht. Wie kann eine Überwachungsliste erstellt werden?

Überwachen Sie die Aktivität des Endbenutzers im Webdienst, oder übergeben Sie die Identität des Benutzers explizit als Parameter des Aufrufs für den Datenzugriff.

Verwandte Szenarien

Wenn Sie Verbindungen zu anderen Datenbanken als SQL herstellen müssen oder momentan die SQL-Authentifizierung verwendet wird, müssen Sie die Anmeldeinformationen des Datenbankkontos explizit in der Verbindungszeichenfolge übergeben. In diesem Fall müssen Sie sicherstellen, dass die Verbindungszeichenfolge sicher gespeichert wird.

Weitere Informationen finden Sie unter "Datenzugriffssicherheit" in Kapitel 12, "Datenzugriffssicherheit".

ASP.NET über Remoting zu SQL Server

In diesem Szenario stellt ein Webserver, auf dem ASP.NET-Seiten ausgeführt werden, sichere Verbindungen zu einer Remotekomponente auf einem Remoteanwendungsserver her. Der Webserver kommuniziert mithilfe von .NET Remoting über den HTTP-Kanal mit der Komponente. Die Remotekomponente wird von ASP.NET gehostet (siehe Abbildung 5.7).

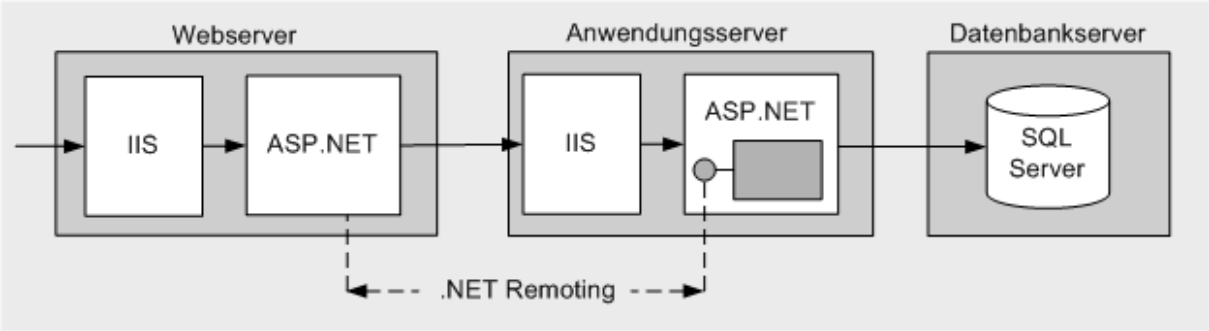

Abbildung 5.7
ASP.NET zu Remoting mithilfe von .NET Remoting zu SQL Server

Merkmale

- Die Benutzer verfügen über verschiedene Typen von Webbrowsern.
- Die Remotekomponente wird von ASP.NET gehostet.
- Die Webanwendung kommuniziert über den HTTP-Kanal mit der Remotekomponente.
- Die ASP.NET-Anwendung ruft die .NET-Remotekomponente auf und übergibt die Anmeldeinformationen des ursprünglichen Aufrufers zur Authentifizierung. Die Anmeldeinformationen stehen aus der Standardauthentifizierung zur Verfügung.
- Die Daten sind vertraulich und müssen daher zwischen den Prozessen und Computern gesichert werden.

Sichern des Szenarios

In diesem Szenario authentifiziert der Webserver, auf dem sich die ASP.NET-Webanwendung befindet, die ursprünglichen Aufrufer. Die Webanwendung kann die Anmeldeinformationen des Aufrufers für die Authentifizierung (Benutzername und Kennwort) aus HTTP-Servervariablen abrufen. Sie können anschließend zum Herstellen der Verbindung zum Anwendungsserver verwendet werden, auf dem sich die Remotekomponente befindet, indem der Proxy für die Remotekomponente entsprechend konfiguriert wird. Die Datenbank verwendet die Windows-Authentifizierung zur Authentifizierung der ASP.NET-Prozessidentität (d. h. die Datenbank vertraut der Remotekomponente). Die Remotekomponente ruft wiederum die Datenbank auf und übergibt die Identität des ursprünglichen Aufrufers auf Anwendungsebene mithilfe von Parametern von gespeicherten Prozeduren.

Tabelle 5.4: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	● Authentifizieren Sie die Benutzer mit der Standardauthentifizierung von IIS (zusätzlich zu SSL). ● Verwenden Sie die Windows-Authentifizierung von der Remotekomponente (ASP.NET/IIS). ● Verwenden Sie die Windows-Authentifizierung, um die Verbindung zu der Datenbank mit einem ASP.NET-Konto mit möglichst geringen Rechten herzustellen.
Autorisierung	● ACL-Prüfungen für den ursprünglichen Aufrufer auf dem Webserver. ● Rollenprüfungen für den ursprünglichen Aufrufer in der Remotekomponente. ● Datenbankberechtigungen für die ASP.NET-Identität (Remotekomponente).
Sichere Kommunikation	● Die vertraulichen Daten, die zwischen den Benutzern, der Webanwendung und Remoteobjekten, die sich in IIS befinden, gesendet werden, werden mit SSL gesichert. ● Die vertraulichen Daten, die zwischen dem Webserver und der Datenbank gesendet werden, werden mit IPSec gesichert.

Das Ergebnis

Abbildung 5.8 zeigt die empfohlene Sicherheitskonfiguration für dieses Szenario.

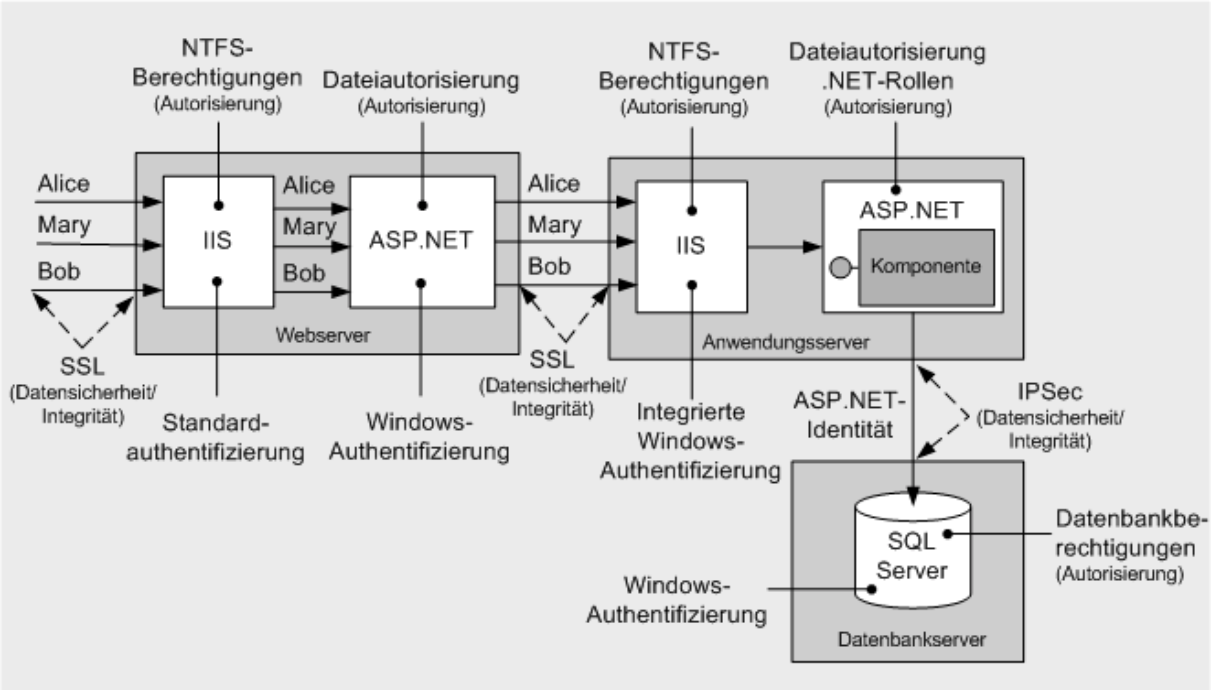

Abbildung 5.8
Intranetszenario: ASP.NET über Remotewebdienst zu SQL Server - Empfohlene Sicherheitskonfiguration

Konfigurationsschritte für die Sicherheit

Bevor Sie beginnen, sollten Sie sich über folgende Themen informieren:

- Erstellen eines Datenbankkontos mit möglichst geringen Rechten (siehe Kapitel 12, "Datenzugriffssicherheit")
- Konfigurieren von SSL auf einem Webserver (siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs)
- Konfigurieren von IPSec (siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs)

Konfigurieren des Webservers

Konfigurieren von IIS	
Schritt	Weitere Informationen
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung	
Aktivieren der Standardauthentifizierung	Schützen Sie die Anmeldeinformationen für die Standardauthentifizierung mit SSL.
Konfigurieren von ASP.NET	
Schritt	Weitere Informationen
Konfigurieren der ASP.NET-Webanwendung für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Stammverzeichnis der Anwendung. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`

Konfigurieren des Anwendungsservers

Konfigurieren von IIS	
Schritt	**Weitere Informationen**
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung Aktivieren der integrierten Windows-Authentifizierung	

Konfigurieren von ASP.NET	
Schritt	**Weitere Informationen**
Konfigurieren der Remotekomponente (in ASP.NET) für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Stammverzeichnis der Remotekomponente. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`

Ändern des ASPNET-Kennworts in einen bekannten Wert	Bei ASPNET handelt es sich um ein lokales Konto mit möglichst geringen Rechten, das standardmäßig für die Ausführung von ASP.NET-Webanwendung verwendet wird (in diesem Fall der Hostprozess der Remotekomponente).
	Legen Sie das Kennwort des Kontos ASPNET mit **Lokale Benutzer und Gruppen** auf einen bekannten Wert fest.
	Bearbeiten Sie **Machine.config** in **%windir%\Microsoft.NET\Framework\v1.0.3705\CONFIG**, und ändern Sie die Konfiguration des **password**-Attributs im <**processModel**>-Element.
	Die Standardeinstellung
	
	`<!-- userName="machine" password="AutoGenerate" -->`
	
	wird geändert in
	
	`<!-- userName="machine" password="YourNewStrongPassword" --> `

Sicherstellen, dass Identitätswechsel deaktiviert sind	Identitätswechsel sind standardmäßig deaktiviert. Überprüfen Sie in **Web.config** folgendermaßen, ob sie nicht doch aktiviert sind: `<identity impersonate="false" />` Der gleiche Effekt kann erzielt werden, indem das <**identity**>-Element entfernt wird.

Konfigurieren von SQL Server

Schritt	Weitere Informationen
Erstellen eines Windows-Kontos auf dem SQL Server-Computer, das dem ASP.NET-Prozesskonto entspricht, mit dem der Webdienst ausgeführt wird	Benutzername und Kennwort müssen mit dem benutzerdefinierten ASP.NET-Konto übereinstimmen. Erteilen Sie dem Konto die folgenden Rechte: - Auf diesen Computer vom Netzwerk aus zugreifen - Lokale Anmeldung verweigern - Anmelden als Stapelverarbeitungsauftrag
Konfigurieren von SQL Server für die Windows-Authentifizierung	
Erstellen eines SQL Server-Benutzernamens für das benutzerdefinierte ASP.NET-Konto	Dadurch wird der Zugriff auf SQL Server gewährt.
Erstellen eines neuen Datenbankbenutzers und Zuordnen des Benutzernamens zum Datenbankbenutzer	Dadurch wird der Zugriff auf die angegebene Datenbank gewährt.
Erstellen einer neuen Datenbankbenutzerrolle und Hinzufügen des Datenbankbenutzers zur Rolle	
Einrichten der Datenbankberechtigungen für die Datenbankbenutzerrolle	Gewähren Sie minimale Rechte.

Konfigurieren der sicheren Kommunikation

Schritt	Weitere Informationen
Konfigurieren der Website auf dem Webserver für SSL	Siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren der Website auf dem Anwendungsserver für SSL	Siehe "Vorgehensweise: Einrichten von SSL auf einem Webserver" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren von IPSec zwischen Anwendungsserver und Datenbankserver	Siehe "Vorgehensweise: Verwenden von IPSec zum Sichern der Kommunikation zwischen zwei Servern" im Abschnitt "Referenz" dieses Handbuchs.

Analyse

- ASP.NET wird auf dem Webserver und dem Anwendungsserver als lokales Konto mit möglichst geringen Rechten ausgeführt, sodass ein mögliches Risiko einer Offenlegung gemindert wird. In beiden Fällen wird das Standardkonto ASPNET verwendet.

  Durch die Verwendung des lokalen Kontos ASPNET (dupliziert auf dem SQL Server-Computer) werden mögliche Sicherheitsrisiken weiter reduziert. Ein dupliziertes Windows-Konto auf dem Datenbankserver ermöglicht die Ausführung der Remotekomponente auf dem Anwendungsserver mit einem ASP.NET-Konto mit möglichst geringen Rechten.

- Die Standardauthentifizierung im Webserver ermöglicht die Verwendung der Anmeldeinformationen des Benutzers von der Webanwendung, um die Windows-Authentifizierungsherausforderung des Anwendungsservers zu beantworten.

  Um die Remotekomponente mit den Anmeldeinformationen des Aufrufers aufzurufen, konfiguriert die Webanwendung den Proxy der Remotekomponente wie im folgenden Codefragment.

```
string pwd = Request.ServerVariables["AUTH_PASSWORD"];
string uid = Request.ServerVariables["AUTH_USER"];
IDictionary channelProperties =
                        ChannelServices.GetChannelSinkProperties(proxy);
NetworkCredential credentials;
credentials = new NetworkCredential(uid, pwd);
ObjRef objectReference = RemotingServices.Marshal(proxy);
Uri objectUri = new Uri(objectReference.URI);
CredentialCache credCache = new CredentialCache();
credCache.Add(objectUri, "Negotiate", credentials);
channelProperties["credentials"] = credCache;
channelProperties["preauthenticate"] = true;
```

  Weitere Informationen zur Übertragung der Sicherheitsanmeldeinformationen an eine Remotekomponente finden Sie in Kapitel 11, ".NET Remoting-Sicherheit".
- In der ASP.NET-Webanwendung sind keine Identitätswechsel aktiviert, da der Remoteproxy unter Verwendung der Anmeldeinformationen des Benutzers, die aus der Standardauthentifizierung stammen, speziell konfiguriert ist. Alle anderen Ressourcen, auf die die Webanwendung zugreift, verwenden den Sicherheitskontext, der durch das ASP.NET-Prozesskonto bereitgestellt wird.
- Durch SSL zwischen dem Benutzer und dem Webserver werden die Daten geschützt, die zu und vom Webserver gesendet werden, sowie die Anmeldeinformationen der Standardauthentifizierung, die während des Authentifizierungsvorgangs unverschlüsselt übertragen werden.
- Die integrierte Windows-Authentifizierung im Anwendungsserver führt .NET-Rollenprüfungen für den ursprünglichen Aufrufer durch. Die Rollen entsprechen Windows-Gruppen.

  Auch ohne Identitätswechsel können rollenbasierte Prüfungen durchgeführt werden.
- Die ASP.NET-Dateiautorisierung prüft bei allen Dateitypen, für die in der IIS-Metabasis eine Zuordnung besteht, die den Dateityp **aspnet_isapi.dll** zuordnet, die ACLs in Bezug auf den Aufrufer. Die Zugriffsprüfungen für statische Dateien (die in IIS keiner ISAPI-Erweiterung zugeordnet sind) werden von IIS durchgeführt.
- Da auf dem Anwendungsserver keine Identitätswechsel aktiviert sind, erfolgt jeder Zugriff auf lokale oder Remoteressourcen durch die Remotekomponente im Sicherheitskontext von ASPNET. Die ACLs müssen entsprechend festgelegt werden.
- Durch die Windows-Authentifizierung gegenüber SQL Server wird das Speichern der Anmeldeinformationen auf dem Anwendungsserver sowie das Senden der Anmeldeinformationen über das Netzwerk an den SQL Server-Computer vermieden.

Fallstricke

- Die Verwendung eines duplizierten Windows-Kontos im Datenbankserver (das mit dem ASP.NET-Prozesskonto übereinstimmt) führt zu einem höheren Verwaltungsaufwand. Die Kennwörter sollten regelmäßig von Hand aktualisiert und synchronisiert werden.
- Da die .NET-rollenbasierte Sicherheit auf der Windows-Gruppenmitgliedschaft basiert, wird bei dieser Lösung vorausgesetzt, dass die Windows-Gruppen entsprechend den Kategorien der Benutzer (mit gleichen Sicherheitsrechten), die auf die Anwendung zugreifen, feinstufig genug eingerichtet sind.

Verwandte Szenarien

Der Webserver verwendet Kerberos, um die Aufrufer zu authentifizieren. Mithilfe der Kerberos-Delegierung wird der Sicherheitskontext des ursprünglichen Aufrufers an die Remotekomponente auf dem Anwendungsserver übermittelt.

Bei dieser Vorgehensweise müssen alle Benutzerkonten für die Delegierung konfiguriert sein. Die Webanwendung muss weiterhin für Identitätswechsel konfiguriert werden und die Standardanmeldeinformationen verwenden, um den Proxy der Remotekomponente zu konfigurieren. Dieses Verfahren wird unter "Übermitteln des ursprünglichen Aufrufers" in Kapitel 11, ".NET Remoting-Sicherheit", ausführlicher behandelt.

Übermitteln des ursprünglichen Aufrufers an die Datenbank

In den oben beschriebenen Szenarien wurde das Modell mit vertrauenswürdigen Subsystemen verwendet. In allen Fällen vertraute die Datenbank dem Anwendungsserver bzw. dem Webserver, dass die Benutzer ordnungsgemäß authentifiziert und autorisiert werden. Obwohl das Modell mit vertrauenswürdigen Subsystemen viele Vorteile bietet, müssen Sie bei einigen Szenarien (möglicherweise aus Gründen einer Überwachung) eventuell das Modell mit Identitätswechsel/Delegierung verwenden und den Sicherheitskontext des ursprünglichen Aufrufers über Computergrenzen hinweg zur Datenbank übertragen.

Zu typischen Gründen für die Notwendigkeit, den ursprünglichen Aufrufer an die Datenbank zu übermitteln, zählen:

- Sie benötigen einen feinstufigen Zugriff in der Datenbank, und die Berechtigungen werden für einzelne Objekte eingeschränkt. Bestimmte Benutzer und Gruppen dürfen einzelne Objekte lesen, während andere einzelne Objekte beschreiben können.

  Dies steht im Widerspruch zur weniger feinstufigen aufgabenorientierten Autorisierung, bei der die Rollenmitgliedschaft die Lese- und Schreibeigenschaften für bestimmte Objekte festlegt.
- Sie sollten die Überwachungsmöglichkeiten der Plattform nutzen, anstatt die Identität zu übertragen und die Überwachung auf Anwendungsebene durchzuführen.

Wenn Sie sich für ein Modell mit Identitätswechsel/Delegierung entscheiden (oder dazu aufgrund von Firmensicherheitsrichtlinien gezwungen sind) und den Kontext des ursprünglichen Aufrufers durch die Ebenen der Anwendung an das Back-End übermitteln, müssen Sie beim Entwurf Delegierung und Netzwerkzugriff berücksichtigen (was bei mehreren Computern nicht trivial ist). Das Pooling gemeinsam genutzter Ressourcen (z. B. Datenbankverbindungen) bildet ebenfalls ein wesentliches Problem und kann die Skalierbarkeit der Anwendung erheblich reduzieren.

In diesem Abschnitt wird gezeigt, wie Sie Identitätswechsel/Delegierung für die beiden gebräuchlichsten Anwendungsszenarien implementieren:

- ASP.NET zu SQL Server
- ASP.NET über Enterprise Services zu SQL Server

Weitere Informationen zu den Modellen mit vertrauenswürdigen Subsystemen und Identitätswechsel/Delegierung und ihren relativen Vorzügen finden Sie in Kapitel 3, "Authentifizierung und Autorisierung".

ASP.NET zu SQL Server

In diesem Szenario werden die Aufrufe an die Datenbank mit dem Sicherheitskontext des ursprünglichen Aufrufers durchgeführt. Zu den in diesem Abschnitt beschriebenen Authentifizierungsoptionen gehören die Standardauthentifizierung und die integrierte Windows-Authentifizierung. Im Abschnitt "ASP.NET über Enterprise Services zu SQL Server" wird ein Szenario mit Kerberos-Delegierung beschrieben.

Verwenden der Standardauthentifizierung im Webserver

Die folgenden Konfigurationseinstellungen für die Standardauthentifizierung ermöglichen die Übermittlung des ursprünglichen Aufrufers bis zur Datenbank.

Tabelle 5.5: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	● Authentifizieren der Benutzer mithilfe der Standardauthentifizierung von IIS. ● Verwenden der Windows-Authentifizierung in ASP.NET. ● Aktivieren von Identitätswechseln in ASP.NET. ● Verwenden der Windows-Authentifizierung für die Kommunikation mit SQL Server.
Autorisierung	● Verwenden von ACL-Prüfungen für den ursprünglichen Aufrufer auf dem Webserver. ● Wenn die ursprünglichen Aufrufer Windows-Gruppen zugeordnet sind (abhängig von den Anforderungen der Anwendung, z. B. Manager, Kassierer usw.), können Sie .NET-Rollenprüfungen für den ursprünglichen Aufrufer durchführen, um den Zugriff auf Methoden einzuschränken.
Sichere Kommunikation	● Die unverschlüsselten Anmeldeinformationen, die zwischen dem Webserver und der Datenbank gesendet werden, werden mit SSL gesichert. ● Alle vertraulichen Daten, die zwischen der Webanwendung und der Datenbank gesendet werden, werden mit IPSec gesichert.

Bei diesem Verfahren müssen Sie die folgenden Punkte beachten:

- Bei der Standardauthentifizierung wird ein Popupdialogfeld anzeigt, in dem der Benutzer zur Eingabe der Anmeldeinformationen (Benutzername und Kennwort) aufgefordert wird.
- Die Datenbank muss den ursprünglichen Aufrufer erkennen. Wenn sich Webserver und Datenbank in verschiedenen Domänen befinden, müssen geeignete Vertrauensbeziehungen bestehen, die eine Authentifizierung des ursprünglichen Aufrufers ermöglichen.

Verwenden der integrierten Windows-Authentifizierung im Webserver

Die integrierte Windows-Authentifizierung führt abhängig von den Konfigurationen der Client- und Servercomputer zu einer NTLM- oder Kerberos-Authentifizierung.

Die NTLM-Authentifizierung unterstützt keine Delegierung. Daher kann der Sicherheitskontext des ursprünglichen Aufrufers nicht vom Webserver an eine physisch entfernte Datenbank übermittelt werden. Der einzige Netzwerkhop, der bei der NTLM-Authentifizierung zulässig ist, wird zwischen Browser und Webserver benötigt. Wenn die NTLM-Authentifizierung verwendet werden soll, muss SQL Server auf dem Webserver installiert werden. Dies ist in der Regel allerdings nur bei sehr kleinen Intranetanwendungen geeignet.

ASP.NET über Enterprise Services zu SQL Server

● In diesem Szenario rufen ASP.NET-Seiten Unternehmenskomponenten auf, die in einer Enterprise Services-Remoteanwendung enthalten sind, die wiederum mit einer Datenbank kommuniziert. Der Sicherheitskontext des ursprünglichen Aufrufers wird vom Browser bis zur Datenbank übertragen (siehe Abbildung 5.9).

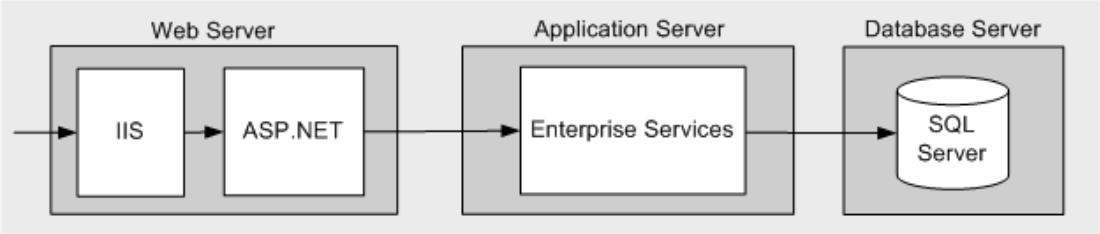

Abbildung 5.9
ASP.NET ruft eine Komponente in Enterprise Services auf, die die Datenbank aufruft

Merkmale

- Die Benutzer verfügen über Internet Explorer 5.x oder höher.
- Auf allen Computern wird Windows 2000 oder höher ausgeführt.
- Die Benutzerkonten befinden sich in Active Directory in einer einzigen Gesamtstruktur.
- Die Anwendung überträgt den Sicherheitskontext des ursprünglichen Aufrufers (auf Betriebssystemebene) bis an die Datenbank.
- In allen Ebenen wird die Windows-Authentifizierung verwendet.
- Die Domänenbenutzerkonten sind für eine Delegierung konfiguriert. Das Konto, mit dem die Enterprise Services-Anwendung ausgeführt wird, muss in Active Directory als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet werden.

Sichern des Szenarios

In diesem Szenario authentifiziert der Webserver den Aufrufer. Sie müssen dann ASP.NET für Identitätswechsel konfigurieren, damit der Sicherheitskontext des ursprünglichen Aufrufers an die Enterprise Services-Remoteanwendung gelangt. In der Enterprise Services-Anwendung muss der Komponentencode explizit die Identität des Aufrufers wechseln (mit **CoImpersonateClient**), damit der Kontext des Aufrufers an die Datenbank übermittelt wird.

Tabelle 5.6: *Sicherheitsmaßnahmen*

Kategorie	Details
Authentifizierung	• Alle Ebenen (Webserver, Anwendungsserver und Datenbankserver) unterstützen die Kerberos-Authentifizierung.
Autorisierung	• Die Autorisierung wird in der mittleren Ebene mit Enterprise Services (COM+)-Rollen in Bezug auf die Identität des ursprünglichen Aufrufers geprüft.
Sichere Kommunikation	• Die vertraulichen Daten werden zwischen Browser und Webserver mit SSL gesichert. • Zwischen ASP.NET und den Serviced Components in der Enterprise Services-Remoteanwendung wird die RPC-Paketsicherheit (mit Verschlüsselung) verwendet. • Zwischen den Serviced Components und der Datenbank wird IPSec verwendet.

Das Ergebnis

Abbildung 5.10 zeigt die empfohlene Sicherheitskonfiguration für dieses Szenario.

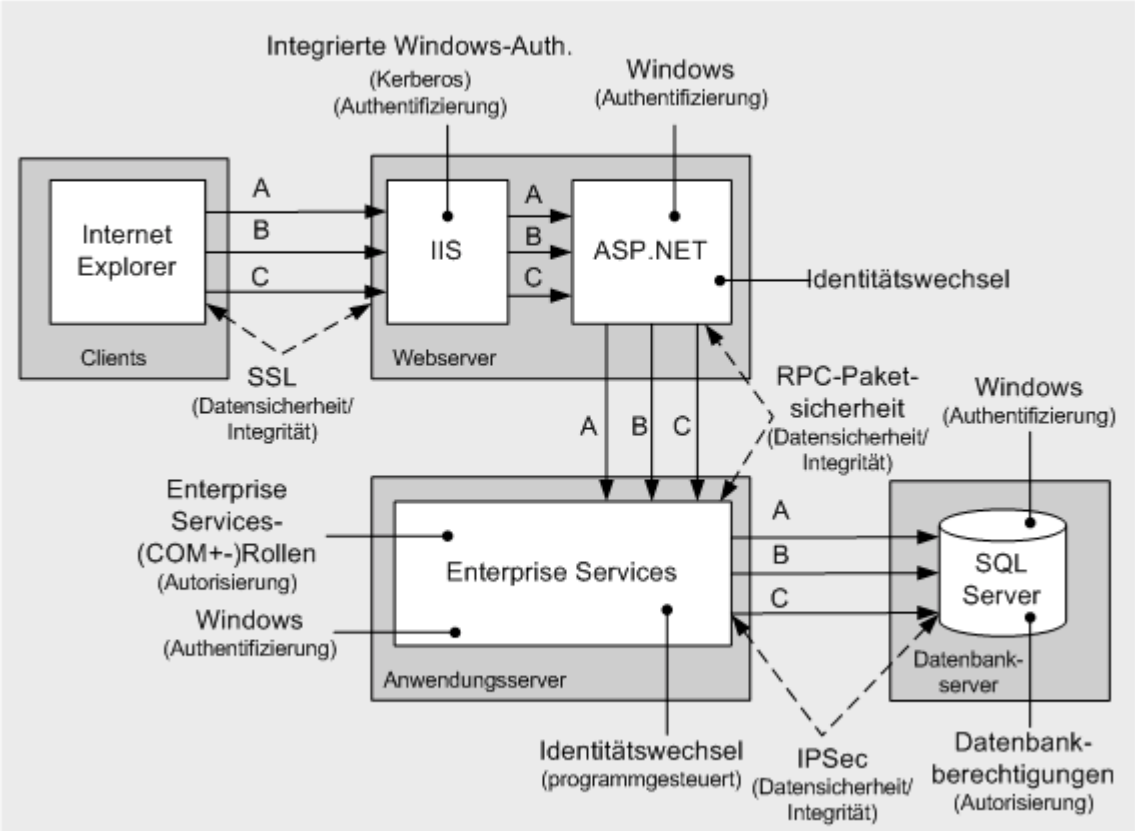

Abbildung 5.10

ASP.NET ruft eine Komponente in Enterprise Services auf, die die Datenbank aufruft. Der Sicherheitskontext des ursprünglichen Aufrufers wird an die Datenbank übertragen.

Konfigurationsschritte für die Sicherheit

Vor Beginn sollten Sie die folgenden Konfigurationsprobleme kennen:

- Das Enterprise Services-Prozesskonto muss in Active Directory als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet sein. Die Benutzerkonten dürfen nicht als **Konto kann nicht delegiert werden** gekennzeichnet sein. Weitere Informationen finden Sie unter "Vorgehensweise: Implementieren der Kerberos-Delegierung unter Windows 2000" im Abschnitt "Referenz" dieses Handbuchs.
- Auf allen Computern ist Windows 2000 oder höher erforderlich. Dies umfasst Clientcomputer (Browser) und alle Server.
- Alle Computer müssen sich in Active Directory befinden und zu einer einzigen Gesamtstruktur gehören.
- Auf dem Anwendungsserver, auf dem sich die Enterprise Services befinden, muss Windows 2000 SP3 ausgeführt werden.
- Wenn Sie Internet Explorer 6.0 unter Windows 2000 verwenden, wird standardmäßig die NTLM-Authentifizierung anstelle der erforderlichen Kerberos-Authentifizierung verwendet. Informationen zum Aktivieren der Kerberos-Delegierung finden Sie in der Microsoft Knowledge Base im Artikel Q299838, "Can't Negotiate Kerberos Authentication After Upgrading to Internet Explorer 6", (US).

Konfigurieren des Webservers (IIS)	
Schritt	**Weitere Informationen**
Deaktivieren des anonymen Zugriffs für das virtuelle Stammverzeichnis der Webanwendung Aktivieren der integrierten Windows-Authentifizierung	

Konfigurieren des Webservers (ASP.NET)	
Schritt	**Weitere Informationen**
Konfigurieren der ASP.NET-Webanwendung für die Windows-Authentifizierung	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis der Webanwendung. Legen Sie das <**authentication**>-Element fest auf: `<authentication mode="Windows" />`
Konfigurieren der ASP.NET-Webanwendung für Identitätswechsel	Bearbeiten Sie **Web.config** im virtuellen Verzeichnis der Webanwendung. Legen Sie das <**identity**>-Element fest auf: `<identity impersonate="true" />`
Konfigurieren der DCOM-Ebene für Identitätswechsel, die von der ASP.NET-Webanwendung für ausgehende Aufrufe verwendet wird	Die ASP.NET-Webanwendung ruft die Remote Serviced Components über DCOM auf. Die Standardebene für Identitätswechsel, die für ausgehende Aufrufe von ASP.NET verwendet wird, ist **Impersonate**. Diese Ebene muss in **Machine.config** in **Delegate** geändert werden. Öffnen Sie **Machine.config**, suchen Sie das <**processModel**>-Element, und legen Sie das **comImpersonateLevel**-Attribut wie folgt auf **Delegate** fest. `<processModel comImpersonationLevel="Delegate"`
Konfigurieren der DCOM-Ebene für die Authentifizierung im Client	Die DCOM-Ebenen für die Authentifizierung werden durch Client und Server bestimmt. DCOM-Client ist in diesem Fall ASP.NET. Öffnen Sie **Machine.config**, suchen Sie das <**processModel**>-Element, und legen Sie das **comAuthenitcationLevel**-Attribut wie folgt auf **PktPrivacy** fest. `<processModel comAuthenticationLevel="PktPrivacy"`

Konfigurieren der Serviced Components (und des Anwendungsservers)	
Schritt	Weitere Informationen
Verwaltete Klassen müssen von der **ServicedComponent**-Klasse abstammen	Siehe Artikel Q306296, "HOW TO: Create a Serviced .NET Component in Visual C# .NET" (US) in der Microsoft Knowledge Base.

Hinzufügen von Code zu der Serviced Component, um die Identität des Aufrufers vor dem Zugriff auf Remoteressourcen (z. B. eine Datenbank) durch Aufrufen der APIs **CoImpersonateClient()** und **CoRevertToSelf()** von **OLE32.DLL** zu wechseln, damit der Kontext des Aufrufers verwendet wird. Standardmäßig wird der Kontext des Enterprise Services-Prozesses für ausgehende Aufrufe verwendet.	Hinzufügen von Verweisen auf **OLE32.DLL**:
	
	`class COMSec`
	`{`
	`[DllImport("OLE32.DLL", CharSet=CharSet.Auto)]`
	`public static extern long CoImpersonateClient();`
	
	`[DllImport("OLE32.DLL", CharSet=CharSet.Auto)]`
	`public static extern long CoRevertToSelf();`
	`}`
	
	Aufrufen dieser externen Funktionen vor dem Aufrufen von Remoteressourcen:
	
	`COMSec.CoImpersonateClient();`
	`COMSec.CoRevertToSelf();`
	
	Weitere Informationen finden Sie in Kapitel 9, "Enterprise Services-Sicherheit".

Konfigurieren der Enterprise Services-Anwendung als Serveranwendung	Diese Konfiguration kann im Tool für Komponentendienste oder durch Einfügen des folgenden .NET-Attributs in die Serviced Component-Assembly erfolgen. `[assembly: ApplicationActivation(ActivationOption.Server)]`

Konfigurieren der Enterprise Services-Anwendung, damit die RPC-Paketsicherheits-authentifizierung verwendet wird (um eine sichere Kommunikation mit Verschlüsselung bereitzustellen)	Fügen Sie das folgende .NET-Attribut zu der Serviced Component-Assembly hinzu.
	
	`[assembly: ApplicationAccessControl(`
	`Authentication =`
	`AuthenticationOption.Privacy)]`

Konfigurieren der Enterprise Services-Anwendung für rollenbasierte Sicherheit auf Komponentenebene	Verwenden Sie das folgende Attribut, um Rollenprüfungen auf Prozess- und Komponentenebene (einschließlich Schnittstellen und Klassen) zu konfigurieren.
	
	`[assembly: ApplicationAccessControl(AccessChecksLevel= AccessChecksLevelOption. ApplicationComponent)]`
	
	Erweitern Sie die Klassen mit dem folgenden Attribut:
	
	`[ComponentAccessControl(true)]`
	
	Weitere Informationen zum Konfigurieren von Rollenprüfungen auf Schnittstellen- und Methodenebene finden Sie unter "Konfigurieren der Sicherheit" in Kapitel 9, "Enterprise Services-Sicherheit".

Erstellen eines benutzerdefinierten Kontos für die Ausführung von Enterprise Services und Kennzeichnen als **Konto wird für Delegierungszwecke vertraut** in Active Directory	Die Enterprise Services-Anwendung muss als Domänenkonto ausgeführt werden, das in Active Directory als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet ist. Weitere Informationen finden Sie in "Vorgehensweise: Implementieren der Kerberos-Delegierung unter Windows 2000" im Abschnitt "Referenz" dieses Handbuchs.
Konfigurieren der Ausführung von Enterprise Services mit dem benutzerdefinierten Konto	Diese Konfiguration muss mit dem Tool für Komponentendienste oder einem Skript durchgeführt werden. In der Serviced Component-Assembly können keine .NET-Attribute verwendet werden.

Konfigurieren des Datenbankservers (SQL Server)	
Schritt	**Weitere Informationen**
Konfigurieren von SQL Server für die Windows-Authentifizierung	
Erstellen der SQL Server-Benutzernamen für die Windows-Gruppen, zu denen die Benutzer gehören	Dadurch wird der Zugriff auf SQL Server gewährt. Die Zugriffssteuerungsrichtlinie behandelt Windows-Gruppen als Rollen, z. B. die Gruppen **Employees**, **HRDept** und **PayrollDept**.
Erstellen neuer Datenbankbenutzer für alle SQL Server-Benutzernamen	Dadurch wird der Zugriff auf die angegebene Datenbank gewährt.
Einrichten der Datenbankberechtigungen für die Datenbankbenutzer	Gewähren Sie minimale Rechte. Weitere Informationen finden Sie in Kapitel 12, "Datenzugriffssicherheit".

Analyse

- Der Schlüssel für die Übermittlung des Sicherheitskontexts des ursprünglichen Aufrufers liegt in der Kerberos-Authentifizierung, die ein Token auf der Ebene **Delegate** erzeugt. Nachdem der Serverprozess (IIS) das Token auf der Ebene **Delegate** empfangen hat, kann es an andere Prozesse übergeben werden, die auf demselben Computer unter einem beliebigen Konto ausgeführt werden, ohne dass die Delegierungsebene geändert wird. Dabei ist es unerheblich, ob der Workerprozess als lokales Konto oder als Domänenkonto ausgeführt wird. Allerdings ist das Konto, unter dem IIS ausgeführt wird, durchaus von Belang. Wenn für die Ausführung ein anderes Konto als **LocalSystem** verwendet wird, muss das Konto in Active Directory als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet werden.

  Wenn IIS als **LocalSystem** ausgeführt wird, muss das Computerkonto als **Konto wird für Delegierungszwecke vertraut** gekennzeichnet werden. Weitere Informationen finden Sie unter "Vorgehensweise: Implementieren der Kerberos-Delegierung unter Windows 2000" im Abschnitt "Referenz" dieses Handbuchs.
- In diesem Szenario ist die integrierte Windows-Authentifizierung (mit Kerberos) ideal, da alle Benutzer über Windows-Konten verfügen und Internet Explorer 5.x oder höher verwenden. Der Vorteil der integrierten Windows-Authentifizierung besteht darin, dass das Kennwort des Benutzers niemals über das Netzwerk gesendet wird. Die Anmeldung ist darüber hinaus transparent, da Windows die Anmeldesitzung des aktuellen interaktiven Benutzers verwendet.
- ASP.NET erstellt ein **WindowsPrincipal**-Objekt, das mit dem Kontext der aktuellen Webanforderung verknüpft wird (**HttpContext.User**). Wenn Sie in der Webanwendung Autorisierungsprüfungen durchführen müssen, können Sie dafür den folgenden Code verwenden.

  ```
  WindowsPrincipal wp = (HttpContext.Current.User as WindowsPrincipal);
  if ( wp.IsInRole("Manager") )
  {
    // User is authorized to perform manager-specific functionality
  }
  ```

  Das **FileAuthorizationModule** von ASP.NET ermöglicht bei ASP.NET-Dateitypen, die in IIS **Aspnet_isapi.dll** zugeordnet sind, ACL-Prüfungen für den ursprünglichen Aufrufer. Bei statischen Dateitypen, wie z. B. JPG-, GIF- und HTM-Dateien, fungiert IIS als Gatekeeper. Die Zugriffsprüfungen werden dann unter Verwendung der Identität des ursprünglichen Aufrufers durchgeführt.
- Durch die Verwendung der Windows-Authentifizierung bei SQL Server vermeiden Sie das Speichern der Anmeldeinformationen in Dateien auf dem Anwendungsserver und ebenfalls deren Übertragung über das Netzwerk. Fügen Sie beispielsweise das **Trusted_Connection**-Attribut in die Verbindungszeichenfolge ein:

  ```
  ConStr="server=YourServer; database=yourdatabase; Trusted_Connection=Yes;”
  ```

- Der Kontext des ursprünglichen Aufrufers wird durch alle Ebenen übertragen, wodurch eine Überwachung erheblich vereinfacht wird. Sie können eine Überwachung auf Plattformebene verwenden (z. B. die Überwachungsfeatures, die von Windows und SQL Server bereitgestellt werden).

Fallstricke

- Wenn Sie Internet Explorer 6.0 unter Windows 2000 verwenden, wird standardmäßig der Authentifizierungsmechanismus NTLM (und nicht Kerberos) verwendet. Weitere Informationen finden Sie in der Microsoft Knowledge Base im Artikel Q299838, "Can't Negotiate Kerberos Authentication After Upgrading to Internet Explorer 6" (US).
- Das Delegieren von Benutzern über die Ebenen hinweg beeinträchtigt im Vergleich zum Modell mit vertrauenswürdigen Subsystemen die Systemleistung und die Skalierbarkeit der Anwendung. Sie können nicht die Vorteile eines Verbindungspoolings zur Datenbank nutzen, da die Verbindungen zu der Datenbank an den Sicherheitskontext des ursprünglichen Aufrufers gebunden sind und daher nicht effizient gepoolt werden können.
- Diese Vorgehensweise hängt auch davon ab, wie feinstufig die Windows-Gruppen die Sicherheitsanforderungen der Anwendung erfüllen. Die Windows-Gruppen müssen daher entsprechend feinstufig eingerichtet werden, um die Benutzerkategorien (Benutzer mit den gleichen Sicherheitsrechten) abzubilden, die auf die Anwendung zugreifen.

Zusammenfassung

In diesem Kapitel wurde beschrieben, wie häufige Intranetanwendungsszenarien gesichert werden.

Informationen für Extranet- und Internetanwendungsszenarien finden Sie in Kapitel 6, "Extranetsicherheit", und Kapitel 7, "Internetsicherheit".