# Schlüssel und Zertifikate

Bei der asymmetrischen Verschlüsselung wird ein Schlüsselpaar bestehend aus einem öffentlichen und einem privaten Schlüssel verwendet. Daten, die mit dem privaten Schlüssel verschlüsselt wurden, können nur mit dem zugehörigen öffentlichen Schlüssel wieder entschlüsselt werden und umgekehrt.

Öffentliche Schlüssel werden (wie der Name schon andeutet) öffentlich zugänglich gemacht. Im Gegensatz dazu befindet sich der private Schlüssel im alleinigen Besitz der jeweiligen Einzelperson. Als Verteilungsmechanismus zur Übermittlung öffentlicher Schlüssel an die Benutzer werden Zertifikate verwendet. Zertifikate werden in der Regel von einer Zertifizierungsstelle (Certification Authority oder CA) signiert, wodurch bestätigt wird, dass der öffentliche Schlüssel auch von der Person stammt, die angibt, den öffentlichen Schlüssel gesendet zu haben. Die Zertifizierungsstelle ist für beide Seiten vertrauenswürdig.

Die Standardimplementierung einer digitalen Zertifizierung umfasst den Prozess des Signierens des Zertifikats. Dieses Verfahren ist in Abbildung 1 dargestellt.

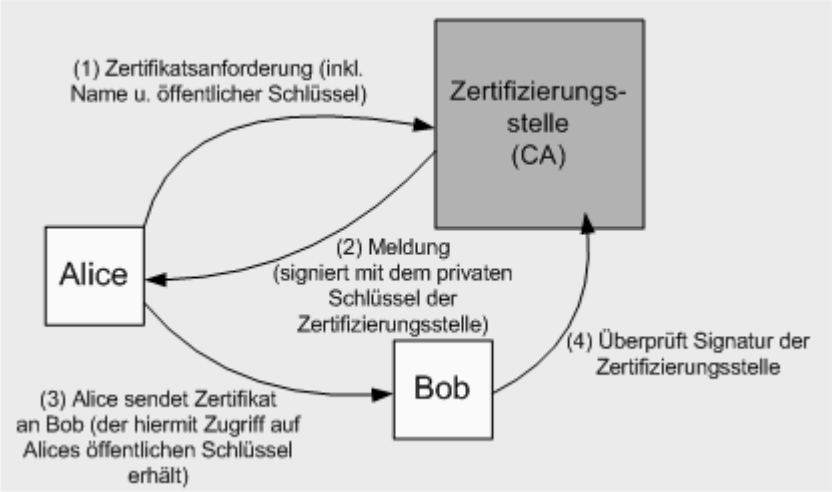


Abbildung 1
*Verfahren bei der digitalen Zertifizierung*

Die in Abbildung 1 dargestellten Ereignisse laufen nach folgendem Schema ab:

1. Alice sendet eine Anforderung betreffend eines signierten Zertifikats an eine Zertifizierungsstelle, die ihren Namen, ihren öffentlichen Schlüssel und ggf. einige Zusatzinformationen umfasst.
2. Die Zertifizierungsstelle erzeugt aus der von Alice übermittelten Anforderung eine Nachricht. Die Zertifizierungsstelle signiert die Nachricht mit ihrem privaten Schlüssel und erstellt auf diese Weise eine separate Signatur. Die Nachricht und die Signatur werden von der Zertifizierungsstelle an Alice zurückgesendet. Diese Kombination aus Nachricht und Signatur ergibt nun Alices Zertifikat.
3. Alice sendet ihr Zertifikat an Bob, um ihm den Zugriff auf ihren öffentlichen Schlüssel zu ermöglichen.
4. Bob prüft die Signatur des Zertifikats unter Verwendung des öffentlichen Schlüssels der Zertifizierungsstelle. Wenn die Signatur gültig ist, akzeptiert er den öffentlichen Schlüssel im Zertifikat als Alices öffentlichen Schlüssel.

Wie bei jeder digitalen Signatur kann jeder Empfänger mit Zugriff auf den öffentlichen Schlüssel der Zertifizierungsstelle feststellen, ob das Zertifikat von der jeweiligen Zertifizierungsstelle signiert wurde. Bei diesem Verfahren ist kein Zugriff auf irgendwelche geheimen Informationen erforderlich. Beim vorstehenden Szenario wird davon ausgegangen, dass Bob Zugriff auf den öffentlichen Schlüssel der Zertifizierungsstelle hat. Der Zugriff auf diesen Schlüssel ist für Bob möglich, wenn er über eine Kopie des Zertifikats der Zertifizierungsstelle verfügt, die diesen öffentlichen Schlüssel enthält.

## Digitale X.509-Zertifikate

Digitale X.509-Zertifikate enthalten nicht nur den Namen und den öffentlichen Schlüssel des Benutzers, sondern auch weitere, den Benutzer betreffende Informationen. Diese Zertifikate sind mehr als nur Trittsteine in einer digitalen Vertrauenshierarchie. Hiermit ist die Zertifizierungsstelle in der Lage, für den Empfänger eines Zertifikats eine Vertrauensbasis zu schaffen, die sich nicht nur auf den öffentlichen Schlüssel des Übermittlers des Zertifikats erstreckt, sondern auch für die anderen hierin enthaltenen Informationen gilt. Diese anderen Informationen können u. a. die E-Mail-Adresse, das Recht zum Unterzeichnen von Dokumenten eines gegebenen Werts oder das Recht, als selbständige Zertifizierungsstelle andere Zertifikate zu unterzeichnen, umfassen.

X.509-Zertifikate und viele andere Zertifikate verfügen über eine bestimmte Gültigkeitsdauer. Der Gültigkeitszeitraum eines Zertifikats kann also ablaufen, sodass das Zertifikat dann nicht mehr gültig ist. Eine Zertifizierungsstelle kann ein Zertifikat auch aus unterschiedlichen Gründen sperren. Für den Umgang mit solchen Sperrungen unterhält und veröffentlicht die Zertifizierungsstelle eine Liste der gesperrten Zertifikate, die als Zertifikatsperrliste (Certificate Revocation List, CRL) bezeichnet wird. Netzwerkbenutzer greifen auf die Zertifikatsperrliste zu, um die Gültigkeit eines Zertifikats zu prüfen.

## Zertifikatspeicher

Zertifikate werden an sicheren Speicherorten abgelegt, die als *Zertifikatspeicher* bezeichnet werden. Ein Zertifikatspeicher kann Zertifikate, Zertifikatsperrlisten und Zertifikatvertrauenslisten (Certificate Trust Lists, CTLs) umfassen. Jeder Benutzer verfügt über einen persönlichen Speicher (als eigener Speicher oder "MY store" bezeichnet), in dem die Zertifikate des jeweiligen Benutzers gespeichert sind. Der eigene Speicher kann physikalisch an einer Reihe von Speicherorten implementiert sein, wozu die Registrierung ebenso gehört wie ein lokaler oder ein Remotecomputer, eine Datenträgerdatei, eine Datenbank, ein Verzeichnisdienst, ein Smartgerät oder ein anderer Speicherort.

Im eigenen Speicher können zwar beliebige Zertifikate gespeichert werden, er sollte jedoch den persönlichen Zertifikaten des Benutzers vorbehalten sein, d. h. den Zertifikaten, die zum Signieren und Verschlüsseln der Nachrichten dieses speziellen Benutzers verwendet werden.

Neben dem eigenen Speicher bietet Windows noch die folgenden Zertifikatspeicher:

- **CA und ROOT**. Dieser Speicher enthält die Zertifikate von Zertifizierungsstellen, denen der Benutzer vertraut, sodass diese Zertifikate für andere ausstellen können. Einige Zertifikate vertrauenswürdiger Zertifizierungsstellen (CA) gehört zum Lieferumfang des Betriebssystems, andere können von Administratoren hinzugefügt werden.
- **Andere**. Dieser Speicher enthält die Zertifikate anderer Personen, mit denen der Benutzer signierte Nachrichten austauscht.

Die CryptoAPI stellt Funktionen zum Verwalten von Zertifikaten bereit. Auf diese APIs kann nur über unverwalteten Code zugegriffen werden. Auch CAPICOM ist eine COM-basierte API für die CryptoAPI, auf die über COM Interop zugegriffen werden kann.

## Weitere Informationen

Weitere Informationen finden Sie in MSDN unter "Cryptography, CryptoAPI, and CAPICOM" (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/security/Security/cryptography_cryptoapi_and_capicom.asp, englischsprachig).

# Kryptografie

Kryptografie wird verwendet, um Folgendes bereitzustellen:

- **Vertraulichkeit**. Hiermit soll sichergestellt werden, dass die Vertraulichkeit von Daten gewahrt bleibt. Vertraulichkeit wird in der Regel unter Verwendung von Verschlüsselung erreicht. Verschlüsselungsalgorithmen (die Verschlüsselungsschlüssel verwenden) werden herangezogen, um Klartext in verschlüsselten Text umzuwandeln, und mit dem zugehörigen Entschlüsselungsschlüssel kann der verschlüsselte Text wieder in Klartext umgewandelt werden. Bei symmetrischen Verschlüsselungsalgorithmen wird für Ver- und Entschlüsselung der gleiche Schlüssel verwendet, bei asymmetrischen Algorithmen wird ein aus öffentlichem und privatem Schlüssel bestehendes Schlüsselpaar verwendet.
- **Datenintegrität**. Hiermit wird sichergestellt, dass Daten vor zufälliger oder vorsätzlicher (böswilliger) Änderung geschützt sind. Die Integrität wird normalerweise durch Nachrichtenauthentifizierungscodes (Message Authentication Codes, MACs) oder Hashs bereitgestellt. Ein Hashwert ist ein numerischer Wert mit einer festen Länge, der aus einer Datenfolge abgeleitet wird. Hashwerte werden verwendet, um die Integrität von Daten zu prüfen, die über unsichere Kanäle gesendet wurden. Hierbei wird der Hashwert der empfangenen Daten mit dem Hashwert der Daten beim Versand verglichen, um festzustellen, ob die Daten verändert wurden.
- **Authentifizierung**. Hiermit wird sichergestellt, dass Daten von einem bestimmten Sender stammen. Für die Authentifizierung werden digitale Zertifikate verwendet. Hierbei werden in der Regel die Hashwerte digital signiert, da diese wesentlich kleiner als die Quelldaten sind, die sie repräsentieren.

## Technische Wahlmöglichkeiten

- Verwenden Sie einen Hashwert, wenn Sie eine Möglichkeit zur Überprüfung benötigen, dass Daten auf dem Übertragungsweg nicht verfälscht wurden.
- Verwenden Sie einen verschlüsselten Hashwert, wenn Sie sicherstellen möchten, dass eine Entität den geheimen Schlüssel kennt, ohne dass dieser hin- und her gesendet werden muss, oder wenn Sie die Daten unter Verwendung eines einfachen Hashs vor einem Abfangen auf dem Übertragungsweg schützen möchten.
- Verwenden Sie Verschlüsselung, wenn Sie Daten bei der Übertragung über ein unsicheres Medium unlesbar machen oder wenn Sie Daten dauerhaft speichern möchten.
- Verwenden Sie ein Zertifikat, wenn Sie die Person überprüfen möchten, die angibt, der Eigentümer des öffentlichen Schlüssels zu sein.
- Verwenden Sie die symmetrische Verschlüsselung aus Geschwindigkeitsgründen und wenn beide Parteien den Schlüssel im Vorfeld ausgetauscht haben.
- Verwenden Sie die asymmetrische Verschlüsselung, wenn Sie Daten sicher über ein unsicheres Medium austauschen möchten.
- Verwenden Sie eine digitale Signatur, wenn Sie Authentifizierung und Nachweisbarkeit wünschen.
- Verwenden Sie einen Salt-Wert (eine kryptografisch erzeugte Zufallszahl), um sich gegen Verzeichnisangriffe (Wörterbuchangriffe) zu schützen.

## Kryptografie in .NET

Der **System.Security.Cryptography**-Namespace stellt Verschlüsselungsdienste einschließlich der sicheren Ver- und Entschlüsselung von Daten, Hashalgorithmen, Erzeugung von Zufallszahlen und Nachrichtenauthentifizierung bereit.

Das .NET Framework bietet Implementierungen einer Vielzahl von standardmäßigen Verschlüsselungsalgorithmen, die aufgrund der gut definierten Vererbungshierarchie bestehend aus abstrakten Klassen, die die Arten der Basisalgorithmen definieren, einfach erweitert werden können. Diese Basisalgorithmen umfassen symmetrische, asymmetrische und Hashalgorithmen sowie Algorithmusklassen.

Tabelle 1: Diese Tabelle enthält die Algorithmen, für die das .NET Framework "einsatzfertige" Implementierungsklassen bietet.

| Symmetrische Algorithmen | Asymmetrische Algorithmen | Hashalgorithmen |
|---|---|---|
| DES (Data Encryption Standard) | DSA (Digital Signature Algorithm) | HMAC SHA1 (hashbasierter Nachrichtenauthentifizierungscode, der den Hashalgorithmus SHA1 verwendet) |
| Dreifach-DES (TripleDES, Triple Data Encryption Standard) | RSA | MAC Triple DES (Nachrichtenauthentifizierungscode, der Dreifach-DES verwendet) |
| Rijndael | | MD5 |
| RC2 | | SHA1, SHA256, SHA384, SHA512 (sicherer Hashalgorithmus, der unterschiedliche Hashgrößen verwendet) |

## Unterstützung für symmetrische Algorithmen

.NET umfasst die folgenden Implementierungsklassen, die symmetrische Verschlüsselungsalgorithmen mit geheimen Schlüsseln bereitstellen:

- DESCryptoServiceProvider
- RC2CryptoServiceProvider
- RijndaelManaged
- TripleDESCryptoServiceProvider

---

**Hinweis:** Klassen, die auf "CryptoServiceProvider" enden, sind Wrapperklassen, die die zugrunde liegenden Dienste des Kryptografiedienstanbieters (Cryptographic Service Provider, CSP) nutzen; und Klassen, die auf "Managed" enden, werden mit verwaltetem Code implementiert.

---

Abbildung 2 zeigt die Vererbungshierarchie, die von .NET Framework verwendet wird. Die Algorithmustyp-Basisklasse (z. B. **SymmetricAlgorithm**) ist abstrakt. Von der abstrakten Typbasisklasse wird eine Reihe abstrakter Algorithmusklassen abgeleitet. Algorithmusimplementierungsklassen stellen konkrete Implementierungen des ausgewählten Algorithmus bereit; Beispiele hierfür sind DES, Dreifach-DES (Triple DES), Rijndael und RC2.

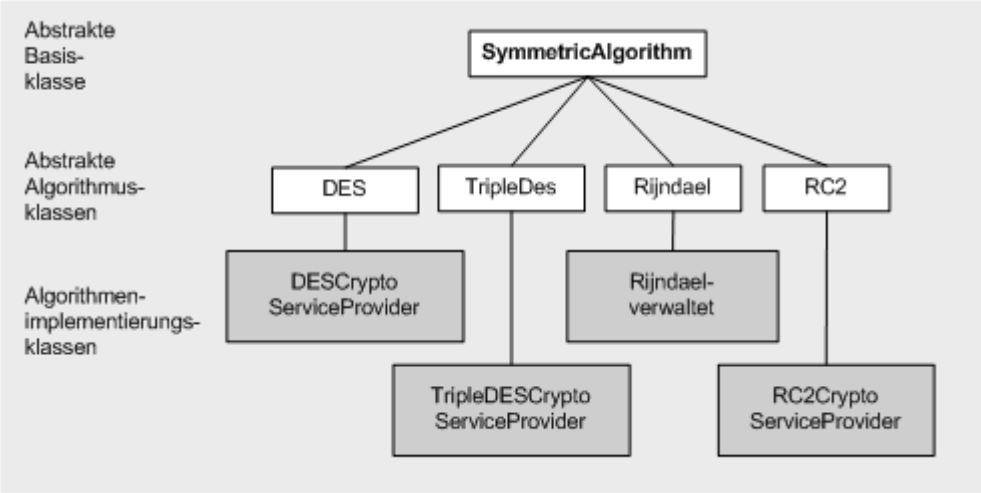


Abbildung 2
*Die Vererbungshierarchie der symmetrischen Verschlüsselungsklasse*

## Unterstützung für asymmetrische Algorithmen

.NET stellt die folgenden asymmetrischen Verschlüsselungsalgorithmen (mit öffentlichem und privatem Schlüssel) über die abstrakte Basisklasse (**System.Security.Cryptography.AsymmetricAlgorithm**) bereit:

- DSACryptoServiceProvider
- RSACryptoServiceProvider

Diese werden verwendet, um Daten digital zu signieren und zu verschlüsseln. Abbildung 3 zeigt die Vererbungshierarchie.

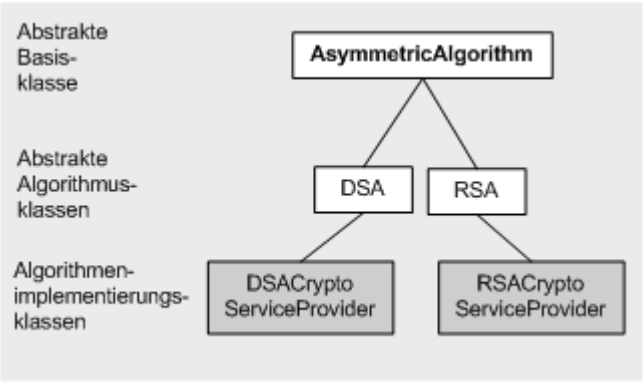


Abbildung 3
*Die Vererbungshierarchie der asymmetrischen Verschlüsselungsklasse*

## Unterstützung für Hashalgorithmen

.NET stellt die folgenden Hashalgorithmen bereit:

- SHA1, SHA256, SHA384, SHA512
- MD5
- HMACSHA (verschlüsselter Hashalgorithmus)
- MACTripleDES (verschlüsselter Hashalgorithmus)

Abbildung 4 zeigt die Vererbungshierarchie für die Hashalgorithmusklassen.

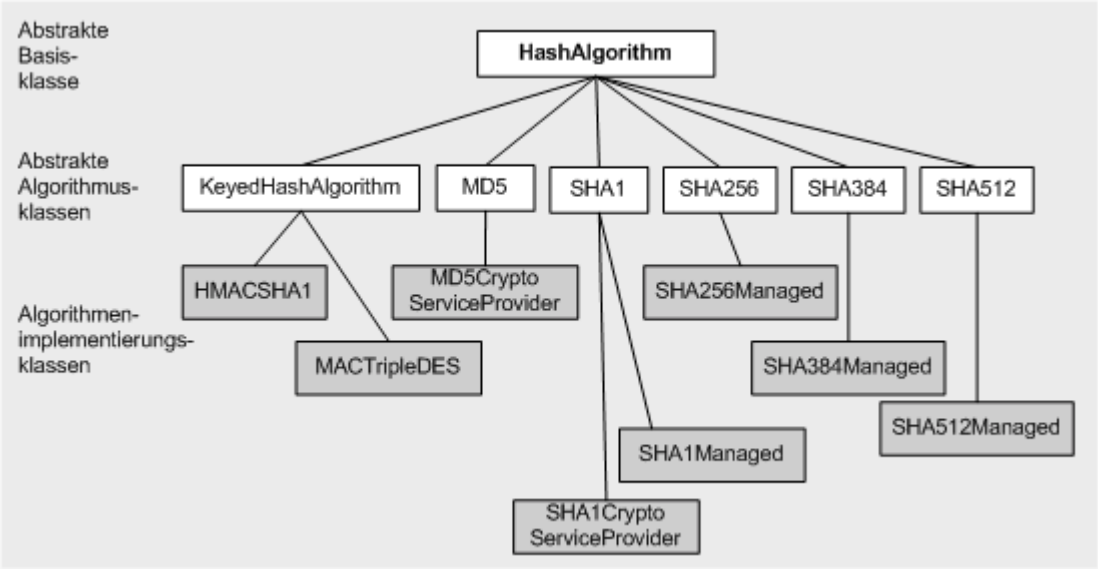


Abbildung 4
*Die Vererbungshierarchie der Hashalgorithmusklasse*

# Zusammenfassung

Die Kryptografie ist eine wichtige Technologie für das Erstellen sicherer Webanwendungen. In diesem Anhang wurden einige grundlegende Kenntnisse über Zertifikate und Verschlüsselung vermittelt. Darüber hinaus wurden einige der Klassen vorgestellt, die vom **System.Security.Cryptography**-Namespace bereitgestellt werden und mit denen Sie in der Lage sind, auf Kryptografie basierende Sicherheitslösungen einfacher in Ihre .NET-Anwendungen einzubinden.

Weitere Informationen über Kryptografie in .NET finden Sie in MSDN auf der Seite mit dem Titel ".NET Framework Cryptography Model".