Digitale Signaturen

10

Digitale Signaturen sind eines der wichtigsten kryptografischen Werkzeuge, die heutzutage zum Einsatz kommen. Sie haben u. a. Anwendungen bei digitalen Zertifikaten, um Webbrowser abzusichern, beim rechtlich bindenden Signieren digitaler Verträge oder bei der sicheren Aktualisierung von Software. Neben dem Austausch von Schlüsseln über unsichere Kanäle bilden digitale Signaturen die wichtigste Anwendung von asymmetrischer Kryptografie.

Digitale Signaturen sind in Grenzen mit konventionellen Signaturen auf Papier vergleichbar. Mit ihnen kann sichergestellt werden, dass eine Nachricht tatsächlich von der Person stammt, die angibt, sie versendet zu haben. Darüber hinaus bieten sie aber noch weitere Möglichkeiten, die in diesem Kapitel eingeführt werden.

In diesem Kapitel erlernen Sie

- das Grundprinzip digitaler Signaturen,
- Sicherheitsdienste, d. h. Schutzziele, die mit einem Sicherheitssystem erreicht werden können,
- digitale Signaturen mit RSA,
- digitale Signaturen nach Elgamal sowie zwei wichtige Varianten, den Digital Signature Algorithm (DSA) und den Elliptic Curve Digital Signature Algorithm (ECDSA)

10.1 Einführung

In diesem Abschnitt erläutern wir zunächst, warum digitale Signaturen gebraucht werden und warum sie auf asymmetrischer Kryptografie basieren müssen. Dann zeigen wir das Prinzip der digitalen Signatur. Praktische Signaturalgorithmen werden in den nachfolgenden Abschnitten eingeführt.

C. Paar, J. Pelzl, *Kryptografie verständlich*, eXamen.press,
DOI 10.1007/978-3-662-49297-0_10

10.1.1 Autos in ungewöhnlichen Farben oder warum symmetrische Kryptografie allein nicht ausreicht

Die Kryptoverfahren, denen wir bisher begegnet sind, hatten zwei Zielsetzungen: entweder die eigentliche Verschlüsselung von Daten (z. B. mit AES, 3DES oder RSA) oder die Berechnung eines gemeinsamen Geheimnisses über einen unsicheren Kanal (z. B. DHKE oder ECDH). Nun könnte man versucht sein anzunehmen, mit diesen Techniken seien alle praktischen Sicherheitsprobleme zu lösen. Es gibt allerdings eine ganze Reihe weiterer Problemstellungen neben Verschlüsselung und Schlüsselaustausch, die auch Sicherheitsdienste genannt werden und in Abschn. 10.1.3 besprochen werden. Wir stellen nun ein Szenario vor, für das es mit symmetrischer Kryptografie keine zufriedenstellende Lösung gibt.

Betrachten wir wieder unsere beiden Teilnehmer Alice und Bob, die einen gemeinsamen geheimen Schlüssel besitzen. Mit diesem Schlüssel können Daten mithilfe einer Blockchiffre verschlüsselt werden. Wenn Alice eine Nachricht erhält, die semantisch korrekt ist (beispielsweise ist der entschlüsselte Text ein korrekter deutscher Satz), kann sie in vielen Fällen davon ausgehen, dass der Klartext von einer Person erzeugt wurde, mit der sie auch den geheimen Schlüssel teilt[1]. Wenn nur Alice und Bob im Besitz des Schlüssels sind, können sie einigermaßen sicher seien, dass kein Angreifer die Nachricht während der Übermittlung manipuliert hat. Bisher sind wir immer davon ausgegangen, dass der Angreifer eine dritte Partei darstellt (z. B. Oskar). In der Praxis tritt allerdings oft die Situation auf, dass Alice und Bob zwar sicher miteinander kommunizieren möchten, gleichzeitig aber auch ein Interesse daran haben könnten, sich unehrlich zu verhalten. Allgemein können wir sagen, dass Protokolle basierend auf symmetrischer Kryptografie keinen Schutz bieten, wenn die zwei Teilnehmer sich *gegenseitig* angreifen. Wir betrachten das folgende Szenario:

Alice hat ein Autohaus, in dem man neue Fahrzeuge über das Internet auswählen und konfigurieren kann. Der Kunde Bob und die Händlerin Alice haben einen gemeinsamen geheimen Schlüssel k_{AB} ausgehandelt, beispielsweise unter Verwendung des Diffie-Hellman-Protokolls. Bob konfiguriert nun ein Auto genau nach seinen Wünschen. Hierzu gehören leider auch die Farbe Rosa als Außenlackierung und Orange für die Innenausstattung – eine Farbkombination, die vermutlich nur wenige Kunden wählen würden. Bob verschlüsselt das Bestellformular mit AES und sendet es an Alice. Alice entschlüsselt das Chiffrat und freut sich, dass sie ein weiteres Fahrzeug für 25.000 € verkauft hat. Als das Fahrzeug drei Wochen später ausgeliefert wird, kommen Bob allerdings Zweifel an seiner Farbwahl, u. a. weil seine Gattin mit Scheidung droht, nachdem sie das Auto *gesehen* hat. Für Bob (und seine Familie) ist die Situation besonders unangenehm, doch eine Rücknahme von Online-Bestellungen ist ausgeschlossen. Da Alice eine erfahrene Autohändlerin ist, weiß sie, dass es extrem schwierig sein würde, ein rosa-oranges Auto zu

[1] Man sollte mit dieser Schlussfolgerung im allgemeinen Fall allerdings vorsichtig sein. Wenn Alice und Bob beispielsweise eine Stromchiffre benutzen, kann ein Angreifer den Wert individueller Bits auf dem Kanal ändern. Dies führt zu geänderten Bits im Klartext. In manchen Anwendungen kann ein Angreifer den Klartext nun so ändern, dass er semantisch immer noch korrekt ist.

verkaufen. Sie ist daher nicht geneigt, in diesem Fall eine Ausnahme zu machen. Da Bob nun behauptet, er hätte das Auto nie bestellt, bleibt ihr keine andere Wahl, als ihn zu verklagen. Vor Gericht legt Alice' Rechtsanwalt die Online-Bestellung von Bob zusammen mit der verschlüsselten Version der Bestellung vor. Der Rechtsanwalt argumentiert, dass Bob die Bestellung und die verschlüsselte Version erzeugt haben muss, da er in Besitz des Schlüssels k_{AB} war. Der erfahrene Anwalt von Bob hingegen wird dem Gericht gegenüber erklären, dass auch Alice in Besitz des Schlüssels war und dass sie als Autohändlerin einen hohen Anreiz hat, gefälschte Bestellungen zu generieren. Es stellt sich nun heraus, dass es für den Richter unmöglich ist zu entscheiden, wer das Klartext-Chiffrat-Paar erzeugt hat – Bob oder Alice? Nach der Gesetzgebung in den meisten Ländern würde Bob wahrscheinlich freigesprochen.

Obwohl sich die oben geschilderte Situation wie ein sehr spezifisches Beispiel anhört, handelt es sich dabei tatsächlich um ein allgemeines Problem in der Datensicherheit. Es tritt sehr häufig der Fall auf, dass gegenüber einer neutralen dritten Instanz (hier in der Rolle des Richters) bewiesen werden muss, dass eine von zwei Parteien eine bestimmte Nachricht generiert hat. Unter *beweisen* versteht man, dass der Richter zweifelsfrei feststellen kann, von wem die Nachricht stammt, auch wenn alle Teilnehmer sich potenziell unehrlich verhalten. Warum kann man dieses Ziel nicht mit (komplexen) Verfahren basierend auf symmetrischer Kryptografie erreichen? Der Grund hierfür ist recht einfach: Da symmetrische Kryptografie eingesetzt wird, haben Alice und Bob das gleiche Wissen (in Form der symmetrischen Schlüssel) und können daher die gleichen Aktionen ausführen. Alle Berechnungen, die Alice durchführen kann, können auch von Bob ausgeführt werden. Deshalb kann eine neutrale dritte Partei nicht unterscheiden, ob eine bestimmte kryptografische Berechnung von Alice oder von Bob durchgeführt wurde. Die allgemeine Lösung zu diesem Problem liegt in der asymmetrischen Kryptografie. Durch die inhärente Asymmetrie solcher Protokolle kann ein Richter zwischen Aktionen unterscheiden, die nur von einer Person durchgeführt werden können (nämlich von dem Teilnehmer, der den privaten Schlüssel besitzt) und solchen, die beiden Teilnehmern möglich sind (nämlich Berechnungen, die mit dem öffentlichen Schlüssel ausgeführt werden). Digitale Signaturen sind asymmetrische Verfahren, die die Eigenschaft haben, das Problem unehrlicher Teilnehmer zu lösen. In dem oben stehenden Beispiel des Autokaufs müsste Bob seine Bestellung mit seinem privaten Schlüssel signieren. Diese Aktion kann nur von ihm durchgeführt werden, da er der Einzige in Besitz seines privaten Schlüssels ist.

10.1.2 Das Prinzip digitaler Signaturen

Es ist natürlich nicht nur in der digitalen Welt notwendig, dass bewiesen werden muss, dass eine Nachricht oder ein Schriftstück von einer bestimmten Person stammt. Hierfür werden in der analogen Welt i .d. R. Unterschriften auf Papier verwendet. Wenn ein Vertrag oder eine Banküberweisung mit einer Unterschrift versehen ist, kann der Empfänger vor Gericht nachweisen, von wem das Schriftstück stammt. (Man kann natürlich versuchen, die Unterschrift zu fälschen, aber die juristischen und sozialen Barrieren sind so

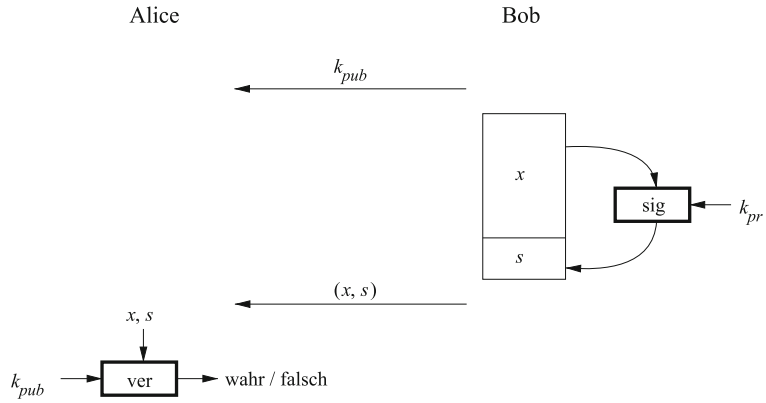

Abb. 10.1 Prinzip digitaler Signaturen, bestehend aus Signatur und Verifikation einer Nachricht

hoch, dass dies nur selten geschieht.) Wie bei konventionellen Unterschriften soll auch bei digitalen Dokumenten nur der Absender in der Lage sein, eine gültige Signatur zu erzeugen. Hierfür werden asymmetrische Verfahren benötigt. Die Grundidee ist hierbei, dass der Absender der Nachricht den privaten Schlüssel verwendet und der Empfänger den dazugehörenden öffentlichen Schlüssel. Das Prinzip digitaler Signaturen ist in Abb. 10.1 dargestellt.

Der Prozess startet, indem Bob die Nachricht x signiert. Der hierfür verwendete Signaturalgorithmus benötigt Bobs privaten Schlüssel k_{pr} als Eingabe. Solange gewährleistet ist, dass sein privater Schlüssel geheim gehalten wird, kann nur Bob Nachrichten mit seiner Unterschrift versehen. Um einen Zusammenhang zwischen Signatur und der Nachricht herzustellen, muss auch die Nachricht x als Eingabe für den Signaturalgorithmus dienen. Nachdem die Nachricht signiert wurde, wird die Signatur s an die Nachricht x angehängt, und das Paar (x, s) wird an Alice gesandt. Es sollte an dieser Stelle betont werden, dass die digitale Signatur selbst nutzlos ist, wenn sie ohne die dazugehörende Nachricht versandt wird. Eine digitale Signatur ohne Nachricht wäre das Gleiche wie eine konventionelle Signatur auf einem dünnen Papierstreifen ohne den dazugehörenden Vertrag oder die Überweisung, die damit unterschrieben werden soll.

Die digitale Signatur ist lediglich eine (große) natürliche Zahl, die z. B. 2048 Bit lang ist. Damit die Signatur für Alice von Wert ist, muss sie eine Möglichkeit haben, die Gültigkeit der Signatur zu *verifizieren*. Hierzu benötigt sie einen Verifikationsalgorithmus, der sowohl die Nachricht x als auch die Signatur s als Eingangswerte bekommt. Um den Zusammenhang zu Bob herzustellen, benötigt die Funktion auch dessen öffentlichen Schlüssel. Obwohl die Verifikationsfunktion sehr lange Eingangswerte hat, ist der Ausgang nur die binäre Aussage *wahr* oder *falsch*. Wenn x tatsächlich mit dem privaten Schlüssel signiert wurde, der zu dem öffentlichen Verifikationsschlüssel gehört, ist die Ausgabe wahr, anderenfalls falsch.

Aus diesen allgemeinen Beobachtungen kann man leicht das folgende generische Protokoll für digitale Signaturen entwickeln: