(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 696 36 815 T2** 2007.11.08

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 0 873 617 B1**
(21) Deutsches Aktenzeichen: **696 36 815.3**
(86) PCT-Aktenzeichen: **PCT/US96/16608**
(96) Europäisches Aktenzeichen: **96 944 186.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 1998/018234**
(86) PCT-Anmeldetag: **18.10.1996**
(87) Veröffentlichungstag der PCT-Anmeldung: **30.04.1998**
(97) Erstveröffentlichung durch das EPA: **28.10.1998**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **03.01.2007**
(47) Veröffentlichungstag im Patentblatt: **08.11.2007**

(51) Int Cl.[8]: ***H04L 9/08*** (2006.01)
H04L 9/32 (2006.01)

(73) Patentinhaber:
Certicom Corp., Mississauga, Ontario, CA

(74) Vertreter:
KRAMER - BARSKE - SCHMIDTCHEN, 81245 München

(84) Benannte Vertragsstaaten:
CH, DE, FR, GB, LI

(72) Erfinder:
VANSTONE, A., Scott, Waterloo, Ontario N2T 2H4, CA; MENEZES, John, Alfred, Auburn, AL 36830, US; QU, Mingua, Waterloo, Ontario N2L 3E5, CA

(54) Bezeichnung: **VERFAHREN ZUR SITZUNGSSCHLÜSSELERZEUGUNG MIT IMPLIZITEN UNTERSCHRIFTEN**

Beschreibung

[0001] Die vorliegende Erfindung betrifft Protokolle zum Vereinbaren von Schlüsseln zum Übertragen und Authentisieren von Schlüsseln zum Verschlüsseln.

[0002] Um während eines Informationsaustauschs eine Geheimhaltung aufrechtzuerhalten, ist es weithin bekannt, Daten unter Verwendung eines Schlüssels zu verschlüsseln. Der Schlüssel muss so gewählt sein, dass die Korrespondenten dazu in der Lage sind, Nachrichten zu verschlüsseln und zu entschlüsseln, jedoch so, dass eine Person, die die Nachricht abfängt, den Inhalt der Nachricht nicht ermitteln kann.

[0003] In einem geheimen Verschlüsselungsprotokoll mit geheimem Schlüssel teilen sich die Korrespondenten einen gemeinsamen Schlüssel, der für die Korrespondenten geheim ist. Dies erfordert, dass der Schlüssel zwischen den Korrespondenten vereinbart wird, und macht es erforderlich, dass Vorsorge dafür getroffen wird, dass der Schlüssel weiterhin geheim gehalten wird, und dass ein Wechsel des Schlüssels vorgesehen wird, sollte die grundlegende Sicherheit gefährdet sein.

[0004] Verschlüsselungsprotokolle mit öffentlichem Schlüssel wurden zuerst von Diffie-Hellman im Jahr 1976 vorgeschlagen und verwendeten einen öffentlichen Schlüssel, der allen potentiellen Korrespondenten verfügbar gemacht wurde, und einen privaten Schlüssel, der nur dem bestimmungsgemäßen Empfänger bekannt war. Die öffentlichen und privaten Schlüssel sind so miteinander verknüpft, dass eine Nachricht, die mit dem öffentlichen Schlüssel eines Empfängers verschlüsselt wurde, nur mit dem privaten Schlüssel leicht entschlüsselt werden kann, jedoch der private Schlüssel nicht aus dem Klartext, dem verschlüsselten Text und dem öffentlichen Schlüssel hergeleitet werden kann.

[0005] Ein Key-Establishment ist ein Vorgang, bei dem zwei (oder mehrere) Parteien einen gemeinsamen geheimen Schlüssel festlegen, den so genannten Sitzungsschlüssel. Der Sitzungsschlüssel wird daraufhin verwendet, um einige Verschlüsselungsziele wie beispielsweise Geheimhaltung zu erzielen. Es gibt zwei Arten von Key-Agreement-Protokollen: Key-Transport-Protokolle, bei denen ein Schlüssel von einer Partei erzeugt wird und geheim zu der zweiten Partei übertragen wird, und Key-Agreement-Protokolle, bei denen beide Parteien Informationen beisteuern, die zusammen den gemeinsamen geheimen Schlüssel festlegen. Die Anzahl der Nachrichtenaustauschvorgänge, die zwischen den Parteien erforderlich sind, wird als Durchlaufanzahl bezeichnet. Man sagt, dass ein Key-Establishment-Protokoll eine implizierte Schlüsselauthentisierung (oder einfache Schlüsselauthentisierung) leistet, wenn eine Partei sich sicher ist, dass außer einer speziell identifizierten zweiten Partei keine andere Partei den Wert des Sitzungsschlüssel erfahren kann. Die Eigenschaft der implizierten Schlüsselauthentisierung bedeutet aber nicht zwangsweise, dass die zweite Partei tatsächlich den Sitzungsschlüssel besitzt. Ein Key-Establishment-Protokoll soll eine Schlüsselbestätigung bieten, wenn eine Partei sich sicher ist, dass eine bestimmte identifizierte Partei tatsächlich im Besitz eines bestimmten Sitzungsschlüssels ist. Wenn die Authentisierung für beide Parteien, die in dem Protokoll involviert sind, vorgesehen ist, dann kann gesagt werden, dass die Schlüsselauthentisierung gegenseitig ist. Wenn sie nur für eine einzige Partei vorgesehen ist, dann ist die Authentisierung nur einseitig.

[0006] Es gibt verschiedene frühere Vorschläge, die beanspruchen, dass sie eine implizierte Schlüsselauthentisierung bieten.

[0007] Beispiele hierfür sind das Nyberg-Rueppel-Protokoll mit einem Durchlauf und das Matsumoto-Takashima-Imai (MTI)- und das Goss und Yacobi-Protokoll mit zwei Durchläufen zur Vereinbarung eines Schlüssels. Das Nyberg-Rueppel-Protokoll und das MTI-Protokoll sind in der EP 0 639 907 und in Matsumoto, T., Takashima, Y. und Imai, H.: On Seeking Smart Public-Key-Distribution Systems, The Transactions of the IECE of Japan, E69: 99-106, 1986 beschrieben. Das Goss-Protokoll ist in dem US-Patent Nr. 4,956,865 beschrieben. Das Yacobi-Protokoll ist in Y. Yacobi, "A Key Distribution Paradox", Advances in Cryptology, Crypto '90, Lecture Notes in Computer Science 537, Springer-Verlag, 1991, Seiten 268 bis 273 beschrieben.

[0008] Diese früheren Vorschläge stellen sicher, dass Übertragungen zwischen Korrespondenten zum Festlegen eines gemeinsamen Schlüssels sicher sind und dass ein Lauscher den Sitzungsschlüssel nicht abfragen und den verschlüsselten Text nicht entschlüsseln kann. Auf diese Weise wird Sicherheit für heikle Transaktionen wie beispielsweise Geldtransaktionen geboten.

[0009] Beispielsweise legt das MTI/AO-Key-Agreement-Protokoll einen gemeinsamen sicheren Schlüssel K, der den beiden Korrespondenten bekannt ist, auf folgende Weise fest:

1. Während eines einmaligen ersten Setups wird die Erzeugung eines Schlüssels und dessen Veröffentlichung dadurch vorgenommen, dass ein geeigneter Primwert p und ein Generator $\alpha \in Z_p^*$ des Systems in einer Weise ausgewählt und veröffentlicht werden, die eine Authentizität garantieren. Der Korrespondent A wählt als privaten Langzeitschlüssel eine ganzzahlige Zufallszahl „a", $1 < a < p - 1$ und berechnet einen öffentlichen Lang-

zeitschlüssel $z_A = \alpha^a \bmod p$. B erzeugt analog die Schlüssel b, z_B. A und B haben Zugang zu authentisierten Kopien vom jeweiligen öffentlichen Langzeitschlüssel des anderen.
2. Das Protokoll erfordert den Austausch der folgenden Nachrichten:

$$A \rightarrow B{:}\alpha^x \bmod p \quad (1)$$

$$A \leftarrow B{:}\alpha^y \bmod p \quad (2)$$

Die Werte von x und y bleiben während derartiger Übertragungen geheim, da es unmöglich ist, den Exponenten zu ermitteln, sogar wenn der Wert von a und die Exponentation bekannt ist, selbstverständlich vorausgesetzt, dass p ausreichend groß gewählt wurde.
3. Um das Protokoll zu implementieren, werden jedes Mal, wenn ein gemeinsamer Schlüssel gefordert wird, die folgenden Schritte ausgeführt.
(a) A wählt eine ganzzahlige Zufallszahl x, $1 \le x \le p - 2$, und sendet B die Nachricht (1), das heißt $\alpha^x \bmod p$.
(b) B wählt eine ganzzahlige Zufallszahl y, zu der $1 \le y \le p - 2$ und sendet A die Nachricht (2), das heißt $\alpha^y \bmod p$.
(c) A berechnet den Schlüssel $K = (\alpha^y)^a z_B{}^x \bmod p$.
(d) B berechnet den Schlüssel $K = (\alpha^x)^b z_A{}^y \bmod p$.
(e) Beide teilen sich den Schlüssel $K = \alpha^{bx+ay}$.

[0010] Um den Schlüssel K zu berechnen, muss A seinen geheimen Schlüssel a und die ganzzahlige Zufallszahl x verwenden, die beide nur ihm bekannt sind. Gleichermaßen muss B seinen geheimen Schlüssel b und die ganzzahlige Zufallszahl y verwenden, um den Sitzungsschlüssel K zu berechnen.

[0011] Vorausgesetzt die geheimen Schlüssel a, b bleiben ungefährdet, kann ein Lauscher keinen Sitzungsschlüssel erzeugen, der mit dem anderen Korrespondenten identisch ist. Entsprechend wird irgendein verschlüsselter Text für beide Korrespondenten nicht entschlüsselbar sein.

[0012] Die EP 0 639 907 beschreibt ein Key-Agreement-Verfahren. Ein Benutzer A besitzt einen privaten Key-Agreement-Schlüssel s_A. Ein Benutzer B besitzt einen öffentlichen Key-Agreement-Schlüssel k_B, der dem privaten Key-Agreement-Schlüssel s_B über die Vorschrift $k_B = g^{-s_B} \bmod p$ entspricht.

[0013] Diese und andere Protokolle wurden als solche für die Festlegung eines Schlüssels und gegenüber herkömmlichen Lauschangriffen oder man-in-the-middle-Angriffen als ausreichend widerstandsfähig erachtet.

[0014] Unter gewissen Umständen könnte es für einen Gegner vorteilhaft sein, einen Korrespondenten im Hinblick auf die wahre Identität des anderen Korrespondenten zu täuschen.

[0015] Bei einem solchen Angriff modifiziert ein aktiver Gegner oder Lauscher E Nachrichten, die zwischen A und B ausgetauscht werden, mit dem Ergebnis, dass B glaubt, dass er einen Schlüssel K mit E teilt, während A glaubt, dass sie den gleichen Schlüssel K mit B teilt. Obwohl E den Wert von K nicht erfährt, kann die Fehlinformation in Bezug auf die Identität der Korrespondenten nützlich sein.

[0016] Ein praktisches Szenario, bei dem eine solche Attacke erfolgreich eingesetzt werden kann, ist wie folgt. Es wird angenommen, dass B eine Zweigstelle einer Bank ist und A ist ein Kontoinhaber. Von der Hauptgeschäftsstelle der Bank werden Zertifikate ausgegeben und in dem Zertifikat befinden sich die Informationen betreffend das Konto des Inhabers. Angenommen, das Protokoll zum elektronischen Zahlen einer Geldsumme besteht darin, einen Schlüssel mit einer Zweigstelle einer Bank über ein gegenseitig authensierendes Key-Agreement-Verfahren auszutauschen. Sobald B die sendende Identität authentisiert hat, werden die verschlüsselten Geldsummen auf die Kontonummer in dem Zertifikat eingezahlt. Wenn keine weitere Authentisierung in der verschlüsselten Zahlungsanweisung erfolgt (was der Fall sein könnte, um Bandbreite einzusparen), dann wird die Zahlung auf das Konto von E geleistet.

[0017] Es ist somit eine Aufgabe der vorliegenden Erfindung, ein Protokoll bereitzustellen, bei dem die obigen Nachteile überwunden oder verringert sind.

[0018] Somit wird gemäß der vorliegenden Erfindung ein Verfahren zum Authentisieren eines Schlüssels geschaffen, der zwischen einem Korrespondentenpaar festgelegt wird, wie es in den anhängigen Ansprüchen beschrieben ist.

[0019] Obwohl der Lauscher E bei der Übertragung deren öffentlichen Schlüssel $p_E = \alpha^{ae}$ als Teil der Nachricht austauschen kann, wird B somit p_E anstatt p_A verwenden, wenn die Nachricht authentifiziert wird. Entsprechend werden die berechneten und übertragenen Werte der Exponentialfunktionen nicht einander entsprechen.

[0020] Es werden nun rein beispielhaft Ausführungsformen der Erfindung unter Bezugnahme auf die beigefügten Zeichnungen geschrieben, wobei:

[0021] Fig. 1 eine schematische Darstellung eines Datenkommunikationssystems ist.

[0022] Es wird deshalb auf die Fig. 1 Bezug genommen, in der ein Korrespondentenpaar **10**, **12**, die als Korrespondent A und Korrespondent B bezeichnet sind, über einen Kommunikationskanal **14** Informationen austauschen. Eine Verschlüsselungseinheit **16**,

18 ist zwischen jedem der Korrespondenten **10**, **12** und dem Kanal **14** zwischengeschaltet. Jeder Verschlüsselungseinheit **16**, **18** ist ein Schlüssel **20** zugeordnet, um einen Klartext, der zwischen jeder Einheit **16**, **18** und deren entsprechenden Korrespondenten **10**, **12** übertragen wird, in verschlüsselten Text zu konvertieren, der über den Kanal **14** übertragen wird. Während des Betriebs wird eine Nachricht, die durch den Korrespondenten A, **10** erzeugt wird, durch die Einheit **16** mit dem Schlüssel **20** verschlüsselt und als verschlüsselter Text über den Kanal **14** an die Einheit **18** übermittelt.

[0023] Der Schlüssel **20** wird in der Einheit **18** auf den verschlüsselten Text angewandt, um eine Nachricht mit Klartext für den Korrespondenten B, **12** zu erzeugen. Vorausgesetzt, die Schlüssel **20** entsprechen einander, dann wird die Nachricht, die der Korrespondent **12** erhält, die sein, die von dem Korrespondenten **10** gesendet wurde.

[0024] Damit das in der Fig. 1 gezeigte System einwandfrei arbeitet, ist es notwendig, dass die Schlüssel **20** identisch sind und deswegen wird ein Key-Agreement-Protokoll eingerichtet, das die öffentliche Übertragung von Informationen erlaubt, um die identischen Schlüssel festzulegen. Für eine solche Schlüsselerzeugung sind etliche Protokollen verfügbar, die Abwandlungen des Schlüsselaustausches gemäß Diffie-Hellman sind. Deren Zweck besteht darin, für die Parteien A und B einen geheimen Sitzungsschlüssel K festzulegen.

[0025] Die Systemparameter für diese Protokolle sind eine Primzahl p und ein Generator α der multiplikativen Gruppe Z_p^*. Der Korrespondent A besitzt einen privaten Schlüssel a und einen öffentlichen Schlüssel $p_A = \alpha^a$. Der Korrespondent B besitzt einen privaten Schlüssel b und einen öffentlichen Schlüssel $p_B = \alpha^b$. In dem folgenden beispielhaften Protokoll ist der $Text_A$ ein Informationsstring, der die Partei A identifiziert. Wenn der andere Korrespondent B eine authentische Kopie des öffentlichen Schlüssels des Korrespondenten A besitzt, dann wird der $Text_A$ das von einer gesicherten Stelle ausgegebene Zertifikat für den öffentlichen Schlüssel von A enthalten. Der Korrespondent B kann diese authentische Kopie des öffentlichen Schlüssels der gesicherten Stelle verwenden, um das Zertifikat des Korrespondenten A zu verifizieren, und somit erhält er eine authentische Kopie des öffentlichen Schlüssels des Korrespondenten A.

[0026] In jedem nachfolgenden Beispiel wird angenommen, dass ein Lauscher E versucht, dass Nachrichten von A als von E stammend identifiziert werden. Um dies zu erreichen, wählt E eine ganzzahlige Zufallszahl e aus, $1 \le e \le p - 2$, berechnet $p_E = (p_A)^e = \alpha^{ae} \bmod p$, und erhält dies als seinen öffentlichen Schlüssel zertifiziert. E kennt den Exponenten ae nicht, obwohl er e kennt. Indem der $Text_A$ durch den $Text_E$ ersetzt wird, wird der Korrespondent B annehmen, dass die Nachricht von E anstatt von A stammt und verwendet den öffentlichen Schlüssel E, um den Sitzungsschlüssel K zu erzeugen. E fängt auch die Nachricht von B ab und verwendet dessen geheime ganzzahlige Zufallszahl e, um dessen Inhalte zu modifizieren. A wird dann diese Information dazu benutzen, den gleichen Sitzungsschlüssel zu erzeugen, der A erlaubt, mit B zu kommunizieren.

[0027] Um zu verhindern, dass der Lauscher E die Partei B davon überzeugt, dass er mit E kommuniziert, ist das folgende Protokoll eingerichtet.

[0028] Der Zweck des Protokolls besteht darin, dass die Parteien A und B einen Sitzungsschlüssel K einrichten. Die beispielhaft erläuterten Protokolle sind rollen-symmetrisch [role-symmetric] und nicht interaktiv [non-interactive]. Die als erstes Protokoll, modifiziertes erstes Protokoll, zweites Protokoll, drittes Protokoll und Key-Transport-Protokoll bezeichneten Protokolle werden erläutert, um allgemeine Konzepte vorzustellen. Diese speziellen Protokolle sind aber nicht Teil der derzeit beanspruchten Erfindung.

[0029] Die Systemparameter für diese Protokolle sind eine Primzahl p und ein Generator α der multiplikativen Gruppe Z_p^*. Ein Benutzer A besitzt einen privaten Schlüssel a und einen öffentlichen Schlüssel $p_A = \alpha^a$. Der Benutzer B besitzt einen privaten Schlüssel b und einen öffentlichen Schlüssel $p_B = \alpha^b$.

Erstes Protokoll

1. A wählt eine ganzzahlige Zufallszahl x, $1 \le x \le p - 2$, aus und berechnet $r_A = \alpha^x$ und eine Signatur $s_A = x - r_A a \bmod (p - 1)$. A sendet $\{r_A, s_A, Text_A\}$ an B.
2. B wählt eine ganzzahlige Zufallszahl y, $1 \le y \le p - 2$, aus und berechnet $r_B = \alpha^y$ und eine Signatur $s_B = y - r_B b \bmod (p - 1)$. B sendet $\{r_B, s_B, Text_B\}$ an A.
3. A berechnet $\alpha^{s_B} (p_B)^{r_B}$ und verifiziert, dass dies mit r_B übereinstimmt. A berechnet den Sitzungsschlüssel $K = (r_B)^x = \alpha^{xy}$.
4. B berechnet $\alpha^{s_A} (p_A)^{r_A}$ und verifiziert, dass dies gleich r_A ist. B berechnet den Sitzungsschlüssel $K = (r_A)^y = \alpha_{xy}$.

[0030] Sollte E den $Text_A$ durch den $Text_E$ ersetzen, wird B $\alpha^{s_B} (p_E)^{r_A}$ berechnen, was nicht mit dem übermittelten Wert von r_A übereinstimmen wird. B wird somit vor dem Lauscher E gewarnt und wird damit weitermachen, einen anderen Sitzungsschlüssel einzuführen.

[0031] Ein Nachteil des ersten Protokolls besteht darin, dass keine perfekte Vorwärtssicherheit bietet. Das heißt, wenn ein Gegner den privaten Langzeit-

schlüssel a der Partei A erfährt, dann kann der Gegner alle letzten Sitzungsschlüssel von A herleiten. Die perfekte Vorwärtssicherheit kann durch Modifizierung des Protokolls 1 in der nachfolgenden Weise erzielt werden.

Modifiziertes erstes Protokoll

[0032] Im Schritt 1 sendet A auch α^{x_1} an B, wobei x_1 eine zweite ganzzahlige Zufallszahl ist, die von A generiert wurde. In gleicher Weise sendet im obigen Schritt 2B auch α^{y_1} an A, wobei y_1 eine ganzzahlige Zufallszahl ist. A und B berechnen nun den Schlüssel $K = \alpha^{xy} \oplus \alpha^{x_1 y_1}$.

[0033] Ein weiterer Nachteil des ersten Protokolls besteht darin, dass für den Fall, dass ein Gegner die private ganzzahlige Zufallszahl x von A erfährt, der Gegner den privaten Langzeitschlüssel a der Partei A aus der Gleichung $s_A = x - r_A a \bmod p - 1$ berechnen kann. Dieser Nachteil ist primär theoretischer Natur, da eine gut gestaltete Implementierung des Protokolls verhindern wird, dass die privaten ganzzahligen Zahlen offengelegt werden.

Zweites Protokoll

[0034] Ein zweites Protokoll, das nachfolgend erläutert wird, zielt auf diese zwei Nachteile ab.

1. A wählt eine ganzzahlige Zufallszahl x, $1 \le x \le p - 2$, aus und berechnet $(p_B)^x$, α^x und eine Signatur $s_A = x + a(p_B)^x \bmod(p - 1)$. A sendet $\{\alpha^x, s_A, \text{Text}_A\}$ an B.
2. B wählt eine ganzzahlige Zufallszahl y, $1 \le y \le p - 2$ aus und berechnet $(p_A)^y$, α^y und eine Signatur $s_B = y + b(p_A)^y \bmod(p-1)$. B sendet $\{\alpha^y, s_B, \text{Text}_B\}$ an A.
3. A berechnet $(a^y)^a$ und verifiziert, dass $\alpha^{s_B}(p_B)^{-\alpha^{ay}} = a^y$. A berechnet dann den Sitzungsschlüssel $K = \alpha^{ay}(pB)^x$.
4. B berechnet $(\alpha^x)^b$ und verifiziert, dass $\alpha^{s_A}(p_A)^{-\alpha^{bx}} = \alpha^x$. A berechnet dann den Sitzungsschlüssel $K = \alpha^{by}(pA)^y$.

[0035] Das zweite Protokoll verbessert das erste Protokoll dahingehend, dass es die perfekte Vorwärtssicherheit bietet. Es ist zwar weiterhin der Fall, dass eine Offenbarung einer privaten ganzzahligen Zufallszahl x einem Gegner erlaubt, den privaten Schlüssel a zu erfahren, dies wird aber in der Praxis kein Problem sein, weil A x zerstören kann, sobald sie diesen im Schritt 1 des Protokolls verwendet.

[0036] Wenn A keine authentisierte Kopie des öffentlichen Schlüssels von B besitzt, dann muss B zu Beginn des Protokolls eine zertifizierte Kopie dieses Schlüssels an B übermitteln. In diesem Fall ist das zweite Protokoll ein Protokoll mit drei Durchläufen.

[0037] Die Größe s_A dient als Signatur von A auf den Wert α^x. Diese Signatur hat die neue Eigenschaft, dass sie nur die Partei B verifiziert werden kann. Diese Idee kann auf alle ElGamalartigen Signaturschema generalisiert werden.

[0038] Die obigen ersten und zweiten Protokolle können modifiziert werden, um die Bandbreitenanforderungen und die Recheneffizienz der Schlüsselvereinbarung zu verbessern. Die modifizierten Protokolle sind nachfolgend als Protokoll 1' und Protokoll 2' bezeichnet. In jedem Fall teilen sich A und B den gemeinsamen Schlüssel $\alpha^{s_A s_B}$.

Protokoll 1'

1. A wählt eine ganzzahlige Zufallszahl x, $1 \le x \le p - 2$, aus und berechnet $r_A = \alpha^a$ und $s_A = x + r_A a\alpha^a \bmod(p - 1)$. A sendet $\{r_A, \text{Text}_A\}$ an B.
2. B wählt eine ganzzahlige Zufallszahl y, $1 \le y \le p - 2$, aus und berechnet $r_B = \alpha^y$ und $s_B = y + r_B b\alpha^b \bmod(p - 1)$. B sendet $\{r_B, \text{Text}_B\}$ an A. 6
3. A berechnet $K = (r_B(p_B)^{r_B\alpha^b})^{s_A}$, das äquivalent zu $\alpha^{s_A s_B}$ ist.
4. B berechnet $K = (r_A(p_A)^{r_B\alpha^a})^{s_B}$, das ebenfalls äquivalent zu $\alpha^{s_A s_B}$ ist.

[0039] A und B teilen sich somit den gemeinsamen Schlüssel, es ist aber zu beachten, dass die Signaturen s_A und s_B nicht übermittelt werden müssen.

Protokoll 2'

1. A wählt eine ganzzahlige Zufallszahl x, $1 \le x \le p - 2$, aus und berechnet $(p_B)^x$, α^x und $s_A = x + a(p_B)^x \bmod(p - 1)$. A sendet $\{\alpha^x, \text{Text}_A\}$ an B.
2. B wählt eine ganzzahlige Zufallszahl y, $1 \le y \le p - 2$, aus und berechnet $(p_A)^y$, α^y und $s_B = y + b(p_A)^y \bmod(p - 1)$.B sendet $\{\alpha^y, \text{Text}_B\}$ an A.
3. A berechnet $(\alpha^y)^a$ und $K = (\alpha^y(p_B)^{\alpha^b\alpha^{ay}})^{s_A}$, d.h. $\alpha^{s_A s_B}$.
4. B berechnet $(\alpha^x)^b$ Und $K = (\alpha^x(p_A)^{\alpha^a\alpha^{bx}})^{s_B}$, d.h. $\alpha^{s_A s_B}$.

[0040] Abermals wird somit eine Übermittlung von s_A und s_B vermieden.

[0041] Ein weiteres Protokoll ist für die Parteien A und B zum Einrichten eines Sitzungsschlüssels K verfügbar.

Drittes Protokoll

[0042] Die Systemparameter für dieses Protokoll sind eine Primzahl p und ein Generator α für die multiplikative Gruppe Z_p^*. Der Benutzer A besitzt einen privaten Schlüssel a und einen öffentlichen Schlüssel $p_A = \alpha^a$. Der Benutzer B besitzt einen privaten Schlüssel b und einen öffentlichen Schlüssel $p_B = \alpha^b$.

1. A wählt zwei ganzzahlige Zufallszahlen x, x_1, 1

≤ x, x_1 ≤ p – 2 aus und berechnet $r_{x_1} = \alpha^{x_1}$, $r_A = \alpha^x$ und $(r^A)^{r_{x1}}$, dann berechnet er eine Signatur $s_A = xr_{x_1} - (r_A)^{r_{x1}} a\alpha^a \bmod(p-1)$. A sendet $\{r_A, s_A, \alpha^{x_1}$, Text$_A\}$an B.

2. B wählt zwei ganzzahlige Zufallszahlen y, y_1, 1 ≤ y, y_1 ≤ p – 2 aus uns berechnet $r_{y_1} = \alpha^{y_1}$, $r_B = \alpha^y$ und $(r^B)^{r_{y1}}$, dann berechnet er eine Signatur $s_B = yr_{y_1} - (r_B)^{r_{y1}} b \bmod(p-1)$. A sendet $\{r_B, s_B, \alpha^{y_1}$ Text$_B\}$ an A.

3. A berechnet $\alpha^{S_B}(p_B)^{(r_B)^{r_{y1}}}$und verifiziert, dass dies gleich $(r_B)^{r_{y1}}$ ist. A berechnet einen Sitzungsschlüssel $K = (\alpha^{y_1})^{x_1} = \alpha^{x_1 y_1}$.

4. B berechnet $\alpha^{s_A}(p_A)^{(r_A)^{r_{x1}}}$ und verifiziert, dass dies gleich $(r_A)^{r_{x1}}$ ist. B berechnet den Sitzungsschlüssel $K = (\alpha^{x_1})^{y_1} = \alpha^{x_1 y_1}$.

[0043] In diesen Protokollen kann (r_A, s_A) als Signatur von r_{x_1} sein, mit der Eigenschaft, dass nur A die Nachricht r_{x_1} signieren kann.

Key-Transport-Protokoll

[0044] Die oben beschriebenen Protokolle erlauben die Einrichtung und Authentisierung eines Sitzungsschlüssels K. Es ist auch wünschenswert, ein Protokoll einzurichten, bei dem es A erlaubt ist, einen Sitzungsschlüssel K an die Partei B zu schicken. Ein solches Protokoll wird nachfolgend beispielhaft erläutert.

1. A wählt eine ganzzahlige Zufallszahl x, 1 ≤ x ≤ p – 2, aus und berechnet ein $r_A = \alpha^x$ und eine Signatur $s_A = x - r_A a\alpha^a \bmod(p-1)$. A berechnet den Sitzungsschlüssel $K = (P_B)^x$ und sendet $\{r_A, s_A$, Text$_A\}$ an B.

2. B berechnet $\alpha^{s_A}(p_A)^{-r_A\alpha^a}$ und verifiziert, dass diese Größe gleich r_A ist. B berechnet den Sitzungsschlüssel $K = (r_A)^b$.

Modifiziertes Key-Transport-Protokoll

[0045] Das obige Protokoll kann modifiziert werden, um die Bandbreite zu reduzieren, indem die Notwendigkeit, die Signatur s_A zu übertragen, vermieden wird, was folgendermaßen erfolgt:

1. A wählt eine ganzzahlige Zufallszahl x, 1 ≤ x ≤ p – 2, aus und berechnet $r_A = \alpha^x$ und $s_A = x - r_A a\alpha^a \bmod(p-1)$. A berechnet $K = (P_B)^{S_A}$ und sendet $\{r_A$, Text$_A\}$ an B.

2. B berechnet $K = (\alpha^x (p_A)^{-r_A\alpha^a})^b = \alpha^{bs_A}$.

[0046] Alle Key-Transport-Protokolle mit einem Durchlauf weisen das folgende Wiedergabeproblem auf. Es wird angenommen, dass ein Key-Transport-Protokoll mit einem Durchlauf dazu verwendet wird, einen Sitzungsschlüssel K wie auch irgendeinen Text, der mit dem Sitzungsschlüssel K verschlüsselt wurde, von A an B zu übertragen. Es wird auch angenommen, dass E die Übermittlung von A an B aufzeichnet. Wenn E zu einem späteren Zeitpunkt Zugang zu B's Entschlüsselungsmaschine (aber nicht auf den inneren Gehalt der Maschine, wie beispielsweise B's privaten Schlüssel) erhalten kann, dann kann E durch Wiedergeben der Übertragung an die Maschine den Originaltext wiederherstellen (in diesem Szenario erfährt E den Sitzungsschlüssel K nicht).

[0047] Diese Wiedergabeattacke kann mit gewöhnlichen Verfahren vereitelt werden, wie beispielsweise der Verwendung von Zeitstempeln. Es gibt aber einige praktische Situationen, wenn B begrenzte Rechenkapazitäten hat, bei denen es zu Beginn jeder Sitzung zweckmäßiger ist, dass B einen zufälligen Bitstring k an A übermittelt. Der Sitzungsschlüssel, der zum Verschlüsseln des Textes verwendet wird, ist dann $k \oplus K$, d.h. k XOR'd mit K.

[0048] Die Signiergleichung $s_A = x - r_A a\alpha^a$, wobei im Protokoll 1 gilt: $r_A = \alpha^x$, und die Key-Transport-Protokolle, wobei im Protokoll 2 $r_A = \alpha^{xb}$ gilt, können durch verschiedene Varianten ersetzt werden. Einige dieser Varianten sind:

$r_A = s_A x + z$

$s_A = x\alpha^a + ar_A$

$s_A = xr_A + a\alpha^a$

$1 = ar_A + xs_A$

[0049] Alle zuvor erläuterten Protokolle wurden in Verbindung mit der multiplikativen Gruppe Z_p^* beschrieben. Sie können aber alle ohne weiteres modifiziert werden, so dass sie in irgendeiner finiten Gruppe arbeiten, in der das diskrete Logarithmusproblem hartnäckig erscheint. Eine geeignete Auswahl umfasst die multiplikative Gruppe einer Teilüberdeckung [finite filed] (insbesondere die Teilüberdeckung $GR(2^n)$, Untergruppen von Z_p^* der Ordnung q und die Gruppe der Punkte auf einer elliptischen Kurve, die über ein Galoisfeld [finite field] bestimmt sind. In jedem Fall wird ein geeigneter Generator α dazu eingesetzt, die öffentlichen Schlüssel zu definieren.

[0050] Die zuvor erläuterten Protokolle können auch in direkter Weise modifiziert werden, um die Situation im griff zu haben, wenn jeder Benutzer seines eigenen Systemparameter p und α auswählt (oder analoge Parameter, wenn anstatt Z_p^* eine andere Gruppe verwendet wird).

[0051] In den obigen Protokollen wurde eine Signaturkomponente der allgemeinen Form $s_A = x + r \cdot a \cdot \alpha^a$ verwendet.

[0052] Die Protokolle können modifiziert werden, um eine einfachere Signaturkomponente der allgemeinen Form $s_A = x + r_a \cdot a$ zu verwenden, ohne dass die Sicherheit gefährdet wird.

[0053] Nachfolgend werden Beispiele für solche Protokolle unter Verwendung der gleichen Notation beschrieben, obwohl es klar ist, dass die Protokolle in einer alternativen Notation ausgedrückt werden könnten, wenn dies bevorzugt würde.

Protokoll 1"

[0054] Dieses Protokoll wird unter Verwendung einer Implementierung in der multiplikativen Gruppe Z_p^* mit der folgenden Notation beschrieben:
p ist eine Primzahl,
α ist ein Generator von Z_p^*,
a und b sind die jeweiligen privaten Langzeitschlüssel der Parteien A und B,
α^a mod p ist der öffentliche Langzeitschlüssel der Partei A,
α^b mod p ist der öffentliche Langzeitschlüssel der Partei B,
x ist eine ganzzahlige Zufallszahl, die von A als privater Kurzzeitschlüssel ausgewählt wird,
$r_a = \alpha^x$ mod p ist der öffentliche Kurzzeitschlüssel der Partei A,
y ist eine ganzzahlige Zufallszahl, die von B als privater Kurzzeitschlüssel ausgewählt wird,
$r_b = \alpha^y$ mod p ist der öffentliche Kurzzeitschlüssel der Partei B,
$\bar{r}_a$ ist ein ganzzahliger Wert, der von r_a hergeleitet ist, und umfasst typischerweise die 80 niedrigwertigsten Bit von r_a,
$\bar{r}_b$ ist ein ganzzahliger Wert, der von r_b abgeleitet ist, und umfasst typischerweise die 80 niedrigwertigsten Bit von r_b.

[0055] Zur Implementierung des Protokolls wird folgendes Ausgeführt:

1. A sendet r_a an B.
2. B sendet r_B an A.
3. A berechnet $s_A = x + \bar{r}_a \cdot a \bmod (p-1)$.
4. A berechnet den Sitzungsschlüssel K, wobei $K = (\alpha^y(\alpha^b)^{\bar{r}_b})^{s_1} \bmod p$.
5. B berechnet $s_B = y + \bar{r}_b \cdot b \bmod (p-1)$.
6. B berechnet den Sitzungsschlüssel K, wobei $K = (\alpha^x(\alpha^a)^{\bar{r}_a})^{s_B} \bmod p$.
7. Das gemeinsame Geheimnis ist $\alpha^{s_B s_A} \bmod p$.

[0056] In diesem Protokoll sind die Anforderungen an die Bandbreite wiederum dadurch reduziert, dass die Signaturkomponenten die Kurz- und Langzeitschlüssel der Korrespondenten kombinieren, um eine Attacke eines Lauschers zu verhindern.

[0057] Das obige Protokoll kann auch unter Verwendung einer Untergruppe von Z_p^* implementiert werden. In diesem Fall wird q ein Primzahldivisor von (p – 1) sein und g wird ein Element der Ordnung p in Z_p^* sein.

[0058] Die öffentlichen Schlüssel von A und B werden die Form g^a bzw. g^b haben und die Kurzzeitschlüssel r_a, r_b werden die Form g^x, g^y haben.

[0059] Die Signaturkomponenten s_A, s_B werden mit mod q und der Sitzungsschlüssel k wird mit mod q wie zuvor berechnet. Das gemeinsame Geheimnis ist dann $\alpha^{s_B s_A} \bmod p$.

[0060] Wie zuvor erwähnt, können die Protokolle in anderen Gruppen als der Z_p^* implementiert werden und eine besonders robuste Gruppe ist die Gruppe von Punkten auf einer elliptischen Kurfe über ein Galoisfeld. Ein Beispiel einer solchen Implementierung wird nachfolgend als Protokoll 1"' erläutert.

Protokoll 1"'

[0061] Es wird die folgende Notation verwendet:
E ist eine elliptische Kufe, die über Fq definiert ist,
P ist ein Punkt primer Ordnung n in E(Fq),
da (1 < da < n – 1) ist der private Langzeitschlüssel der Partei A,
d_b (1 < d_b < n – 1) ist der private Langzeitschlüssel der Partei B,
Q_a = daP ist der öffentliche Langzeitschlüssel der Partei A,
$Q_b = d_bP$ ist der öffentliche Langzeitschlüssel der Partei B,
k(1 < k < n – 1) ist der private Kurzzeitschlüssel der Partei A,
r_a = kP ist der öffentliche Kurzzeitschlüssel der Partei A,
m(1 < m < n – 1) ist der private Kurzzeitschlüssel der Partei B,
r_b = mP ist der öffentliche Kurzzeitschlüssel der Partei B,
$\bar{r}_a$ und $\bar{r}_b$ sind jeweils Bitstrings, beispielsweise die 80 niedrigwertigsten Bit der x-Koordinate von r_a und r_b.

[0062] Zum Implementieren des Protokolls erfolgt Folgendes:

1. A sendet r_a an B.
2. B sendet r_b an A.
3. A berechnet $s_A = (k + \bar{r}_b \cdot d_a) \bmod n$.
4. A berechnet den Sitzungsschlüssel K, wobei $K = s_a(r_b + \bar{r}_b Q_b)$.
5. B berechnet $s_B = (m + \bar{r}_b d_b) \bmod n$.
6. B berechnet den Sitzungsschlüssel K, wobei $K = s_b(r_a + \bar{r}_a Q_a)$.
7. Das gemeinsame Geheimnis ist $s_a s_b P$.

[0063] Es ist wiederum anzumerken, dass es nicht notwendig ist, die Signaturkomponenten s_A, s_B zwischen den Korrespondenten hin und her zu senden, jedoch sind die Kurz- und Langzeitschlüssel der Korrespondenten durch die Form der Komponenten kombiniert. (Um eine Verwirrung mit den Koordinaten (x, y) des Punktes auf der Kurve zu vermeiden, wird man es zu schätzen wissen, dass die Notation m für x, y in den vorherigen Beispielen ersetzt wurde).

Patentansprüche

1. Verfahren zum Authentisieren eines zwischen zwei Korrespondenten (**10**, **12**) A, B in einem Datenkommunikationssystem mit öffentlichen Schlüsseln hergestellten Schlüssels (**20**) zum Ermöglichen des Austausches von Informationen (**16**, **18**) zwischen diesen über einen Kommunikationskanal (**14**), wobei jeder der Korrespondenten (**10**, **12**) einen jeweiligen privaten Schlüssel und einen von einem Generator und jeweiligen der privaten Schlüssel abgeleiteten öffentlichen Schlüssel besitzt, mit den folgenden Schritten:
i) ein erster der Korrespondenten A (**10**) wählt eine erste zufällige ganze Zahl und erhebt eine den Generator enthaltende Funktion zu einer Potenz, um eine erste zur Potenz erhobene Funktion r_a bereitzustellen;
ii) der erste Korrespondent A (**10**) erzeugt eine erste Signatur s_A aus der zufälligen ganzen Zahl und einer von der zur Potenz erhobenen Funktion r_a abgeleiteten ganzen Zahl $\overline{r}_a$;
iii) der erste Korrespondent A (**10**) leitet eine von der ersten zur Potenz erhobenen Funktion r_a abgeleitete Nachricht zu einem zweiten Korrespondenten B weiter;
iv) der zweite Korrespondent B (**12**) wählt eine zweite zufällige ganze Zahl und erhebt eine den Generator enthaltende Funktion zu einer Potenz, um eine zweite zur Potenz erhobene Funktion r_b bereitzustellen, und erzeugt eine Signatur s_B, die aus der zweiten zufälligen ganzen Zahl und einer von der zweiten zur Potenz erhobenen Funktion r_b abgeleiteten ganzen Zahl $\overline{r}_b$ erhalten wird;
v) der zweite Korrespondent B (**12**) leitet eine Nachricht zu dem ersten Korrespondenten A weiter, die von der zweiten zur Potenz erhobenen Funktion r_b abgeleitet wird;
vi) jeder der Korrespondenten (**10**, **12**) konstruiert einen Sitzungsschlüssel K (**20**) durch Potenzieren von Informationen, die durch den anderen Korrespondenten veröffentlicht werden, mit seiner eigenen privaten Signatur.

2. Verfahren nach Anspruch 1, wobei die von dem ersten Korrespondenten (**10**) weitergeleitete Nachricht eine Identifikation des ersten Korrespondenten enthält.

3. Verfahren nach Anspruch 1 oder 2, wobei die von dem zweiten Korrespondenten (**12**) weitergeleitete Nachricht eine Identifikation des zweiten Korrespondenten (**12**) enthält.

4. Verfahren nach Anspruch 1, 2 oder 3, wobei die erste Funktion, die den Generator enthält, der Generator selbst ist.

5. Verfahren nach einem der vorhergehenden Ansprüche, wobei die zweite Funktion, die den Generator enthält, der Generator selbst ist.

6. Verfahren nach einem der vorhergehenden Ansprüche, wobei die erste Funktion, die den Generator enthält, den öffentlichen Schlüssel des zweiten Korrespondenten (**12**) enthält.

7. Verfahren nach einem der vorhergehenden Ansprüche, wobei die zweite Funktion, die den Generator enthält, den öffentlichen Schlüssel des ersten Korrespondenten (**10**) enthält.

8. Verfahren nach einem der vorhergehenden Ansprüche, wobei die von einem jeweiligen der Korrespondenten (**10**, **12**) erzeugte Signatur die zufällige ganze Zahl, die potenzierte Funktion und den privaten Schlüssel dieses einen Korrespondenten (**10**, **12**) kombiniert.

9. Verfahren nach einem der vorhergehenden Ansprüche, wobei die von einem anderen Korrespondenten (**10**, **12**) veröffentlichten Informationen den öffentlichen Schlüssel, die potenzierte Funktion und die ganze Zahl dieses einen Korrespondenten (**10**, **12**) kombinieren.

10. Verfahren nach Anspruch 9, wobei der öffentliche Schlüssel und die potenzierte Funktion multipliziert werden und der resultierende Wert mit der ganzen Zahl potenziert wird.

11. Verfahren nach einem der vorhergehenden Ansprüche, wobei der Generator ein Element α einer multiplikativen Gruppe Z_p^* ist, wobei p eine Primzahl ist.

12. Verfahren nach Anspruch 11, wobei das gemeinsame Geheimnis $\alpha^{s_A s_B}$ ist.

13. Verfahren nach Anspruch 11 oder 12, wobei die erste Signatur aus der ersten zufälligen ganzen Zahl x, der ganzen Zahl $\overline{r}_a$ und dem privaten Schlüssel a des ersten Korrespondenten A gemäß $s_A = x + \overline{r}_a a \bmod(p - 1)$ berechnet wird.

14. Verfahren nach Anspruch 11, 12 oder 13, wobei die zweite Signatur s_b aus der zweiten zufälligen ganzen Zahl y, der ganzen Zahl $\overline{r}_b$ und dem privaten Schlüssel b des zweiten Korrespondenten gemäß $s_B = y + \overline{r}_b b \bmod(p - 1)$ berechnet wird.

15. Verfahren nach Anspruch 11, 12, 13 oder 14, wobei das gemeinsame Geheimnis durch den ersten Korrespondenten aus dem öffentlichen Schlüssel α^y, der potenzierten Funktion α^b und der ganzen Zahl $\overline{r}_b$ des zweiten Korrespondenten B gemäß $K = \left(\alpha^y \left(\alpha^b\right)^{\overline{r}_b}\right)^{a} \bmod p$ berechnet wird.

16. Verfahren nach Anspruch 11, 12, 13, 14 oder 15, wobei das gemeinsame Geheimnis durch den

zweiten Korrespondenten aus dem öffentlichen Schlüssel α^x, der potenzierten Funktion α^a und der ganzen Zahl $\overline{r}_a$ des ersten Korrespondenten A gemäß $K = \left(\alpha^{x}\left(\alpha^{a}\right)^{\overline{r}_a}\right)^{y_a} \bmod p$ berechnet wird.

17. Verfahren nach einem der Ansprüche 11 bis 16, wobei das Element α ein Generator von Z_p^* ist.

18. Verfahren nach einem der Ansprüche 11 bis 16, wobei das Element α ein Generator g einer Untergruppe von Z_p^* der Ordnung q ist, wobei q ein Primdivisor von p – 1 ist.

19. Verfahren nach einem der Ansprüche 1 bis 10, wobei der Generator α ein Punkt P der Ordnung n auf einer über einem Galoisfeld Fq definierten elliptischen Kurve ist, Potenzierung wird durch Skalarmultiplikation auf der elliptischen Kurve durchgeführt.

20. Verfahren nach Anspruch 19, wobei die ganzen Zahlen Bitketten von Koordinaten der potenzierten Funktionen sind.

21. Verfahren nach Anspruch 20, wobei die Bitketten niedrigstwertige Bit der x-Koordinaten der potenzierten Funktionen sind.

22. Verfahren nach Anspruch 21, wobei die Bitketten die 80 niedrigstwertigen Bit der x-Koordinaten der potenzierten Funktionen sind.

23. Verfahren nach einem der Ansprüche 19 bis 22, wobei die erste Signatur aus der ersten zufälligen ganzen Zahl k, der ganzen Zahl $\overline{r}_a$ und dem privaten Schlüssel d_a des ersten Korrespondenten A gemäß $s_A = \left(k + \overline{r}_a d_a\right) \bmod n$ berechnet wird.

24. Verfahren nach einem der Ansprüche 19 bis 23, wobei die zweite Signatur aus der zweiten zufälligen ganzen Zahl m, der ganzen Zahl $\overline{r}_b$ und dem privaten Schlüssel d_b des zweiten Korrespondenten B gemäß $s_B = \left(m + \overline{r}_b d_b\right) \bmod n$ berechnet wird.

25. Verfahren nach einem der Ansprüche 19 bis 24, wobei der erste Korrespondent (**10**) den Sitzungsschlüssel aus der ersten Signatur s_A, der zweiten potenzierten Funktion r_b, der ganzen Zahl $\overline{r}_b$ und dem öffentlichen Schlüssel Q_b des zweiten Korrespondenten B gemäß $K = s_a\left(r_b + \overline{r}_b Q_b\right)$ berechnet.

26. Verfahren nach einem der Ansprüche 19 bis 25, wobei der zweite Korrespondent (**12**) den Sitzungsschlüssel aus der zweiten Signatur s_B, der ersten potenzierten Funktion r_a, der ganzen Zahl $\overline{r}_a$ und dem öffentlichen Schlüssel Q_a des ersten Korrespondenten A gemäß $K = S_b\left(r_a + \overline{r}_a Q_a\right)$ berechnet.

27. Verfahren nach einem der Ansprüche 19 bis 26, wobei der Punkt P eine prime Ordnung aufweist.

Es folgt ein Blatt Zeichnungen

Anhängende Zeichnungen

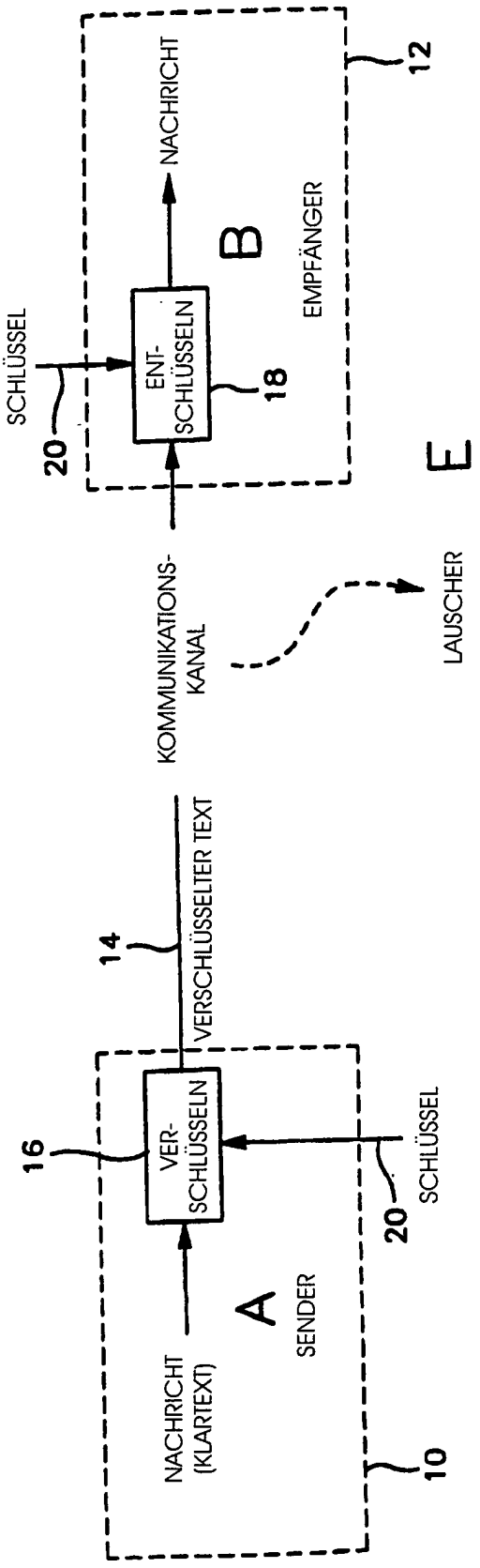

FIG. 1