(19) Deutsches Patent- und Markenamt

(10) **DE 10 2012 017 308 B4** 2016.05.12

(12) # Patentschrift

(21) Aktenzeichen: **10 2012 017 308.3**
(22) Anmeldetag: **03.09.2012**
(43) Offenlegungstag: **06.03.2014**
(45) Veröffentlichungstag der Patenterteilung: **12.05.2016**

(51) Int Cl.: ***H04L 12/951*** (2013.01)
H04N 21/60 (2011.01)

Innerhalb von neun Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten (§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(73) Patentinhaber:
global infinipool GmbH, 35037 Marburg, DE

(74) Vertreter:
Tergau & Walkenhorst Patentanwälte PartGmbB, 60322 Frankfurt, DE

(72) Erfinder:
Scholl, Martin, 35041 Marburg, DE

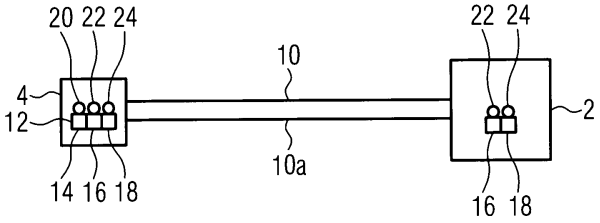
(56) Ermittelter Stand der Technik:

US 2009 / 0 028 142 A1
WO 2011/ 044 402 A1

Norm RFC768 28.August 1980. User Data Protocol

(54) Bezeichnung: **Verfahren zur Übertragung von Daten**

(57) Hauptanspruch: Verfahren zur Übertragung von Daten (12) von einem ersten Client (4, 6, 8) zu einem zweiten Client (4, 6, 8) in einem Netzwerk (1), umfassend die Verfahrensschritte:
– Zerlegen der Daten (12) in Chunks (14, 16, 18) durch den ersten Client (4, 6, 8),
– Schreiben einer Anzahl der Chunks (14, 16, 18) in einen ersten Stream durch den ersten Client (4, 6, 8),
– Senden des ersten Streams von dem ersten Client (4, 6, 8) an einen Server (2),
– Lesen und Speichern der Chunks (14, 16, 18) aus dem ersten Stream auf dem Server (2),
– Schreiben einer Anzahl der Chunks (14, 16, 18) in einen zweiten Stream durch den Server (2),
– Senden des zweiten Streams von dem Server an den zweiten Client (4, 6, 8),
dadurch gekennzeichnet, dass
dem jeweiligen Chunk (14, 16, 18) eine über den Chunk (14, 16, 18) gebildete Prüfsumme (20, 22, 24) zugeordnet wird, und dass
das Schreiben des jeweiligen Chunks (14, 16, 18) in den jeweiligen Stream durch den jeweiligen Sender (2, 4, 6, 8) nur auf eine Bestätigung des jeweiligen Empfängers (2, 4, 6, 8), dass der jeweilige Chunk auf dem jeweiligen Empfänger (2, 4, 6, 8) noch nicht vorhanden ist, auf Übersendung der dem Chunk (14, 16, 18) zugeordneten Prüfsumme (20, 22, 24) erfolgt.



Beschreibung

[0001] Die Erfindung betrifft ein Verfahren zur Übertragung von Daten von einem ersten Client zu einem zweiten Client in einem Netzwerk, umfassend die Verfahrensschritte:

– Zerlegen der Daten in Chunks durch den ersten Client,
– Schreiben einer Anzahl der Chunks in einen ersten Stream durch den ersten Client,
– Senden des ersten Streams von dem ersten Client an einen Server,
– Lesen und Speichern der Chunks aus dem ersten Stream auf dem Server,
– Schreiben einer Anzahl der Chunks in einen zweiten Stream durch den Server,
– Senden des zweiten Streams von dem Server an den zweiten Client.

[0002] Sie betrifft weiter Computerprogrammprodukte sowie eine Server zum Ausführen des Verfahrens. Derartige Verfahren und Vorrichtungen sind dem Fachmann aus der US 2009/0028142 A1, der Norm RFC 768 vom 28. August 1980 und der WO 2011/044402 A1 bekannt.

[0003] Die weltweite Vernetzung von Datennetzwerken und Computern im Internet ermöglicht es, Daten zwischen beliebigen Orten auf der Erde zu versenden, sofern ein Anschluss des jeweils sendenden und empfangenden Clients an das Internet gegeben ist. Die Datenübertragung im Internet basiert dabei weitgehend auf dem Transmission Control Protocol/Internet Protocol (TCP/IP).

[0004] Nachteilig ist hierbei die Impraktikabilität der Datenübertragung gerade bei Datensätzen im Terabyte-Bereich. Kommt es zu einem Verbindungsabbruch, muss unter Umständen der gesamte Datensatz neu geladen werden. Auch ein Datenfehler bei der Übertragung kann dazu führen, dass der gesamte Datensatz beim Empfänger unbrauchbar ist und neu geladen werden muss. Ein weiteres Problem stellt die Übertragungsgeschwindigkeit dar, die sich bei unterschiedlichen Leitungskapazitäten bei Sender und Empfänger stets nach dem langsamsten Teilnehmer der Datenübertragung richtet.

[0005] Aus den oben genannten Gründen wird die Datenübertragung von Datenmengen im Terabyte-Bereich über das Internet kaum praktiziert. Datenmengen dieser Größenordnung werden effektiv bislang über Wechselfestplatten, die physisch transferiert werden, ausgetauscht.

[0006] Der Erfindung liegt daher die Aufgabe zugrunde, ein Verfahren zur Übertragung von Daten anzugeben, welches einen schnellen und sicheren Austausch vergleichsweise großer Datenmengen über das Internet erlaubt.

[0007] Diese Aufgabe wird erfindungsgemäß dadurch gelöst, dass dem jeweiligen Chunk eine über den Chunk gebildete Prüfsumme zugeordnet wird, und dass das Schreiben des jeweiligen Chunks in den jeweiligen Stream durch den jeweiligen Sender nur auf eine Bestätigung des jeweiligen Empfängers, dass der jeweilige Chunk auf dem jeweiligen Empfänger noch nicht vorhanden ist, auf Übersendung der dem Chunk zugeordneten Prüfsumme erfolgt.

[0008] Die Erfindung geht dabei von der Überlegung aus, ein schneller und sicherer Datenaustausch großer Datenmengen über das Internet möglich wäre, wenn einerseits die Anfälligkeit für Fehler bei der Übertragung und andererseits die Abhängigkeit von der Übertragungsgeschwindigkeit von Sender und Empfänger entkoppelt werden könnte. Dies ist möglich, indem die Daten durch den ersten Client, d. h. den Sender in Chunks geringerer Größe zerlegt werden. Weiterhin wird ein Server als Mittler eingesetzt, der selbstständig Daten vom Sender empfangen und an den Empfänger, den zweiten Client senden kann. Der Server agiert damit aus Sicht des Empfängers als Sender und aus Sicht des Senders als Empfänger. Die Chunks werden vom Sender in einen Stream geschrieben, an den Server geschickt und dort gespeichert. Der Sender kann somit bei ausreichend großer Anbindungsgeschwindigkeit des Servers an das Internet mit seiner vollen Bandbreite übertragen, auch wenn dem eigentlichen Empfänger die Bandbreite nicht zur Verfügung steht. Weiterhin wird durch die Pufferung im Server die Zeit von Versand und Empfang weitgehend entkoppelt. Der Empfänger kann bereits mit dem Empfang des Streams vom Server beginnen, wenn die ersten Chunks auf dem Server eingetroffen sind, kann diese jedoch auch später empfangen. Damit sind Versand und Empfang sowohl simultan als auch konsekutiv möglich.

[0009] Dabei wird dem jeweiligen Chunk eine über den Chunk gebildete Prüfsumme zugeordnet. Damit wird erreicht, dass eine Fehlerkorrektur im Stream sowohl auf Sender- als auch auf Empfängerseite möglich ist. Die Prüfsumme kann von dem sendenden Client gebildet und an den Server übertragen werden. Der Server bildet seinerseits eine Prüfsumme und gleich diese mit der vom Client übertragenen ab. Wird eine Abweichung festgestellt, wird der Chunk erneut übertragen. Gleiches gilt analog für die Übertragung der Daten vom Server zum empfangenden Client. Da die Datenmenge in Chunks zerlegt ist, wird bei einem Fehler nicht die Neuübertragung der gesamten Datenmenge notwendig, sondern lediglich die des jeweiligen Chunks.

[0010] Durch eine Abfrage anhand von Prüfsummen wird eine Reduzierung der im Stream übertragenen Datenmenge sowohl zwischen Sender und Server als auch zwischen Server und Empfänger mittels Deduplikation erreicht, was eine erhebliche Erhöhung der

Übertragungsgeschwindigkeit zur Folge hat. Die Bestätigung durch den Server bzw. den empfangenden Client erfolgt in diesem Fall nur, wenn der jeweilige Chunk auf dem Server bzw. empfangenden Client noch nicht vorhanden ist. Ein Fall für eine erhebliche Beschleunigung der Übertragung ist dabei beispielsweise die Übertragung eines Updates der Daten. Hier wird nach der Zerlegung der Daten in Chunks zunächst die Prüfsumme gebildet und übertragen. Der Server kann mittels der Prüfsumme prüfen, ob der jeweilige Chunk durch das Update verändert wurde oder ob er identisch bereits im Speicher vorhanden ist. Im letzteren Fall wird die Übertragung nicht bestätigt. Die Datenmenge wird reduziert. Gleiches gilt für die Übertragung vom Server zum empfangenden Client.

[0011] In vorteilhafter Ausgestaltung wird dem jeweiligen Stream dabei ein Identifikationselement zugeordnet. Dadurch ist der jeweilige Stream eindeutig und potenziell weltweit eindeutig identifizierbar und erreichbar. Dies ermöglicht auch eine einfache Verwaltung der Streams für den Fall, dass der Server eine Mehrzahl von Datenübertragungen zwischen unterschiedlichen Clients vermittelt.

[0012] In alternativer oder zusätzlicher vorteilhafter Ausgestaltung wird dem jeweiligen Chunk ein Identifikationselement und/oder ein Metadatenelement zugeordnet. Die Zuordnung eines Identifikationselements und/oder Metadatenelements ermöglicht es dem jeweiligen Empfänger, Informationen über den jeweiligen Chunk zu erhalten, ohne die Daten des Chunks selbst empfangen zu müssen. Dies gilt insbesondere bei einer Vergabe von derartigen Elementen auf dem Server. Dies ermöglicht es insbesondere serverseitig auch, bestehende Streams zu konkatenieren, um neue Streams zu erzeugen. Diese Operation ist sowohl auf der Ebene der Metadaten bzw. Identifikationselemente, als auch der Ebene der eigentlichen Chunkdaten möglich.

[0013] Zur selektiven Übertragung von Chunks erfolgt vorteilhafterweise erfolgt das Schreiben des jeweiligen Chunks in den jeweiligen Stream durch den jeweiligen Sender nur auf eine Bestätigung des jeweiligen Empfängers auf Übersendung des dem Chunk zugeordneten Identifikationselements bzw. des dem Chunk zugeordneten Metadatenelements. Durch eine Abfrage anhand von Metadaten oder Identifikationselementen kann der jeweilige Empfänger somit selektiv Chunks empfangen, beispielsweise aufgrund von Metadaten, die den Inhalt des Chunks betreffen, oder aufgrund eine zeitbasierten Indexierung oder logischen Indexierung mit eindeutigen Identifikationselementen.

[0014] Vorteilhafterweise erfolgt die Übersendung der Prüfsumme bzw. des Identifikationselements bzw. des Metadatenelements und/oder der Bestätigung in einer von dem jeweiligen Stream separierten Kommunikationsverbindung. Grundsätzlich ist es möglich, sämtliche Daten, d. h. sowohl die Chunks selbst als auch die Prüfsummen, Identifikationselemente und Metadatenelemente über eine Datenverbindung, d. h. in demselben Stream zu übertragen. Eine weitere Beschleunigung der Datenübertragung wird aber möglich, wenn diese beiden Kommunikationskanäle entkoppelt werden, d. h. der Stream der Chunks nicht mit den Prüfsummen, Metadaten und Identifikationselementen vermischt wird, sondern diese in einer separaten Kommunikationsverbindung ausgetauscht werden. Die separate Kommunikationsverbindung kann hierbei – je nach Größe der hier zu übertragenden Daten – auch direkt zwischen Sender und Empfänger aufgebaut werden.

[0015] In weiterer vorteilhafter Ausgestaltung wird der jeweilige Chunk mittels eines kryptographischen Schlüssels verschlüsselt. Dies erhöht die Sicherheit der Datenübertragung gerade bei sensiblen Daten und ermöglicht eine Identifizierung von Sender und Empfänger. Dabei kann insbesondere ein symmetrischer Schlüssel, der sowohl zur Ver- als auch zur Entschlüsselung dient, verwendet werden.

[0016] In vorteilhafter Ausgestaltung bildet dabei der zweite Client eine Prüfsumme über den jeweiligen Chunk und sendet diese an den Server. Bei Übereinstimmung der empfangenen Prüfsumme mit einer vom Server ermittelten Prüfsumme über den jeweiligen Chunk wird dann der kryptographische Schlüssel an den zweiten Client gesendet. Es ist denkbar, dass hierbei nur Teile eines Chunks oder eine Kollektion von Chunks oder Teilen von Chunks verwendet werden, z. B. über ein randomisiertes Verfahren: Eine Anzahl von Chungs wird übertragen und randomisierte Bytes ausgewählt, um eine Prüfsumme darüber zu bilden. Durch ein derartiges Verfahren wird eine Datenübertragung in der Art eines Einschreibens ermöglicht. Die Bildung einer Prüfsumme über den verschlüsselten Chunk ermöglicht es dem Server bei einem Abgleich zu ermitten, ob die Daten vollständig beim empfangenden Client angekommen sind, noch bevor der empfangende Client die Daten öffnen und verändern kann. Erst bei einer Übereinstimmung der Prüfsumme, die einen vollständigen Empfang bestätigt, wird der kryptographische Schlüssel übertragen. Alternativ wäre eine zeitgesteuerte Übermittlung der Entschlüsselungsinformationen in der Art eines Zeitschlosses möglich.

[0017] In weiterer vorteilhafter Ausgestaltung wird der kryptographische Schlüssel anhand eines vordefinierten Verfahrens aus dem Inhalt des jeweiligen Chunks und/oder eines dem Chunk zugeordneten Metadatenelements ermittelt. Bespielsweise könnte die Prüfsumme über den unverschlüsselten Chunk, eine davon abgeleitete Größe oder eine zum Chunk verfasste Inhaltsangabe als Schlüssel dienen. Dies

garantiert, dass eine nach der Verschlüsselung gebildete Prüfsumme über zwei identische Chunks ebenfalls identisch bleibt und ermöglicht somit eine Deduplikation trotz Verschlüsselung mit einem pro Chunk individuellen Schlüssel.

[0018] Ein Computerprogrammprodukt, umfasst vorteilhafterweise Softwarecodeabschnitte, die, wenn sie auf einem Endgerät in einem Netzwerk ausgeführt werden, das Endgerät zum Zusammenwirken als Client oder Server in dem beschriebenen Verfahren ertüchtigen.

[0019] Ein Server in einem Netzwerk, umfasst vorteilhafterweise einen Speicher, in den ein derartiges Computerprogrammprodukt geladen ist.

[0020] Die mit der Erfindung erzielten Vorteile bestehen insbesondere darin, dass durch die Zerlegung großer Datenmengen in Chunks und Übermittlung durch einen vermittelnden Server eine besonders schnelle und sichere Datenübertragung im Internet in der Art einer Datenlogistik möglich wird. Insbesondere durch die Verwendung von Prüfsummen je Chunk ist es möglich, die Austauschprozesse um bis zu 80% zu beschleunigen.

[0021] Die Erfindung wird anhand einer Zeichnung näher erläutert. Darin zeigen:

[0022] **Fig.** 1 den Aufbau eines Netzwerks mit einer Mehrzahl von Clients,

[0023] **Fig.** 2 zwei Kommunikationsverbindungen zwischen einem Client und einem Server mit dem schematisch dargestellten Aufbau eines Streams vom Client zum Server,

[0024] **Fig.** 3 eine Kommunikationsverbindung zwischen Client und einem Server mit dem schematisch dargestellten Aufbau eines Streams vom Server zum Client

[0025] Die **Fig.** 1 zeigt schematisch den Aufbau eines Netzwerks **1**, nämlich des Internets, mit einem Server **2** und mehreren Clients **4**, **6**, **8**. Selbstverständlich sind eine Vielzahl weiterer Rechner im Netzwerk **1** verbunden, von denen nur die genannten dargestellt sind. In die Speicher des Servers **2** und der Clients **4**, **6**, **8** ist ein Computerprogramm geladen, welches Softwarecodeabschnitte umfasst, die den Server **2** bzw. die Clients **4**, **6**, **8** zum Ausführen der im folgenden beschriebenen Verfahren ertüchtigen.

[0026] Das im folgenden beschriebene Verfahren dient dazu eine große Datenmenge zwischen zwei Clients **4**, **6**, **8** auszutauschen. Die Datenmenge liegt im Terabyte-Bereich. Hierzu werden Kommunikationsverbindungen **10** zwischen dem Server **2** und jeweils einem Client **4**, **6**, **8** aufgebaut, wobei vom Server **2** zu einem gegebenen Client **4**, **6**, **8** auch mehrere Kommunikationsverbindungen bestehen können.

[0027] In der **Fig.** 2 und der **Fig.** 3 ist exemplarisch eine Datenübertragung vom Client **4** an den Client **6** dargestellt. **Fig.** 2 zeigt die entsprechenden Kommunikationsverbindungen **10**, **10a**. Der sendende Client **4**, **6**, **8** zerlegt die Daten **12** zunächst in Chunks **14**, **16**, **18**, wobei beliebig viele Chunks **14**, **16**, **18** einer kleineren Größe erzeugt werden können. Im Ausführungsbeispiel der **Fig.** 2 sind nur drei Chunks **14**, **16**, **18** dargestellt. Über jeden der Chunks **14**, **16**, **18** bildet der Client **4** eine Prüfsumme **20**, **22**, **24**, die dem jeweiligen Chunk **14**, **16**, **18** zugeordnet wird.

[0028] Auf der ersten Kommunikationsverbindung **10** überträgt der Client **4** an den Server **2** zunächst die erste Prüfsumme **20**. Der Server **2** prüft die in ihm gespeicherten Daten und stellt fest, dass die Prüfsumme **20** und damit der ihr zugeordnete Chunk **14** noch nicht vorhanden sind. Daraufhin sendet der Server **2** auf der Kommunikationsverbindung **10** an den Client **4** eine Bestätigung, woraufhin der Client **4** den Chunk **14** in einen Stream schreibt, der auf der zweiten Kommunikationsverbindung **10a** an den Server **2** gesendet wird.

[0029] Sodann sendet der Client **4** auf der ersten Kommunikationsverbindung **10** die nächste Prüfsumme **22** an den Server **2**. Der Server **2** stellt fest, dass die Prüfsumme **22** und der ihr zugeordnete Chunk **16** bereits vorhanden sind und sendet keine Bestätigung über die Kommunikationsverbindung **10** oder eine entsprechende negative Nachricht. Der Client **4** schreibt den Chunk **16** entsprechend nicht in den Stream auf der Kommunikationsverbindung **10a**. Analog wird mit dem Chunk **18** und gegebenenfalls weiteren Chunks verfahren. Alternativ kann der Client **4** auch zunächst eine Mehrzahl von Prüfsummen **20**, **22**, **24** senden, woraufhin der Server **2** gesammelt die Prüfsummen **20**, **22**, **24** mit dem in ihm vorhandenen Datensatz abgleicht und entsprechend eine nur auf die Prüfsumme **20** und den ihr zugeordneten Chunk **16** gerichtete Bestätigung an den Client **4** sendet.

[0030] Bei der Übertragung vom Client **4** an den Server **2** können auch weitere, im folgenden anhand **Fig.** 3 beschriebene Möglichkeiten der Bestätigung des Übertragens von Chunks zur Anwendung kommen, so z. B. anhand von Identifikationselementen und Metadaten. Weiterhin kann auch nur eine Kommunikationsverbindung **10** verwendet werden.

[0031] **Fig.** 3 zeigt eine Kommunikationsverbindung **10** vom Server **2** an den Client **6**. Die vom Client **4** empfangenen Daten **12** sind im Server **2** abgelegt. Der für den Client **6** bestimmte Stream wird mit einem eindeutigen Identifikationselement, z. B, einer

eindeutigen URL versehen, so dass er für den Client **6** per HTTP weltweit erreichbar ist.

[0032] Im Ausführungsbeispiel nach der **Fig.** 3 erfolgt ebenfalls eine Prüfung auf Vorhandensein von Chunks **14**, **16**, **18** im Client **6** anhand einer Prüfsumme **20**, **22**, **24**, analog zu **Fig.** 2. Allerdings hat der Client **6** im Ausführungsbeispiel nach **Fig.** 3 noch keine Daten, so dass jeder Chunk **14**, **16**, **18** übertragen werden müsste.

[0033] Anhand von **Fig.** 3 wird jedoch eine weitere Möglichkeit der Datenselektion aufgezeigt. Der Server **2** ordnet dem jeweiligen Chunk **14**, **16**, **18** Metadatenelemente **26**, **28**, **30** und Identifikationselemente **32**, **34**, **36** zu. Die Metadatenelemente **26**, **28**, **30** können beispielsweise stichwortartig zusammengefasste Informationen zum Inhalt des jeweiligen Chunks **14**, **16**, **18** umfassen. Die Identifikationselemente **32**, **34**, **36** können eindeutige Zahlenfolgen sein, die einen Chunk **14**, **16**, **18** eindeutig kennzeichnen, oder aber zeit-basiert sein, z. B. den Zeitpunkt des Versandes im Stream angeben.

[0034] Auf der Kommunikationsverbindung **10** überträgt der Server **2** an den Client **6** zunächst das erste Metadatenelement **26** und das erste Identifikationselement **32**. Der Client **6** prüft anhand vorgebbarer Kriterien, ob der Chunk zu den Metadaten im Metadatenelement **26** erwünscht ist. Bei positivem Resultat sendet der Client **6** auf der Kommunikationsverbindung **10** an den Server **2** eine Bestätigung, woraufhin der Server **2** den Chunk **14** in einen Stream schreibt, der auf der Kommunikationsverbindung **10** an den Client **6** gesendet wird. Bedarfsweise kann der Server **2** auch weitere Daten in den Stream an Client **6** einbauen, d. h. den Stream mit einem anderen Stream konkatenieren.

[0035] Sowohl in der **Fig.** 2 als auch in der **Fig.** 3 werden die übertragenen Chunks **14**, **16**, **18** anhand der Prüfsummen **20**, **22**, **24** beim jeweiligen Empfänger auf korrekte Übertragung geprüft. Stimmt die vom jeweiligen Empfänger über den jeweiligen Chunk **14**, **16**, **18** gebildete Prüfsumme **20**, **22**, **24** nicht mit der zuvor übertragenen Prüfsumme **20**, **22**, **24** überein, wird eine erneute Übertragung durch erneutes Schreiben des jeweiligen Chunks **14**, **16**, **18** in den Stream veranlasst.

[0036] Auch im Beispiel nach **Fig.** 3 können zwei Kommunikationsverbindungen **10**, **10a** zur Trennung des Streams von den übrigen Signalen zur Anwendung kommen, ebenso wie die Verwendung von Prüfsummen **20**, **22**, **24** bei der Übertragung. Wird der Stream sowohl vom Client **4** zum Server **2** als auch vom Server **2** zum Client **6** anhand einer vorherigen Überprüfung mit Prüfsummen **20**, **22**, **24** erstellt, ergibt sich ein hocheffektives, schnelles und fehlerunanfälliges System zur Übertragung großer Datenmengen.

[0037] Weiterhin werden die Chunks **14**, **16**, **18** bedarfsweise mittels eines symmetrischen Schlüssels verschlüsselt. Hierfür werden im folgenden zwei Ausführungsbeispiele anhand der **Fig.** 2 und der **Fig.** 3 erläutert.

[0038] Die Verschlüsselung kann in **Fig.** 2 durch den Client **4** chunkweise erfolgen. Dabei wird der verwendete Schlüssel des jeweiligen Chunks **14**, **16**, **18** anhand einer Prüfsumme über den Inhalt gebildet. Hier können auch komplexere Algorithmen zur Bildung des Schlüssels zur Anwendung kommen, solange das Verfahren deterministisch ist, d. h. ein Chunk **14**, **16**, **18** immer mit dem gleichen Schlüssel verschlüsselt wird. Wird dann eine Prüfsumme **20**, **22**, **24** über den verschlüsselten Chunk **14**, **16**, **18** gebildet, kann das Verfahren aus der **Fig.** 2 weiter angewendet werden, da die Prüfsummen **20**, **22**, **24** auch für verschlüsselte Chunks **14**, **16**, **18** eindeutig und reproduzierbar sind.

[0039] Die vom Client **4** für jeden Chunk **14**, **16**, **18** erzeugten Schlüssel können dann ebenfalls an den Server **2** gesendet werden oder anderweitig hinterlegt werden, z. B. an einen nicht näher gezeigten weiteren Server. In **Fig.** 3 ist ebenfalls eine derartige Verschlüsselung vom Server **2** zum Client **6** möglich.

[0040] Zusätzlich kann in **Fig.** 3 die Verschlüsselung dazu dienen, eine Datenübertragung in der Art eines Einschreibens durchzuführen. Hierzu werden die in den Stream an den Client **6** geschriebenen Chunks **14**, **16**, **18** vom Server **2** verschlüsselt. Nach dem Empfang des jeweiligen Chunks **14**, **16**, **18** bildet der Client **6** jeweils die Prüfsumme **20**, **22**, **24** über den verschlüsselten Chunk **14**, **16**, **18** und sendet diese über die Kommunikationsverbindung an den Server **2**. Der Server **2** gleicht die vom Client **6** übermittelten Prüfsummen **20**, **22**, **24** mit den eigenen über die jeweiligen Chunks **14**, **16**, **18** ermittelten Prüfsummen **20**, **22**, **24**. Wenn alle Prüfsummen **20**, **22**, **24** übereinstimmen, ist dies für den Server **2** die Bestätigung, dass die Daten **12** vollständig vom Client **6** empfangen wurden. Erst dann übermittelt der Server **2** den Schlüssel für die Chunks **14**, **16**, **18** an den Client **6**, **50** dass der Client **6** die Daten entschlüsseln kann.

Bezugszeichenliste

1	Netzwerk
2	Server
4, 6, 8	Client
10, 10a	Kommunikationsverbindung
12	Daten

14, 16, 18	Chunk
20, 22, 24	Prüfsumme
26, 28, 30	Metadatenelement
32, 34, 36	Identifikationselement

Patentansprüche

1. Verfahren zur Übertragung von Daten (**12**) von einem ersten Client (**4**, **6**, **8**) zu einem zweiten Client (**4**, **6**, **8**) in einem Netzwerk (**1**), umfassend die Verfahrensschritte:
– Zerlegen der Daten (**12**) in Chunks (**14**, **16**, **18**) durch den ersten Client (**4**, **6**, **8**),
– Schreiben einer Anzahl der Chunks (**14**, **16**, **18**) in einen ersten Stream durch den ersten Client (**4**, **6**, **8**),
– Senden des ersten Streams von dem ersten Client (**4**, **6**, **8**) an einen Server (**2**),
– Lesen und Speichern der Chunks (**14**, **16**, **18**) aus dem ersten Stream auf dem Server (**2**),
– Schreiben einer Anzahl der Chunks (**14**, **16**, **18**) in einen zweiten Stream durch den Server (**2**),
– Senden des zweiten Streams von dem Server an den zweiten Client (**4**, **6**, **8**),
dadurch gekennzeichnet, dass
dem jeweiligen Chunk (**14**, **16**, **18**) eine über den Chunk (**14**, **16**, **18**) gebildete Prüfsumme (**20**, **22**, **24**) zugeordnet wird, und dass
das Schreiben des jeweiligen Chunks (**14**, **16**, **18**) in den jeweiligen Stream durch den jeweiligen Sender (**2**, **4**, **6**, **8**) nur auf eine Bestätigung des jeweiligen Empfängers (**2**, **4**, **6**, **8**), dass der jeweilige Chunk auf dem jeweiligen Empfänger (**2**, **4**, **6**, **8**) noch nicht vorhanden ist, auf Übersendung der dem Chunk (**14**, **16**, **18**) zugeordneten Prüfsumme (**20**, **22**, **24**) erfolgt.

2. Verfahren nach Anspruch 1, bei dem dem jeweiligen Stream ein Identifikationselement zugeordnet wird.

3. Verfahren nach einem der vorhergehenden Ansprüche, bei dem dem jeweiligen Chunk (**14**, **16**, **18**) ein Identifikationselement (**32**, **34**, **36**) und/oder ein Metadatenelement (**26**, **28**, **30**) zugeordnet wird.

4. Verfahren nach Anspruch 3, bei dem das Schreiben des jeweiligen Chunks (**14**, **16**, **18**) in den jeweiligen Stream durch den jeweiligen Sender (**2**, **4**, **6**, **8**) nur auf eine Bestätigung des jeweiligen Empfängers (**2**, **4**, **6**, **8**) auf Übersendung des dem Chunk (**14**, **16**, **18**) zugeordneten Identifikationselements (**32**, **34**, **36**) und/oder des dem Chunk (**14**, **16**, **18**) zugeordneten Metadatenelements (**26**, **28**, **30**) erfolgt.

5. Verfahren nach Anspruch 4, bei dem die Übersendung der Prüfsumme (**20**, **22**, **24**) und/oder des Identifikationselements (**32**, **34**, **36**) und/oder des Metadatenelements (**26**, **28**, **30**) und/oder der Bestätigung in einer von dem jeweiligen Stream separierten Kommunikationsverbindung (**10a**) erfolgt.

6. Verfahren nach einem der vorhergehenden Ansprüche, bei dem der jeweilige Chunk (**14**, **16**, **18**) mittels eines kryptographischen Schlüssels verschlüsselt wird.

7. Verfahren nach Anspruch 6, bei dem der zweite Client (**4**, **6**, **8**) eine Prüfsumme (**20**, **22**, **24**) über den jeweiligen Chunk (**14**, **16**, **18**) bildet und an den Server (**2**) sendet, und bei dem der Server (**2**) bei Übereinstimmung der empfangenen Prüfsumme (**20**, **22**, **24**) mit einer vom Server (**2**) ermittelten Prüfsumme (**20**, **22**, **24**) über den jeweiligen Chunk (**14**, **16**, **18**) den kryptographischen Schlüssel an den zweiten Client (**4**, **6**, **8**) sendet.

8. Verfahren nach Anspruch 6 oder 7, bei dem der kryptographische Schlüssel anhand eines vordefinierten Verfahrens aus dem Inhalt des jeweiligen Chunks (**14**, **16**, **18**) und/oder eines dem Chunk (**14**, **16**, **18**) zugeordneten Metadatenelements (**26**, **28**, **30**) ermittelt wird.

9. Computerprogrammprodukt, welches Softwarecodeabschnitte umfasst, die, wenn sie auf einem Endgerät in einem Netzwerk (**1**) ausgeführt werden, das Endgerät zum Zusammenwirken als Client (**4**, **6**, **8**) in dem Verfahren nach einem der vorhergehenden Ansprüche ertüchtigen.

10. Computerprogrammprodukt, welches Softwarecodeabschnitte umfasst, die, wenn sie auf einem Endgerät in einem Netzwerk (**1**) ausgeführt werden, das Endgerät zum Zusammenwirken als Server (**2**) in dem Verfahren nach einem der Ansprüche 1 bis 8 ertüchtigen.

11. Server (**2**) in einem Netzwerk (**1**), in dessen Speicher ein Computerprogrammprodukt nach Anspruch 10 geladen ist.

Es folgt eine Seite Zeichnungen

Anhängende Zeichnungen

FIG. 1

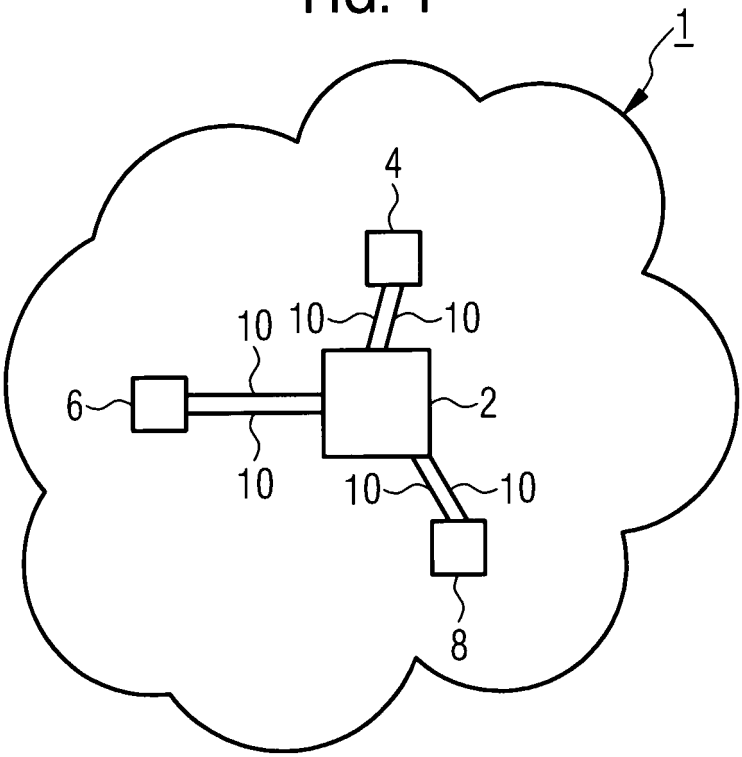


FIG. 2

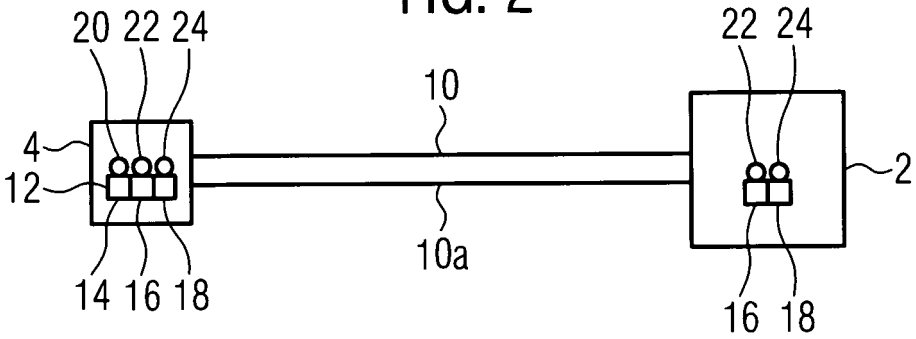


FIG. 3