(19) Deutsches Patent- und Markenamt

(10) **DE 11 2012 000 744 T5** 2014.01.30

(12) # Veröffentlichung

der internationalen Anmeldung mit der
(87) Veröffentlichungs-Nr.: **WO 2012/107255**
in deutscher Übersetzung (Art. III § 8 Abs. 2 IntPatÜG)
(21) Deutsches Aktenzeichen: **11 2012 000 744.1**
(86) PCT-Aktenzeichen: **PCT/EP2012/050304**
(86) PCT-Anmeldetag: **10.01.2012**
(87) PCT-Veröffentlichungstag: **16.08.2012**
(43) Veröffentlichungstag der PCT Anmeldung
in deutscher Übersetzung: **30.01.2014**

(51) Int Cl.: ***G06F 21/00*** (2013.01)

(30) Unionspriorität:
12/931,855 **11.02.2011** **US**

(71) Anmelder:
F-Secure Corp., Helsinki, FI

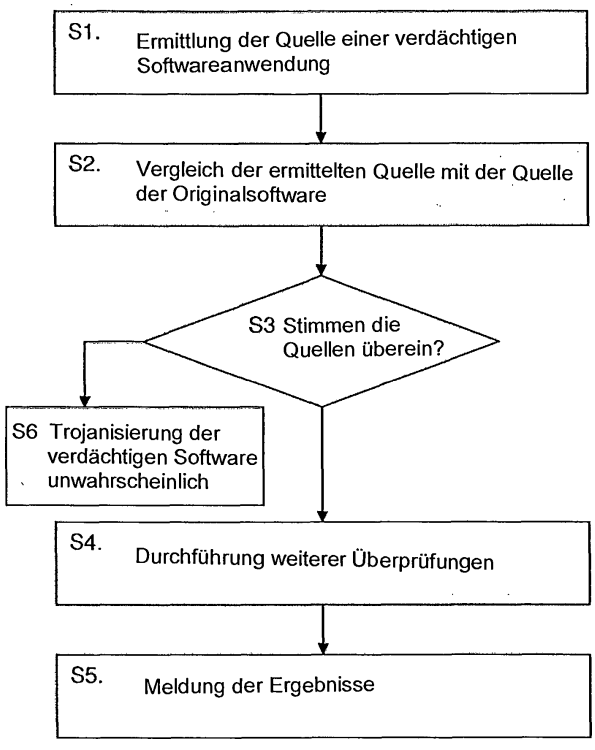
(74) Vertreter:
Gulde Hengelhaupt Ziebig & Schneider, 10179, Berlin, DE

(72) Erfinder:
Stahlberg, Mika, Helsinki, FI; Niemelä, Jarno, Helsinki, FI; Kasslin, Kimmo, Helsinki, FI

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Erkennung eines trojanischen Pferdes**

(57) Zusammenfassung: Verfahren und Vorrichtung zur Erkennung eines Trojanischen Pferdes in einer verdächtigen Softwareanwendung in Form von mindestens einer elektronischen Datei. Ein Computergerät ermittelt die Quelle, aus der eine verdächtige Softwareanwendung bezogen wurde. Anschließend wird ein Vergleich zwischen der Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, und einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, durchgeführt. Falls sich die Quellen unterscheiden, wird festgestellt, dass die verdächtige Anwendung mit höherer Wahrscheinlichkeit ein Trojanisches Pferd enthält als im Fall der Identität der Quellen.



Beschreibung

TECHNISCHES GEBIET

[0001] Die Erfindung betrifft das Gebiet der Erkennung eines Trojanischen Pferdes.

STAND DER TECHNIK

[0002] Infektionen von Computern und Computersystemen mit Schadprogrammen sind ein zunehmendes Problem. Es gab zahlreiche prominente Beispiele von Computerschädlingen, die sich sehr schnell rund um die Welt verbreiteten und durch Datenverluste und Arbeitsausfälle Schäden in Millionenhöhe verursachten.

[0003] Schadprogramme verbreiten sich häufig durch Computerviren. Frühe Computerviren verbreiteten sich durch das Kopieren infizierter elektronischer Dateien auf Disketten und die Übertragung der elektronischen Datei von der Diskette auf einen noch nicht infizierten Computer. Sobald der Benutzer versucht, die infizierte elektronische Datei zu öffnen, wird das Schadprogramm aktiviert und der Computer infiziert. In neuerer Zeit wurden Viren über das Internet verbreitet, beispielsweise durch E-Mails. Es sind auch Viren bekannt, die durch drahtlose Datenübertragung verbreitet werden, beispielsweise bei der Kommunikation zwischen mobilen Kommunikationsgeräten in einem Mobilfunknetz.

[0004] Auf dem Markt sind heute verschiedene Antiviren-Anwendungen erhältlich. Diese arbeiten meist auf der Grundlage einer Datenbank von Signaturen oder Fingerabdrücken bekannter Viren und Schadprogramme. Bei einer Anwendung mit ”Echtzeit”-Virenscan wird, sobald der Benutzer versucht, eine Operation wie Öffnen, Speichern oder Kopieren auf eine Datei anzuwenden, die Anfrage an die Antiviren-Anwendung weitergeleitet. Wenn der Anwendung die elektronische Datei noch nicht bekannt ist, wird die elektronische Datei nach bekannten Signaturen von Viren oder Schadprogrammen durchsucht. Wenn in einer Datei ein Virus oder Schadprogramm identifiziert wurde, informiert die Antiviren-Anwendung den Benutzer darüber, beispielsweise durch das Anzeigen einer Meldung in einem Popup-Fenster. Anschließend kann die Antiviren-Anwendung die Identität der infizierten Datei in ein Verzeichnis infizierter Dateien aufnehmen.

[0005] In den letzten Jahren sind sogenannte ”Application Stores” bei mobilen Anwendern sehr populär geworden. Die bekanntesten Beispiele sind wahrscheinlich der Apple App Store und der Android Market. Ein Application Store ist ein Onlinedienst, der es Benutzern ermöglicht, Softwareanwendungen zu durchsuchen und von einem Remote-Server auf ihr Gerät herunterzuladen. Häufig sind die Anwendungen kostenlos oder sehr kostengünstig, und erfolgreiche Anwendungen werden auf Millionen von Geräten heruntergeladen.

[0006] Obwohl dieselbe Softwareanwendung möglicherweise auch aus einer anderen Quelle erhältlich ist, führt die Bequemlichkeit eines Application Stores dazu, dass die große Mehrheit der Downloads einer Softwareanwendung über einen App Store erfolgt.

[0007] Durch das entstandene Paradigma der Application Stores ist es für Verteiler von Schadprogrammen zunehmend schwieriger geworden, Methoden wie Spam oder Vergiftung von Suchergebnissen durch Suchmaschinenoptimierung (SEO) zu verwenden, um ihre Opfer durch Täuschung zum Installieren bösartiger Anwendungen zu bringen. Es wird also zum erfolgversprechendsten Angriffsvektor, Schadprogramme in einem Application Store zur Verfügung zu stellen. Eine einfache Methode, um Benutzer zum Installieren von Schadprogrammen zu bringen, ist es, das Schadprogramm in Form einer Anwendung anzubieten, die eine für den Benutzer wünschenswerte Funktionalität bereitzustellen scheint. Dieser Typ von Schadprogrammen wird als Trojanisches Pferd oder Trojaner bezeichnet. Ein Trojanisches Pferd scheint eine für den Benutzer wünschenswerte Funktionalität bereitzustellen, enthält aber bösartigen Code. Der bösartige Code kann zusätzlich zur Bereitstellung der gewünschten Funktionalität ausgeführt werden, so dass der Benutzer nicht erkennt, dass auf seinem Computergerät ein Trojanisches Pferd läuft. Ein Trojanisches Pferd kann beispielsweise dazu verwendet werden, unerwünschte Werbung anzuzeigen oder einem bösartigen Dritten den Zugriff auf das Computergerät zu gewähren und ihm das Ausführen unerwünschter Operationen zu ermöglichen, zum Beispiel Anwahl von Mehrwertdienstnummern, Datendiebstahl, Installation unerwünschter Software, Veränderung oder Löschung existierender Dateien und ähnliches.

[0008] Ein Weg zur Erzeugung eines Trojanischen Pferdes ist es, eine existierende Anwendung zu verwenden und sie so zu modifizieren, dass sie zusätzlich eine bösartige Funktionalität erhält. Dieser Vorgang wird gelegentlich als ”Trojanisierung” einer Anwendung bezeichnet. Ein Beispiel einer Trojanisierung ist die Trojanerfamilie Geinimi, die Ende 2010 auf der Android-Plattform aktiv wurde. Eine Beschreibung von Geinimi findet

sich unter http://www.theregister.co.uk/2010/12/31/china_android_trojan/. Geinimi ist ein Schadprogramm für Mobilgeräte, das sich als eine Spieleanwendung ausgibt. Sobald ein Benutzer einen Geinimi-Trojaner installiert hat, kann Geinimi persönliche Daten vom Gerät des Benutzers an einen Remote-Server senden. Außerdem kann Geinimi Befehle von einem fremden Dritten empfangen.

[0009] Trojanische Pferde können erkannt werden, indem der Code einer Softwareanwendung hinsichtlich einer eventuell enthaltenen bösartigen Funktionalität analysiert wird, oder durch Emulation der Softwareanwendung, um die Ausführung etwaiger unerwünschter Operationen festzustellen. Für beide Ansätze kann ein gewisser Aufwand an Zeit und Ressourcen nötig sein.

ZUSAMMENFASSUNG

[0010] Eine Aufgabe der vorliegenden Erfindung besteht in der Bereitstellung einer Möglichkeit, ein Trojanisches Pferd in einer Softwareanwendung zu erkennen, ohne dabei sofort auf detaillierte Codeanalyse oder Emulation zurückzugreifen. Gemäß einem ersten Aspekt der Erfindung wird ein Verfahren zur Erkennung eines Trojaners in einer verdächtigen Softwareanwendung in Form von mindestens einer elektronischen Datei bereitgestellt. Ein Computergerät ermittelt die Quelle, aus der eine verdächtige Softwareanwendung bezogen wurde. Anschließend wird ein Vergleich zwischen der Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, und einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, durchgeführt. Wenn sich die Quellen unterscheiden, wird festgestellt, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält.

[0011] Optional umfassen die Quellen eine Identität eines Verkäufers.

[0012] Das Verfahren umfasst optional ferner das Durchführen eines Vergleichs der verdächtigen Version der Softwareanwendung mit der sauberen Version der Softwareanwendung. Verglichen werden dabei die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte.

[0013] Optional werden die Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weitere auf die saubere Version der Softwareanwendung bezogene Metadaten in einer Datenbank gespeichert.

[0014] Die Erfindung kann optional unter Verwendung eines Backend-Servers implementiert werden. In diesem Fall umfasst das Verfahren optional das Senden einer Nachricht vom Computergerät an einen Remote-Server, wobei die Nachricht die Bestandteile der verdächtigen Anwendung und/oder die auf die verdächtige Anwendung bezogenen Metadaten enthält, die der Server benötigt, um den Vergleich durchzuführen. In diesem Fall kann das Verfahren optional auf dem Gerät das Empfangen einer Antwortnachricht vom Server umfassen. Die Antwortnachricht enthält einen Hinweis, dass die verdächtige Softwareanwendung wahrscheinlich ein Trojanisches Pferd enthält.

[0015] Das Verfahren umfasst optional das Senden einer Nachricht von dem Gerät an eine Remote-Datenbank, wobei die Nachricht mindestens eine Identität der verdächtigen Softwareanwendung enthält. Das Gerät empfängt dann eine Antwort von der Remote-Datenbank, wobei die Antwort die Quelle enthält, aus der die saubere Originalversion der Softwareanwendung bezogen wurde.

[0016] Gemäß einem zweiten Aspekt wird ein Gerät zur Verwendung in einem Kommunikationsnetzwerk bereitgestellt. Das Gerät ist mit einem Prozessor ausgestattet, der zur Feststellung dient, dass eine Softwareanwendung in Form von mindestens einer elektronischen Datei verdächtig ist. Der Prozessor ist ferner so konfiguriert, dass er die Quelle der Softwareanwendung feststellen kann, und ferner so konfiguriert, dass er die Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, vergleichen kann. Falls sich die Quellen unterscheiden, stellt der Prozessor fest, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält.

[0017] Optional umfasst das Gerät ferner einen Sender, der dazu dient, entweder an einen Server oder an eine Datenbank eine Anforderungsnachricht zu senden. Die Anforderungsnachricht enthält eine Anforderung nach der Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde. Außerdem wird ein Empfänger bereitgestellt, der dazu dient, eine Antwort zu empfangen, wobei die Antwort die Quelle enthält, aus der die saubere Originalversion der Softwareanwendung bezogen wurde. Diese Ausgestaltung ermöglicht

es dem Gerät, eine Datenbank abzufragen, die dem Gerät ausreichende Informationen zur Verfügung stellt, um die Feststellung vorzunehmen.

[0018] Der Prozessor ist optional ferner so konfiguriert, dass er die verdächtige Version der Softwareanwendung mit der sauberen Version der Softwareanwendung in Bezug auf die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte, vergleichen kann.

[0019] In einer alternativen Ausgestaltung umfasst das Gerät eine Datenbank zum Speichern von Daten bezüglich der Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weiterer auf die saubere Version der Softwareanwendung bezogener Metadaten.

[0020] Gemäß einem dritten Aspekt wird ein Gerät zur Verwendung in einem Kommunikationsnetzwerk bereitgestellt. Das Gerät ist mit einem Prozessor ausgestattet, der zur Feststellung dient, dass eine Softwareanwendung in Form von mindestens einer elektronischen Datei verdächtig ist. Ein Sender wird bereitgestellt, der dazu dient, eine Anforderungsnachricht an einen Remote-Server zu senden, wobei die Anforderungsnachricht mindestens eine Identität der Quelle enthält, aus der die verdächtige Softwareanwendung bezogen wurde. Außerdem wird ein Empfänger bereitgestellt, der dazu dient, eine Antwort vom Server zu empfangen, wobei die Antwort einen Hinweis darauf enthält, ob die Softwareanwendung wahrscheinlich ein Trojanisches Pferd enthält, wobei die Wahrscheinlichkeit vom Server mindestens durch das Vergleichen der Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, und, falls sich die Quellen unterscheiden, durch die Feststellung, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält, bestimmt wurde.

[0021] Gemäß einem vierten Aspekt wird ein Server zur Verwendung in einem Kommunikationsnetzwerk bereitgestellt. Der Server ist mit einem Empfänger ausgestattet, der dazu dient, eine Anforderungsnachricht von einem Remote-Gerät zu empfangen, wobei die Anforderungsnachricht mindestens eine Identität einer Quelle enthält, aus der eine verdächtige Softwareanwendung in Form von mindestens einer elektronischen Datei bezogen wurde. Der Prozessor ist außerdem dafür ausgelegt, die Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, zu vergleichen. Falls sich die Quellen unterscheiden, stellt der Prozessor optional fest, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält. Ein Sender wird bereitgestellt, der dazu dient, eine Antwortnachricht zu senden, wobei die Antwortnachricht entweder das Ergebnis des Vergleiches oder einen Hinweis darauf enthält, dass die Softwareanwendung wahrscheinlich einen Trojaner enthält.

[0022] Optional umfasst der Server eine Datenbank zum Speichern der Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weiterer auf die saubere Version der Softwareanwendung bezogener Metadaten.

[0023] Optional umfassen die Quellen eine Identität eines Verkäufers.

[0024] Als eine weitere Option ist der Prozessor so konfiguriert, dass er die verdächtige Version der Softwareanwendung mit der sauberen Version der Softwareanwendung in Bezug auf die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte, vergleichen kann.

[0025] Gemäß einem fünften Aspekt wird ein Computerprogramm bereitgestellt, das maschinenlesbaren Code umfasst, der bei Ausführung auf einem Gerät bewirkt, dass das Gerät sich als ein Gerät verhält, wie es entweder im zweiten oder im dritten Aspekt beschrieben wurde.

[0026] Gemäß einem sechsten Aspekt wird ein Computerprogrammprodukt bereitgestellt, das ein maschinenlesbares Medium und ein Computerprogramm gemäß der Beschreibung im fünften Aspekt umfasst, wobei das Computerprogramm auf dem maschinenlesbaren Medium gespeichert ist.

[0027] Gemäß einem siebten Aspekt wird ein Computerprogramm bereitgestellt, das maschinenlesbaren Code umfasst, der bei Ausführung auf einem Server bewirkt, dass der Server sich als ein Server verhält, wie er im vierten Aspekt beschrieben wurde.

[0028] Gemäß einem achten Aspekt wird ein Computerprogrammprodukt bereitgestellt, das ein maschinenlesbares Medium und ein Computerprogramm gemäß der Beschreibung im siebten Aspekt umfasst, wobei das Computerprogramm auf dem maschinenlesbaren Medium gespeichert ist.

KURZBESCHREIBUNG DER ZEICHNUNGEN

[0029] Fig. 1 zeigt in einer Blockdarstellung schematisch eine Netzwerkarchitektur gemäß einer Ausgestaltung der vorliegenden Erfindung;

[0030] Fig. 2 zeigt in einem Ablaufdiagramm die Verfahrensschritte gemäß einer Ausgestaltung der vorliegenden Erfindung; und

[0031] Fig. 3 zeigt in einer Blockdarstellung schematisch ein Gerät gemäß einer weiteren Ausgestaltung der vorliegenden Erfindung.

AUSFÜHRLICHE BESCHREIBUNG

[0032] Um zu erkennen, ob eine verdächtige Softwareanwendung trojanisiert wurde, wird die verdächtige Softwareanwendung mit einer bekannten sauberen Version der Softwareanwendung verglichen. Wenn die verdächtige Softwareanwendung aus einer anderen Quelle als die bekannte saubere Version der Softwareanwendung bezogen wurde, erhöht dies die Wahrscheinlichkeit, dass die verdächtige Softwareanwendung trojanisiert wurde, und weitere Tests können durchgeführt werden.

[0033] Ein Anbieter von Sicherheitsdienstleistungen, beispielsweise von Antiviren-Software, beschafft sich bekannte saubere Versionen von Softwareanwendungen. Das kann durch das Annehmen von Einsendungen, beispielsweise aus einem vertrauenswürdigen Application Store, oder durch systematisches Durchsuchen des Internets erreicht werden. Aus den bekannten sauberen Versionen werden Metadaten extrahiert und in einer Datenbank gespeichert. Als Metadaten können beispielsweise gespeichert werden:

1. Quelle der Softwareanwendung; mit anderen Worten, der Application Store, aus dem sie bezogen wurde, der Verkäufer, der die Softwareanwendung geliefert hat, oder ein URI einer anderen Art von Website, von der sie bezogen wurde.
2. Der Name der Softwareanwendung und eine Identität eines Verkäufers, von dem die Anwendung bezogen wurde. Die Identität kann eine kryptographische Signatur oder auch einfach ein vom Verkäufer verwendeter Identitätswert sein.
3. Die Versionsnummer der Softwareanwendung.
4. Die Größe der Softwareanwendung.
5. Eine digitale Signatur, mit der die Softwareanwendung signiert wurde.
6. Eine Liste von Anwendungsklassen, importierten Programmierschnittstellen (APIs), aufgerufenen API-Funktionen und/oder Größen der Codeblöcke.
7. Eine Liste von Fähigkeiten und Zugriffskontrollen, die angibt, auf welche Funktionen die Anwendung zugreifen möchte.
8. Eine Liste von externen Eigenschaften der Anwendung, beispielsweise Bildabgleich eines Startbildes oder Bildabgleich des Icons der Benutzeroberfläche (UI), wenn es sich nicht um ein Standardicon des Betriebssystems handelt.
9. Cluster von ähnlichem Code, mit anderen Worten eine Anwendung, die beispielsweise 95% identischen Code mit anderen Anwendungen im selben Cluster aufweist (diese Analyse kann nur durch einen Backend-Server vorgenommen werden).
10. Installationsort beim Endbenutzer. Beispielsweise ist unter Microsoft Windows® der Pfad c:\Programme\Anwendungsname eindeutig und kann verwendet werden, um die Softwareanwendung einzuordnen. Unter Symbian® wird mit c:\private\APPUID (zum Beispiel c:\private\10002542b), was auch der einzige Ort ist, wo die Anwendung ihre Daten speichern kann, ein ähnliches Modell verwendet.

[0034] Die Erkennung einer trojanisierten Softwareanwendung kann im Backend durchgeführt werden, wo verdächtige Softwareanwendungen automatisch als Schadprogramme markiert werden, wenn sie als Trojanische Pferde identifiziert wurden. Alternativ kann eine Antiviren-Anwendung auf dem Gerät, das die verdächtige Anwendung bezogen hat, während der Untersuchung einer verdächtigen Anwendung die Metadaten sammeln und die Informationen mit einer lokalen Datenbank oder einer Datenbank in der Cloud abgleichen oder eine Anfrage an einen Backend-Server zum Abgleichen der Metadaten stellen.

[0035] Obwohl die Erkennung einer trojanisierten Softwareanwendung im Backend, auf einem Gerät oder auf einem Online-Antiviren-Server in Kommunikation mit einem Gerät durchgeführt werden kann, wird im folgenden Beispiel angenommen, dass die Erkennung mittels eines Endgeräts und eines Antiviren-Servers stattfindet.

[0036] In beigefügter **Fig.** 1 wird ein Gerät **1** dargestellt, bei dem es sich um einen Personalcomputer, ein Mobilgerät, ein Smartphone oder eine beliebige andere Art von Computergerät handeln kann. Das Gerät **1** kann mit einem Remote-Server **2** kommunizieren, der sich im Netzwerk **3** befindet.

[0037] Das Gerät **1** verfügt über ein maschinenlesbares Medium in Form eines Speichers **4**, in dem Dateien **5** gespeichert werden können. Ein Prozessor **6** identifiziert eine verdächtige Softwareanwendung. Dies geschieht beispielsweise bei der Untersuchung von Softwareanwendungen, die im Speicher **4** gespeichert sind, oder vor dem Speichern einer Softwareanwendung im Speicher, beispielsweise beim Herunterladen der Softwareanwendung aus einem Application Store. Der Prozessor **6** ermittelt die Quelle der Softwareanwendung und sammelt einige oder alle der anderen, oben beispielhaft beschriebenen Metadaten, die sich auf die Softwareanwendung beziehen. Ein Sender **7** sendet eine Nachricht an den Server **2**, die die Metadaten enthält.

[0038] Der Server **2** verfügt über einen Empfänger **8**, der die Nachricht vom Gerät **1** empfängt. Ein Prozessor **9** steht zur Verfügung, der die in der Nachricht enthaltenen Metadaten analysiert und die Metadaten mit Metadaten einer bekannten sauberen Version der Softwareanwendung vergleicht, die in einer Datenbank **10** gespeichert sind, wobei die Datenbank auf einem maschinenlesbaren Medium in Form eines Speichers **11** gespeichert ist. Zu beachten ist, dass sich die Datenbank **10** in der Darstellung auf dem Server **2** befindet, es aber ebenso möglich ist, dass sich die Datenbank **10** auf einem anderen Knoten befindet, den der Server **2** abfragen kann.

[0039] Der Prozessor vergleicht zunächst die Quelle der verdächtigen Softwareanwendung (beispielsweise die Identität eines Application Stores, aus dem sie bezogen wurde, oder des Verkäufers, von dem sie erworben wurde) mit der in der Datenbank **10** gespeicherten Quelle der entsprechenden sauberen Softwareanwendung. Wenn die Quelle dieselbe ist, ist die Wahrscheinlichkeit gering, dass die verdächtige Softwareanwendung trojanisiert wurde, wobei allerdings andere Indikatoren, die unten beschrieben werden, darauf hindeuten können, dass die Softwareanwendung trojanisiert wurde. Es ist möglich, wenn auch unwahrscheinlich, dass eine saubere Originalversion der Softwareanwendung in einem Application Store verfügbar ist und eine trojanisierte Version der Softwareanwendung im selben Application Store verfügbar ist.

[0040] Wenn die verdächtige Softwareanwendung aus einer anderen Quelle als die saubere Version der Softwareanwendung bezogen wurde, erhöht dies die Wahrscheinlichkeit, dass die verdächtige Softwareanwendung trojanisiert wurde. Eine andere Quelle könnte bedeuten, dass verschiedene Versionen der Softwareanwendung in verschiedenen Application Stores verfügbar sind, kann aber auch bedeuten, dass verschiedene Versionen der Softwareanwendung bei verschiedenen Verkäufern im selben oder in verschiedenen Application Stores verfügbar sind. Weitere Untersuchungen können durchgeführt werden. Beispielsweise kann die verdächtige Softwareanwendung zusätzliche Funktionen enthalten, die in der sauberen Softwareanwendung nicht enthalten sind. Dazu zählen beispielsweise die Fähigkeiten, Telefonnummern anzurufen, auf bestimmte Websites zuzugreifen oder Kurznachrichten (SMS) zu verschicken. Wenn die verdächtige Version der Softwareanwendung diese Funktionalität enthält, die saubere Version aber nicht, besteht eine sehr hohe Wahrscheinlichkeit, dass die verdächtige Version trojanisiert wurde.

[0041] Die verdächtige Version der Softwareanwendung kann Anfragen nach mehr Fähigkeiten stellen als die saubere Version der Softwareanwendung, beispielsweise Zugriffsrechte auf bestimmte Dateitypen und ähnliches. Auch dies ist ein starkes Indiz dafür, dass die verdächtige Version der Softwareanwendung trojanisiert wurde.

[0042] Die digitalen Signaturen der verdächtigen Version der Softwareanwendung und der sauberen Version der Softwareanwendung können miteinander verglichen werden. Auch wenn die digitale Signatur der verdächtigen Version der Softwareanwendung gültig ist, ist eine Abweichung von der digitalen Signatur der sauberen Version der Softwareanwendung ein Indiz dafür, dass die verdächtige Version der Softwareanwendung in irgendeiner Form modifiziert wurde. Das erhöht die Wahrscheinlichkeit, dass die verdächtige Version der Softwareanwendung trojanisiert wurde.

[0043] Es kann auch ein Vergleich zwischen den Funktionen der sauberen Version der Softwareanwendung und der verdächtigen Version der Softwareanwendung durchgeführt werden. Wenn beispielsweise die verdächtige Version der Softwareanwendung andere Funktionen aufweist als die saubere Version der Software-

anwendung oder andere Ressourcen erfordert, dann wurde die verdächtige Version der Softwareanwendung in irgendeiner Form modifiziert und die Wahrscheinlichkeit ist höher, dass sie trojanisiert wurde.

[0044] Ein weiterer Test könnte sein, die verdächtige Version der Softwareanwendung auf zu Verschleierungszwecken eingesetzte Verschlüsselungsroutinen zu überprüfen, insbesondere wenn diese in der sauberen Version der Softwareanwendung nicht vorhanden sind. Dies liefert ein weiteres Indiz dafür, dass die verdächtige Version der Softwareanwendung trojanisiert wurde, da jeder in die trojanisierte verdächtige Version der Softwareanwendung eingebrachte Schadcode wahrscheinlich Verschleierungstechniken verwenden würde.

[0045] Andere Metadaten, beispielsweise die Versionsnummer, Versionsgeschichte und ähnliches, können ebenfalls verglichen werden. Jeder Unterschied zwischen den Metadaten der sauberen Version der Softwareanwendung und denen der verdächtigen Version der Softwareanwendung liefert Indizien dafür, dass die verdächtige Version der Softwareanwendung in irgendeiner Form modifiziert wurde, und erhöht damit die Wahrscheinlichkeit, dass die verdächtige Version der Softwareanwendung trojanisiert wurde.

[0046] Die Datenbank **10** kann die Anwendungen auf der Grundlage ihres Namens, ihrer Identität und ihrer Versionsinformationen gruppieren. Anwendungen können auch anhand von externen Eigenschaften gruppiert werden, beispielsweise anhand von Startbild, Bildabgleich des Icons der Benutzeroberfläche (UI) oder, wie oben beschrieben, Clustern von ähnlichem Code. Anwendungen in derselben Gruppe können überprüft werden, um sicherzustellen, dass die gleichen Versionen der sauberen Softwareanwendung und der verdächtigen Softwareanwendung miteinander verglichen werden.

[0047] Während eine einzige Änderung der oben beschriebenen Art noch nicht verdächtig genug sein mag, ist die Erkennung mehrerer Änderungen sehr verdächtig, wenn die verdächtige Version vorgibt, dieselbe Version oder eine aktualisierte Version einer existierenden Softwareanwendung zu sein. Ferner kann die Versionsgeschichte genutzt werden, um die Erkennung einer trojanisierten Softwareanwendung zu unterstützen. Wenn eine Funktionalität in einer bestimmten Version vorhanden ist, nicht aber in anderen (entweder älteren oder neueren) Versionen derselben Softwareanwendung, dann ist dies ein Indiz dafür, dass ein bösartiger Dritter die Softwareanwendung trojanisiert hat.

[0048] Der Speicher **4** auf dem Gerät **1** kann auch genutzt werden, um ein Computerprogramm **14** zu speichern, dass bei Ausführung durch den Prozessor **6** das Gerät **1** dazu bringt, sich wie oben beschrieben zu verhalten. Auf ähnliche Weise kann der Speicher **11** auf dem Server **2** auch genutzt werden, um ein Computerprogramm **15** zu speichern, dass bei Ausführung durch den Prozessor **9** den Server **2** dazu bringt, sich wie oben beschrieben zu verhalten.

[0049] Um die Datenbank **10** zu befüllen, kann der Server **2** systematisch das Internet durchsuchen, um saubere Versionen (oder auf saubere Versionen bezogene Metadaten) von Softwareanwendungen aus vertrauenswürdigen Application Stores zu beziehen. Alternativ können Softwareentwickler dem Server saubere Versionen von Softwareanwendungen (oder aus einer sauberen Kopie der Softwareanwendung gewonnene Metadaten) zur Verfügung stellen.

[0050] Um die Erfindung besser beschreiben zu können, sind in **Fig.** 2 Verfahrensschritte gemäß einer Ausgestaltung der vorliegenden Erfindung dargestellt. Die folgenden Nummern entsprechen den in **Fig.** 2 verwendeten Nummern:

S1. Ein Gerät mit einer verdächtigen Softwareanwendung ermittelt die Quelle der Softwareanwendung. Typischerweise ist diese der Application Store, aus dem die verdächtige Software bezogen wurde.

S2. Zwischen der Quelle der verdächtigen Softwareanwendung und der Quelle der sauberen Originalversion derselben Softwareanwendung wird ein Vergleich durchgeführt.

S3. Wenn die Quellen übereinstimmen, wird der Vorgang bei Schritt S6 fortgesetzt, andernfalls wird der Vorgang bei Schritt S4 fortgesetzt.

S4. Wenn die Quellen nicht übereinstimmen, werden weitere Überprüfungen durchgeführt, die beliebige der oben beschriebenen Überprüfungen sein können.

S5. Die Ergebnisse der weiteren Vergleiche werden gemeldet, um zu bestimmen, ob die verdächtige Software wahrscheinlich trojanisiert wurde oder nicht.

S6. Wenn die Quellen übereinstimmen, ist die Wahrscheinlichkeit gering, dass die verdächtige Software trojanisiert wurde.

[0051] Das oben angeführte Beispiel beschreibt eine Ausgestaltung, bei der ein Gerät **1** Informationen an einen Server **2** sendet, der dann den Vergleich zwischen den Quellen der Anwendungen durchführt. Es ist ebenso möglich, andere Architekturen zu verwenden, wofür in **Fig.** 3 ein Beispiel dargestellt ist.

[0052] In diesem Beispiel umfasst ein Gerät **16** einen Prozessor **17** und ein maschinenlesbares Medium in Form eines Speichers **18**. Der Speicher **18** wird verwendet, um Dateien **19** und außerdem eine Datenbank **20** bekannter sauberer Versionen von Softwareanwendungen zu speichern. Ein Empfänger **21** wird bereitgestellt, der dazu dient, eine Softwareanwendung aus einem Application Store herunterzuladen. Der Prozessor **17** ist so konfiguriert, dass er einen Vergleich zwischen der Quelle der heruntergeladenen Softwareanwendung und einer in der Datenbank **20** gespeicherten sauberen Version der Softwareanwendung durchführt. In diesem Beispiel ist es für das Gerät nicht erforderlich, einen Remote-Server zu kontaktieren, um zu bestimmen, ob die verdächtige Softwareanwendung wahrscheinlich trojanisiert wurde oder nicht.

[0053] Im Speicher **18** kann außerdem ein Computerprogramm **22** gespeichert sein. Das Programm **22** ist so konfiguriert, dass es bei Ausführung durch den Prozessor **17** das Gerät **16** dazu bringt, sich wie oben beschrieben zu verhalten.

[0054] Während das Gerät **16** hier eine lokal auf dem Gerät gespeicherte Datenbank **20** verwendet, kann es in einer alternativen Ausgestaltung möglich sein, einen Backend-Server oder eine Datenbank in einem verteilten Netzwerk abzufragen, um die für den Vergleich zwischen der Quelle der verdächtigen Softwareanwendung und der Quelle der sauberen Originalversion der Anwendung und für jegliche anderen Metadatenvergleiche benötigten Informationen zu erhalten.

[0055] Um die Funktionsweise der Erfindung zu illustrieren, wird das Beispiel einer Softwareanwendung namens MonkeyJump2 verwendet, die Ende 2010 unter Verwendung von Geinimi trojanisiert und über einen chinesischen Application Store verteilt wurde. Die originale Softwareanwendung MonkeyJump2 wurde über einen vertrauenswürdigen Application Store mit Sitz in den USA vertrieben.

[0056] Tabelle 1 zeigt einen Datenvergleich zwischen der sauberen Version von MonkeyJump2 und der trojanisierten Version:

Tabelle 1. Vergleich zwischen sauberer und trojanisierter Version von MonkeyJump2

	Original	Trojaner
Quelle	Vertrauenswürdiger, stark frequentierter Application Store in den USA	Weniger bekannter Application Store in China
Unterzeichner	Unternehmen A	Unternehmen B
Fähigkeiten	Nur wenige und übliche	Äußerst lange Liste von die Privatsphäre verletzenden Fähigkeiten
Anzahl der Klassen	N Klassen	N + 1 Klassen
Funktionsaufrufe	Für Spielprogramme übliche	Kryptographische Aufrufe (DES-Funktionen)

[0057] Die zum Stand der Technik gehörenden Verfahren zur Erkennung der Trojanisierung der verdächtigen Version von MonkeyJump2 führen eine Analyse des Codes durch oder emulieren das Verhalten der verdächtigen Anwendung. Beide Vorgehensweisen sind ressourcenintensiv und zeitaufwendig und können unzuverlässig sein. Mit der vorliegenden Erfindung zeigt hingegen ein schneller Vergleich zwischen der Quelle der verdächtigen Softwareanwendung und der Originalversion der Softwareanwendung, dass die Quellen sich unterscheiden, wodurch sich die Wahrscheinlichkeit erhöht, dass die verdächtige Softwareanwendung trojanisiert wurde. Daraufhin werden weitere Analysen durchgeführt, um die Wahrscheinlichkeit zu bestimmen, dass die verdächtige Softwareanwendung trojanisiert wurde.

[0058] Dabei ist erkennbar, dass die beiden Versionen von unterschiedlichen Unternehmen signiert wurden und dass die trojanisierte Version der Softwareanwendung Fähigkeiten aufwies, die in der Originalversion nicht enthalten waren. Diese zusätzlichen Fähigkeiten waren außerdem geeignet, die Privatsphäre zu verletzen, was auf eine Trojanisierung der verdächtigen Version der Softwareanwendung hindeutet. Zusätzlich verwendete die

verdächtige Version der Softwareanwendung eine Klasse mehr als die Originalversion und enthielt außerdem Funktionsaufrufe, die im Original nicht enthalten waren. Einer dieser zusätzlichen Funktionsaufrufe verwendete kryptographische Funktionen, womit ein weiteres Indiz für die Feststellung gegeben ist, dass die verdächtige Version der Softwareanwendung trojanisiert wurde.

[0059] Die vorliegende Erfindung ermöglicht eine stark vereinfachte und beschleunigte Feststellung, dass eine Softwareanwendung, insbesondere eine aus einem Application Store bezogene Softwareanwendung, trojanisiert wurde, indem sie die Quelle und andere Metadaten einer verdächtigen Softwareanwendung mit denen einer sauberen Originalversion derselben Softwareanwendung vergleicht.

[0060] Für den Fachmann ist erkennbar, dass die oben beschriebene Ausgestaltung auf unterschiedliche Weise modifiziert werden kann, ohne den Anwendungsbereich der vorliegenden Erfindung zu verlassen. Insbesondere kann die Systemarchitektur verändert werden, ein Gerät kann Informationen von einem lokalen Server oder aus einer Datenbank in der Cloud beziehen, und der Vergleich der auf die verdächtige Softwareanwendung bezogenen Metadaten kann von einem Backend-Server durchgeführt werden.

Patentansprüche

1. Verfahren zur Erkennung eines Trojanischen Pferdes in einer verdächtigen Softwareanwendung in Form von mindestens einer elektronischen Datei, umfassend:
auf einem Computergerät, das Ermitteln der Quelle, aus der eine verdächtige Softwareanwendung bezogen wurde;
das Vergleichen der Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde; und,
falls sich die Quellen unterscheiden, das Feststellen, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält.

2. Verfahren nach Anspruch 1, wobei die Quellen eine Identität eines Verkäufers umfassen.

3. Verfahren nach einem der vorhergehenden Ansprüche, ferner umfassend das Vergleichen der verdächtigen Version der Softwareanwendung mit der sauberen Version der Softwareanwendung in Bezug auf die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte.

4. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weitere auf die saubere Version der Softwareanwendung bezogene Metadaten in einer Datenbank gespeichert werden.

5. Verfahren nach einem der vorhergehenden Ansprüche, ferner umfassend das Senden einer Nachricht von einem Computergerät an einen Remote-Server, wobei die Nachricht die Bestandteile der verdächtigen Anwendung und die auf die verdächtige Anwendung bezogenen Metadaten enthält, die der Server benötigt, um den Vergleich durchzuführen.

6. Verfahren nach Anspruch 5, ferner umfassend das Empfangen einer Antwortnachricht vom Server auf dem Gerät, wobei die Antwortnachricht einen Hinweis enthält, dass die verdächtige Softwareanwendung wahrscheinlich ein Trojanisches Pferd enthält.

7. Verfahren nach Anspruch 1, ferner umfassend das Senden einer Nachricht von dem Gerät an eine Remote-Datenbank, wobei die Nachricht mindestens eine Identität der verdächtigen Softwareanwendung enthält, und das Empfangen einer Antwort von der Remote-Datenbank, wobei die Antwort die Quelle enthält, aus der die saubere Originalversion der Softwareanwendung bezogen wurde.

8. Gerät zur Verwendung in einem Kommunikationsnetzwerk, umfassend:
einen Prozessor, der zur Feststellung dient, dass eine Softwareanwendung in Form von mindestens einer elektronischen Datei verdächtig ist;
wobei der Prozessor ferner so konfiguriert ist, dass er die Quelle der Softwareanwendung feststellen kann;
wobei der Prozessor ferner so konfiguriert ist, dass er die Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wur-

de, vergleichen und, falls sich die Quellen unterscheiden, feststellen kann, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält.

9. Gerät nach Anspruch 8, ferner umfassend:
einen Sender, der dazu dient, entweder an einen Server oder an eine Datenbank eine Anforderungsnachricht zu senden, wobei die Anforderungsnachricht eine Anforderung nach der Quelle enthält, aus der die saubere Originalversion der Softwareanwendung bezogen wurde; und
einen Empfänger, der dazu dient, eine Antwort zu empfangen, wobei die Antwort die Quelle enthält, aus der die saubere Originalversion der Softwareanwendung bezogen wurde.

10. Gerät nach Anspruch 8 oder 9, wobei der Prozessor ferner so konfiguriert ist, dass er die verdächtige Version der Softwareanwendung mit der sauberen Version der Softwareanwendung in Bezug auf die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte, vergleichen kann.

11. Gerät nach Anspruch 8, ferner umfassend eine Datenbank zum Speichern von Daten bezüglich der Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weiterer auf die saubere Version der Softwareanwendung bezogener Metadaten.

12. Gerät zur Verwendung in einem Kommunikationsnetzwerk, umfassend:
einen Prozessor, der zur Feststellung dient, dass eine Softwareanwendung in Form von mindestens einer elektronischen Datei verdächtig ist;
einen Sender, der dazu dient, eine Anforderungsnachricht an einen Remote-Server zu senden, wobei die Anforderungsnachricht mindestens eine Identität der Quelle enthält, aus der die verdächtige Softwareanwendung bezogen wurde;
einen Empfänger, der dazu dient, eine Antwort vom Server zu empfangen, wobei die Antwort einen Hinweis darauf enthält, ob die Softwareanwendung wahrscheinlich ein Trojanisches Pferd enthält, wobei die Wahrscheinlichkeit vom Server mindestens durch das Vergleichen der Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, und, falls sich die Quellen unterscheiden, durch die Feststellung, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält, bestimmt wurde.

13. Server zur Verwendung in einem Kommunikationsnetzwerk, umfassend:
einen Empfänger, der dazu dient, eine Anforderungsnachricht von einem Remote-Gerät zu empfangen, wobei die Anforderungsnachricht mindestens eine Identität einer Quelle enthält, aus der eine verdächtige Softwareanwendung in Form von mindestens einer elektronischen Datei bezogen wurde;
einen Prozessor, der dazu dient, die Quelle, aus der die verdächtige Softwareanwendung bezogen wurde, mit einer Quelle, aus der eine saubere Originalversion der Softwareanwendung bezogen wurde, zu vergleichen und, falls sich die Quellen unterscheiden, festzustellen, dass die verdächtige Anwendung wahrscheinlich ein Trojanisches Pferd enthält;
einen Sender, der dazu dient, eine Antwortnachricht zu senden, wobei die Antwortnachricht entweder das Ergebnis des Vergleiches oder einen Hinweis darauf enthält, dass die Softwareanwendung wahrscheinlich einen Trojaner enthält.

14. Server nach Anspruch 13, ferner umfassend eine Datenbank zum Speichern der Quelle, aus der die saubere Originalversion der Softwareanwendung bezogen wurde, und weiterer auf die saubere Version der Softwareanwendung bezogener Metadaten.

15. Server nach Anspruch 13 oder 14, wobei die Quellen eine Identität eines Verkäufers umfassen.

16. Server nach einem der Ansprüche 13 bis 15, wobei der Prozessor ferner so konfiguriert ist, dass er die verdächtige Version der Softwareanwendung mit der sauberen Version der Softwareanwendung in Bezug auf die Versionsnummern, Versionsgeschichten, Anwendungsklassen, Größe der Codeblöcke, importierte Programmierschnittstellen, aufgerufene API-Funktionen, Dateigröße der Komponenten der Softwareanwendung und/oder Fähigkeiten und Zugriffskontrollen, die angeben, auf welche Funktionen die Anwendung zugreifen möchte, vergleichen kann.

17. Computerprogramm, umfassend maschinenlesbaren Code, der bei Ausführung auf einem Gerät bewirkt, dass das Gerät sich als ein Gerät nach einem der Ansprüche 8 bis 12 verhält.

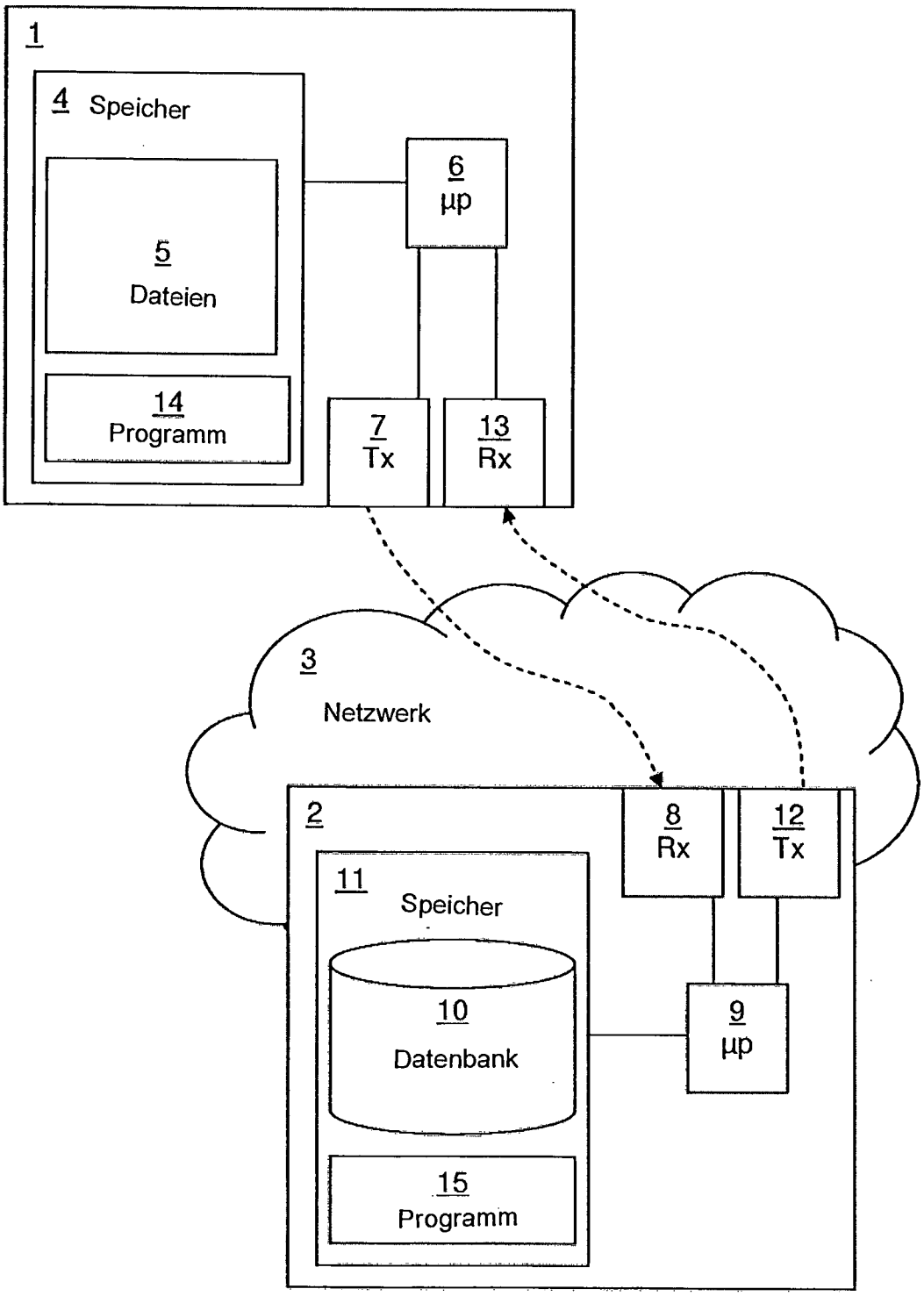
18. Computerprogrammprodukt, umfassend ein maschinenlesbares Medium und ein Computerprogramm nach Anspruch 17, wobei das Computerprogramm auf dem maschinenlesbaren Medium gespeichert ist.

19. Computerprogramm, umfassend maschinenlesbaren Code, der bei Ausführung auf einem Server bewirkt, dass der Server sich als ein Server nach Anspruch 13 verhält.

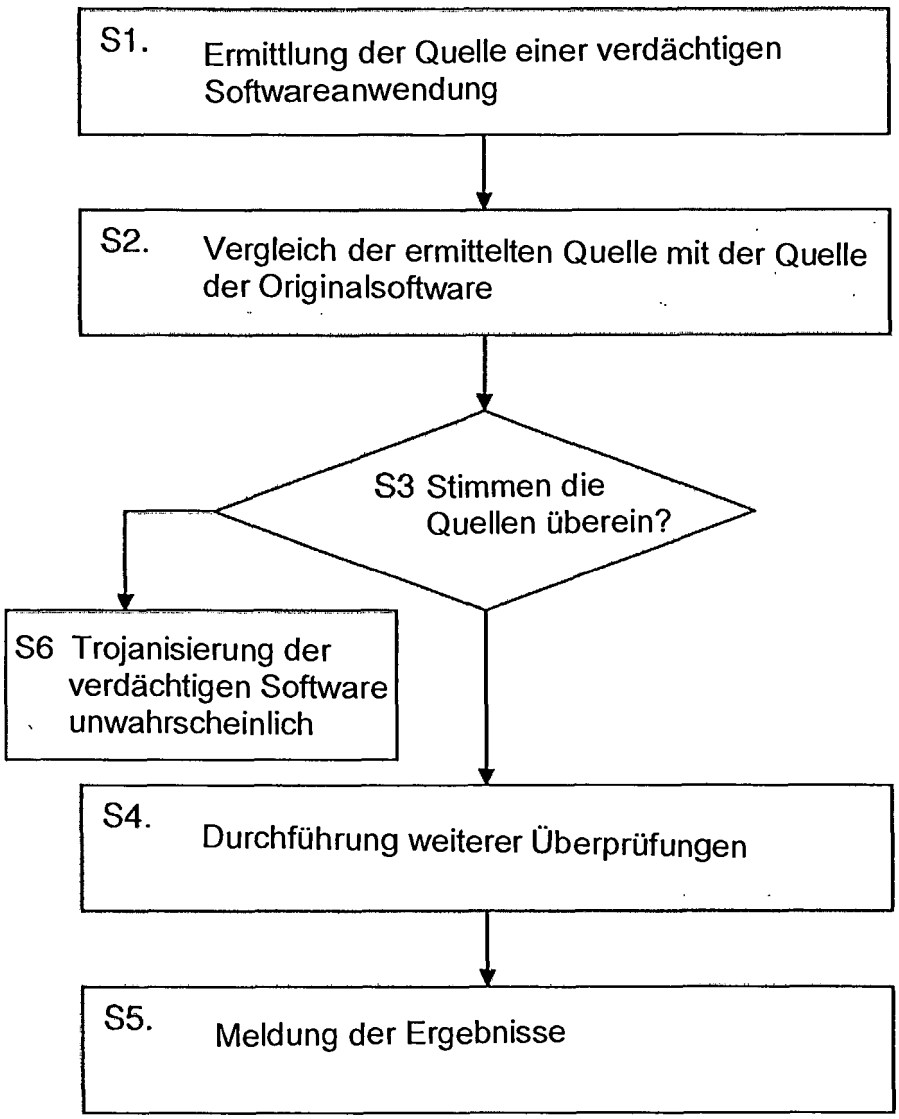
20. Computerprogrammprodukt, umfassend ein maschinenlesbares Medium und ein Computerprogramm nach Anspruch 19, wobei das Computerprogramm auf dem maschinenlesbaren Medium gespeichert ist.

Es folgen 3 Seiten Zeichnungen

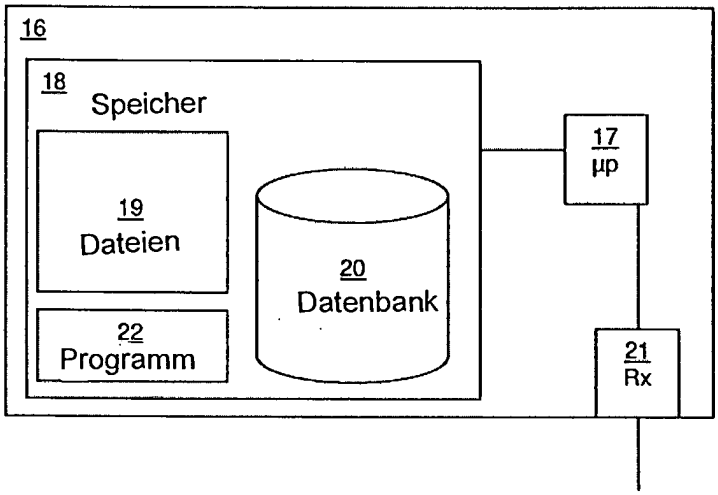
Anhängende Zeichnungen



Figur 1


S1. Ermittlung der Quelle einer verdächtigen Softwareanwendung
S2. Vergleich der ermittelten Quelle mit der Quelle der Originalsoftware
S3 Stimmen die Quellen überein?
S6 Trojanisierung der verdächtigen Software unwahrscheinlich
S4. Durchführung weiterer Überprüfungen
S5. Meldung der Ergebnisse

Figur 2


16
18
Speicher
19
Dateien
22
Programm
20
Datenbank
17
µp
21
Rx

Figur 3