(19) Deutsches Patent- und Markenamt

(10) **DE 10 2013 210 837 A1** 2014.01.02

(12) # Offenlegungsschrift

(21) Aktenzeichen: **10 2013 210 837.0**
(22) Anmeldetag: **11.06.2013**
(43) Offenlegungstag: **02.01.2014**

(51) Int Cl.: ***G06F 21/60*** (2013.01)

(30) Unionspriorität:
13/539,325 **30.06.2012** **US**
13/869,179 **24.04.2013** **US**

(71) Anmelder:
International Business Machines Corp., Armonk, N.Y., US

(74) Vertreter:
RICHARDT PATENTANWÄLTE GbR, 65185, Wiesbaden, DE

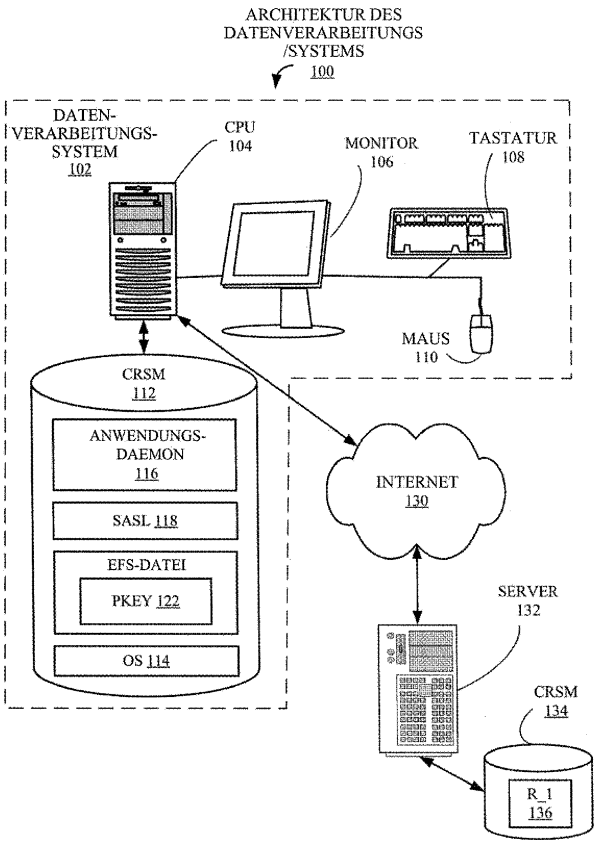
(72) Erfinder:
Mullen, Shawn P., Austin, Tex., US; Murillo, Jessica C., Austin, Tex., US; Shieh, Johnny M., Austin, Tex., US

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Startanwendung kryptographischer Schlüsselspeicher**

(57) Zusammenfassung: Es werden Techniken bereitgestellt zum Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel; zu einem Installationszeitpunkt Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört; Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen; zu einem Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon; Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des abgeleiteten ersten Schlüssels. Zu Beispielen für geschützte Ressourcen gehören Datenbanken, Datenübertragungseinheiten und ein Lightweight Directory Access Protocol Server, ohne jedoch darauf beschränkt zu sein.



Beschreibung

QUERVERWEISE AUF VERWANDTE ANMELDUNGEN

[0001] Die vorliegende Anmeldung ist eine Fortsetzung und beansprucht den Vorteil des Einreichungsdatums einer eigenen Anmeldung mit dem Titel „Start Method for Application Cryptographic Keystores", Seriennummer 13/539,325, die am 30. Juni 2012 eingereicht wurde und durch Verweis darauf als hier mit aufgenommen gilt.

GEBIET DER OFFENBARUNG

[0002] Der beanspruchte Gegenstand bezieht sich allgemein auf die Computersicherheit und insbesondere auf Techniken zum Sicherstellen, dass auf eine verschlüsselte Einheit, einen verschlüsselten Dienst oder verschlüsselte Daten lediglich durch eine ausführbare Datei eines berechtigten Computers zugegriffen werden kann.

KURZDARSTELLUNG

[0003] Es werden Techniken bereitgestellt, die sicherstellen, dass auf eine verschlüsselte Einheit, einen verschlüsselten Dienst oder verschlüsselte Daten lediglich durch eine ausführbare Datei eines berechtigten Computers zugegriffen werden kann. Eine übliche Forderung bei Systemen der Informationstechnologie (IT-Systemen) besteht darin, dass bei Systemen und Anwendungen häufig die Fähigkeit zum Booten oder Neubooten ohne Bedienereingriff gefordert wird. Ein Datenbankserver, der auf verschlüsselte Daten zugreift, kann z. B. mitten in der Nacht eine Netzstörung erleiden. Üblicherweise muss ein Systemadministrator zur Verfügung stehen, um ein Passwort oder einen Schlüssel einzugeben, damit der Server auf die Daten zugreifen kann. Diese Forderung ist bei geschäftskritischen Computersystemen und Anwendungen üblich. Bei einem weiteren Beispiel verwendet ein AIX-Lightweight-Directory-Access-Protocol (LDAP) ein Passwort oder einen Schüssel zum Berechtigen bei einem sicheren LDAP-Server, und AIX stellt eine Datei bereit, in der der unverschlüsselte Schüssel gespeichert ist, bis er beim Booten oder Neustart des LDAP-Client-Daemon erforderlich ist. Um ein sicheres System bereitzustellen, sollte jedoch die Datei mit dem unverschlüsselten Schlüssel durch eine starke Verschlüsselung gesichert sein. Dabei handelt es sich um ein Henne-oder-Ei-Problem.

[0004] Es werden Techniken bereitgestellt zum Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel; zu einem Installationszeitpunkt der Anwendung Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört; Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen; zu einem Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon; Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des ersten abgeleiteten Schlüssels.

[0005] Diese Zusammenfassung ist nicht als umfassende Beschreibung des beanspruchten Gegenstands gedacht, sondern soll einen kurzen Überblick über einige der damit verbundenen Funktionen bereitstellen. Weitere Systeme, Verfahren, Funktionen, Merkmale und Vorteile des beanspruchten Gegenstands werden für einen Fachmann bei einer Prüfung der folgenden Figuren und der genauen Beschreibung erkennbar.

KURZBESCHREIBUNG DER ZEICHNUNGEN

[0006] Ein besseres Verständnis des beanspruchten Gegenstands kann erreicht werden, wenn die folgende genaue Beschreibung der offenbarten Ausführungsformen in Verbindung mit den nachfolgenden Figuren betrachtet wird.

[0007] Fig. 1 ist ein Beispiel für eine Architektur eines Datenverarbeitungssystems, bei der der beanspruchte Gegenstand umgesetzt sein kann.

[0008] Fig. 2 ist ein Blockschaubild eines sicheren Anwendungsstartladeprogramms (SASL – Secure Application Startup Loader), das zuerst in **Fig.** 1 eingeführt wurde.

[0009] Fig. 3 ist ein Ablaufplan eines Prozesses „installiere Anwendung", der Aspekte des beanspruchten Gegenstands umsetzen kann.

[0010] Fig. 4 ist ein Ablaufplan eines Prozesses „Starte Anwendung", der Aspekte des beanspruchten Gegenstands umsetzen kann.

AUSFÜHRLICHE BESCHREIBUNG

[0011] Wie dem Fachmann klar sein wird, können Aspekte der vorliegenden Erfindung als System, Verfahren oder Computerprogrammprodukt ausgeführt werden. Dementsprechend können Aspekte der vorliegenden Erfindung die Form einer reinen Hardware Ausführungsform, einer reinen Software-Ausführungsform (darunter Firmware, residente Software, Mikrocode usw.) oder einer Ausführungsform, die Software- und Hardware-Aspekte kombiniert, annehmen, die hier alle allgemein als "Schaltung", "Modul" oder "System" bezeichnet werden können. Des

Weiteren können Aspekte der vorliegenden Erfindung die Form eines Computerprogrammprodukts annehmen, das in einem oder mehreren computerlesbaren Medien verkörpert ist, auf denen sich computerlesbarer Programmcode befindet.

[0012] Jede Kombination aus einem oder mehreren computerlesbaren Medien kann verwendet werden. Das computerlesbare Medium kann ein computerlesbares Signalmedium oder ein computerlesbares Speichermedium sein. Ein computerlesbares Speichermedium kann z. B. ein elektronisches, magnetisches, optisches, elektromagnetisches, infrarot- oder Halbleitersystem, eine solche Vorrichtung oder Einheit oder jede geeignete Kombination daraus sein, ist jedoch nicht darauf beschränkt. Zu spezifischeren Beispielen (einer unvollständigen Liste) für das computerlesbare Speichermedium würde Folgendes gehören: eine elektrische Verbindung mit einer oder mehreren Leitungen, eine tragbare Computerdiskette, eine Festplatte, ein Direktzugriffsspeicher (RAM), ein Festwertspeicher (ROM), ein löschbarer programmierbarer Festwertspeicher (EPROM oder Flash-Speicher), ein Lichtwellenleiter, ein tragbarer Compactdisk-Festwertspeicher (CD-ROM), eine optische Speichereinheit, eine magnetische Speichereinheit oder jede geeignete Kombination davon. Im Kontext dieses Dokuments kann ein computerlesbares Speichermedium jedes materielle Medium sein, das ein Programm zur Verwendung durch oder in Verbindung mit einem System, einer Vorrichtung oder einer Einheit zur Befehlsausführung enthalten oder speichern kann.

[0013] Ein computerlesbares Signalmedium kann ein verbreitetes Datensignal mit computerlesbarem Programmcode, der darin z. B. im Basisband oder als Teil einer Trägerwelle verkörpert ist, enthalten. Ein derartiges verbreitetes Signal kann jede von einer Vielfalt von Formen annehmen, zu denen elektromagnetische, optische Formen oder jede geeignete Kombination hiervon gehören, ohne jedoch auf diese beschränkt zu sein. Ein computerlesbares Signalmedium kann jedes computerlesbare Medium sein, das kein computerlesbares Speichermedium ist und ein Programm zur Verwendung durch oder in Verbindung mit einem System, einer Vorrichtung oder einer Einheit zur Befehlsausführung übertragen, verbreiten oder transportieren kann.

[0014] Programmcode, der auf einem computerlesbaren Medium verkörpert ist, kann unter Verwendung jedes geeigneten Mediums übertragen werden, wozu drahtlose, leitungsgestützte, Lichtwellenleiterkabel-, HF-Medien usw. oder jede geeignete Kombination davon gehören, ohne jedoch darauf beschränkt zu sein.

[0015] Computerprogrammcode zum Ausführen von Operationen für Aspekte der vorliegenden Erfindung kann in jeder Kombination aus einer oder mehreren Programmiersprachen geschrieben sein, darunter eine objektorientierte Programmiersprache wie Java, Smalltalk, C++ oder dergleichen und herkömmliche prozedurale Programmiersprachen wie etwa die Programmiersprache ”C” oder ähnliche Programmiersprachen. Der Programmcode kann nur auf dem Computer des Benutzers, teilweise auf dem Computer des Benutzers, als eigenständiges Software-Paket, teilweise auf dem Computer des Benutzers und teilweise auf einem fernen Computer oder nur auf dem fernen Computer oder Server ausgeführt werden. Bei dem zuletzt genannten Szenario kann der ferne Computer mit dem Computer des Benutzers über einen beliebigen Netzwerktyp verbunden sein, einschließlich eines Lokalbereichsnetzwerks (LAN) oder eines Weitbereichsnetzwerks (WAN), oder die Verbindung kann zu einem externen Computer (z. B. über das Internet unter Verwendung eines Internet-Dienstanbieters) hergestellt werden.

[0016] Aspekte der vorliegenden Erfindung sind hier unter Bezugnahme auf Ablaufplandarstellungen und/oder Blockschaubilder von Verfahren, Vorrichtungen (Systemen) und Computerprogrammprodukten gemäß Ausführungsformen der Erfindung beschrieben. Es ist klar, dass jeder Block der Ablaufplandarstellungen und/oder Blockschaubilder und Kombinationen von Blöcken in den Ablaufplandarstellungen und/oder Blockschaubildern durch Computerprogrammbefehle umgesetzt werden können. Diese Computerprogrammbefehle können für einen Prozessor eines Universalcomputers, eines Spezialcomputers oder einer anderen programmierbaren Datenverarbeitungsvorrichtung bereitgestellt werden, um eine solche Maschine zu bilden, dass Befehle, die über den Prozessor des Computers oder der anderen programmierbaren Datenverarbeitungsvorrichtung ausgeführt werden, Mittel zum Umsetzen der Funktionen/Handlungen, die in dem Block oder den Blöcken des Ablaufplans und/oder Blockschaubilds spezifiziert sind, erzeugen.

[0017] Diese Computerprogrammbefehle können außerdem in einem computerlesbaren Medium gespeichert sein, das einen Computer, eine andere programmierbare Datenverarbeitungsvorrichtung oder andere Einheiten anleiten kann, in einer bestimmten Weise zu funktionieren, so dass die in dem computerlesbaren Medium gespeicherten Befehle einen Herstellungsgegenstand produzieren, wozu Befehle gehören, die die Funktion/Handlung umsetzen, die in dem Block oder den Blöcken des Ablaufplans und/oder Blockschaubilds spezifiziert sind.

[0018] Die Computerprogrammbefehle können außerdem in einen Computer, eine andere programmierbare Datenverarbeitungsvorrichtung oder andere Einheiten geladen werden, um zu bewirken, dass eine Reihe von Operationsschritten auf dem Compu-

ter, der anderen programmierbaren Datenverarbeitungsvorrichtung oder anderen Einheiten ausgeführt werden und ein mittels Computer umgesetzter Prozess erzeugt wird, so dass die Befehle, die auf dem Computer oder der anderen programmierbaren Vorrichtung ausgeführt werden, Prozesse zum Umsetzen der Funktionen/Handlungen, die in dem Block oder den Blöcken des Ablaufplans und/oder Blockschaubilds spezifiziert sind, bereitstellen.

[0019] Obwohl die offenbarte Technologie auf viele Typen von Anwendungen und Ressourcen angewendet werden kann, verwendet die folgende Beschreibung für Erläuterungszwecke eine Datenbankanwendung und geschützte Daten, auf die die Anwendung zugreift. Bei dem anderen Beispiel, das oben verwendet wurde, handelt es sich um LDAP-Clients und -Server. Für einen Fachmann sollten die vielen Szenarien klar sein, bei denen der beanspruchte Gegenstand verwendet werden kann.

[0020] Wie die Erfinder in diesem Zusammenhang erkannt haben, besteht bei vielen Benutzern einer Anwendung wie DB2, einem Datenbank-Managementsystem (DBMS) von der International Business Machines Corporation in Armonk, New York, die Konformitätsanforderung, die Daten zu verschlüsseln. Heutige Systeme speichern häufig Passwörter in einer EFS-Datei (Encrypted File Systems – Systeme mit verschlüsselten Dateien), so dass ein DB2-Daemon ohne Bedienereingriff booten oder neu starten kann. Diese Anforderung ist bei geschäftskritischen Computersystemen und Anwendungen üblich. Das Speichern von Passwörtern in einer Datei ist jedoch grundsätzlich unsicher, es sei denn, die Datei ist mit einer starken Verschlüsselung geschützt. Die offenbarten Techniken stellen eine zusätzliche Sicherheitsschicht bereit.

[0021] In den Figuren handelt es sich bei **Fig.** 1 um ein Beispiel für eine Architektur **100** eines Datenverarbeitungssystems, in der der beanspruchte Gegenstand umgesetzt sein kann. Ein Datenverarbeitungssystem **102** enthält eine Zentraleinheit (CPU) **104**, die mit einem Monitor **106**, einer Tastatur **108** und einer Zeigevorrichtung oder „Maus" **110** verbunden ist, die gemeinsam ein menschliches Interagieren mit dem Datenverarbeitungssystem **102** und anderen Komponenten der Architektur **100** des Datenverarbeitungssystems ermöglichen. In dem Datenverarbeitungssystem **102** ist außerdem ein computerlesbares Speichermedium (CRSM) **112** enthalten, das an die CPU **104** angeschlossen und entweder im Datenverarbeitungssystem **102** enthalten (d. h. eine interne Einheit) oder extern an die CPU **104** mittels verschiedener handelsüblicher Verbindungseinheiten angeschlossen ist, zu denen ein (nicht gezeigter) USB-Port (Universal Serial Bus – universeller serieller Bus) gehört, ohne jedoch darauf beschränkt zu sein. Das CRSM **112** ist so dargestellt, dass es ein Betriebssystem (OS) **114**, ein Beispiel für einen Anwendungs-Daemon **116**, der in diesem Beispiel ein DB2-Daemon ist, ein sicheres Anwendungsstartladeprogramm (SASL) **118** und ein System verschlüsselter Dateien (EFS) **120** speichert, das einen Schlüssel oder „PKey" **122** enthält, der gemäß dem beanspruchten Gegenstand geschützt ist. Die Komponenten **116**, **118**, **120** und **122** und ihre Interaktion werden nachfolgend in Verbindung mit den **Fig.** 2 bis **Fig.** 5 genauer beschrieben.

[0022] Das Datenverarbeitungssystem **102** und die CPU **104** sind mit dem Internet **130** verbunden, das außerdem mit einem Server-Computer **132** verbunden ist. Obwohl in diesem Beispiel das Datenverarbeitungssystem **102** und der Server **231** über das Internet **130** für einen Datenaustausch verbunden sind, könnten sie auch über eine beliebige Anzahl von Datenübertragungsmedien verbunden sein, wie etwa ein (nicht gezeigtes) Lokalbereichsnetzwerk (LAN), ohne jedoch darauf beschränkt zu sein. An den Server **132** angeschlossen ist ein CRSM **134**, das so dargestellt ist, dass es eine Ressource R_1 **136** speichert, die in diesem Beispiel Daten entspricht, die gespeichert sind und auf die vom DB2-Daemon **116** zugegriffen wird. Es sollte angemerkt werden, dass es viele mögliche Konfigurationen des Datenverarbeitungssystems gibt, für die die Architektur **100** des Datenverarbeitungssystems lediglich ein einfaches Beispiel darstellt, und dass R_1 **136** lediglich ein Beispiel für eine Ressource darstellt, die gemäß dem beanspruchten Gegenstand geschützt sein kann. Zu weiteren Beispielen für geschützte Ressourcen könnten der Zugriff auf ein verschlüsseltes Netzwerk-Datenübertragungsprotokoll und der Zugriff auf einen sicheren LDAP-Server gehören, ohne jedoch darauf beschränkt zu sein.

[0023] **Fig.** 2 ist ein genaueres Blockschaubild des SASL **118**, das zuerst in **Fig.** 1 eingeführt wurde. Das SASL **118** enthält ein Eingabe/Ausgabemodul (E/A-Modul) **140**, ein Datenmodul **142**, ein Verschlüsselungsmodul **144**, ein Entschlüsselungsmodul **146**, ein Hash-Modul **148**, ein Installationsmodul **150** und ein Startmodul **152**. Für die Zwecke der folgenden Beispiele wird angenommen, dass Logik, die zu dem SASL **118** gehört, im Datenverarbeitungssystem **102** (**Fig.** 1) ausgeführt wird und im CRSM **112** (**Fig.** 1) gespeichert ist. Es sollte klar sein, dass der beanspruchte Gegenstand in vielen Typen von Datenverarbeitungssystemen und Datenspeicherstrukturen umgesetzt werden kann, der Einfachheit halber wird er jedoch lediglich in Bezug auf ein Datenverarbeitungssystem **102** und eine Systemarchitektur **100** (**Fig.** 1) beschrieben. Des Weiteren handelt es sich bei der Darstellung des SASL **118** in **Fig.** 2 um ein logisches Modell. Mit anderen Worten können die Komponenten **140**, **142**, **144**, **146**, **148**, **150** und **152** in derselben oder in separaten Dateien gespeichert sein und im System **100** entweder als ein einzelnes System oder

als separate Prozesse, die über beliebige verfügbare Interprozesskommunikationstechniken (IPC-Techniken) interagieren, geladen und/oder ausgeführt werden.

[0024] Das E/A-Modul **140** bewältigt den Datenaustausch, den das SASL **118** mit anderen Komponenten des Datenverarbeitungssystems **102** und der Architektur **100** hat. Bei dem Datenmodul **142** handelt es sich um eine Datenablage für Informationen, die das SASL **118** während des normalen Betriebs benötigt. Zu Beispielen für die Typen von Informationen, die im Datenmodul **142** gespeichert sind, gehören Anwendungsdaten **152**, Systemdaten **154**, Optionsdaten **156** und ein Arbeitszwischenspeicher **158**. Anwendungsdaten **152** speichern Informationen über eine mögliche Anwendung, wie etwa Bezeichnung, genutzte Ressourcen und Lage von Ressourcen, auf die zugegriffen werden soll. In dem nachfolgenden Beispiel speichern Anwendungsdaten **152** Informationen über den Anwendungs-Daemon **116** (**Fig.** 1), R_1 **136** (**Fig.** 1) und die Tatsache, dass auf R_1 **136** über den Server **132** (**Fig.** 1) zugegriffen wird. Systemdaten **154** speichern Informationen über andere Komponenten der Architektur **100**. Bei Systemdaten **154** kann es sich z. B. um Daten handeln, die einen Datenaustausch mit dem Server **132** ermöglichen. Zu Optionsdaten **156** gehören Informationen zu verschiedenen Benutzer- und administrativen Präferenzen. Der Arbeitszwischenspeicher **158** speichert die Ergebnisse laufender Berechnungen.

[0025] Das Verschlüsselungsmodul **144** führt unter Verwendung von Standard-Verschlüsselungstechniken einen Passwortschutz für verschiedene Datenelemente für das SASL **118** aus. Die Datenverschlüsselungsfunktion wird durch die folgende Formel dargestellt: PD (geschützte Daten) = Verschlüsseln (Schlüssel, Daten). Das Entschlüsselungsmodul **146** entschlüsselt zuvor geschützte Daten. Die Datenentschlüsselung wird durch die folgende Formel dargestellt: Daten = Entschlüsseln (Schlüssel, PD). Das Hash-Modul **148** erzeugt Hash-Codes, die von den Verschlüsselungs- und Entschlüsselungsmodulen **144** und **146** verwendet werden, um Daten zu verschlüsseln und zu entschlüsseln, insbesondere Schlüssel für verschiedene Ressourcen wie etwa R_1 **136**.

[0026] Das Installationsmodul **150** ermöglicht die Installation von Anwendungen bei diesem Beispiel in dem Datenverarbeitungssystem **102** (siehe **200**, **Fig.** 3). Das Startmodul **152** ist zuständig für die automatische Ausführung von Anwendungs-Daemons im Datenverarbeitungssystem **102** (siehe **240**, **Fig.** 4). Wie nachfolgend in Verbindung mit **Fig.** 4 dargestellt, können einige Anwendungen wie in jedem üblichen System gestartet werden. Andere Anwendungen wie etwa der Anwendungs-Daemon **116**, die auf geschützte Ressourcen zugreifen, werden gemäß der offenbarten Technologie geladen. Eine Feststellung, ob eine Anwendung in üblicher Weise oder gemäß dem beanspruchten Gegenstand geladen wird, erfolgt anhand von Informationen, die in den Anwendungsdaten **152** gespeichert sind. Die Funktionen und Interaktionen von Modulen **140**, **142**, **144**, **146**, **148**, **150** und **152** werden nachfolgend in Verbindung mit den **Fig.** 3 bis **Fig.** 4 genauer beschrieben.

[0027] **Fig.** 3 ist ein Ablaufplan eines Prozesses **200** „Installiere Anwendung", der Aspekte des beanspruchten Gegenstands umsetzen kann. Bei diesem Beispiel ist Logik, die zu dem Prozess **200** gehört, im CRSM **112** (**Fig.** 1) in Verbindung mit dem SASL **118** (**Fig.** 1 und **Fig.** 2) gespeichert und wird in einem oder mehreren Prozessoren (nicht gezeigt) der CPU **104** (**Fig.** 1) und im Datenverarbeitungssystem **102** (**Fig.** 1) ausgeführt.

[0028] Der Prozess **200** beginnt in einem Block **202** „Beginne mit Installieren der Anwendung" und geht sofort zu einem Block **204** „Empfange Anwendung" über. Während der Verarbeitung, die zu dem Block **204** gehört, wird die Anwendung für eine Installation im Datenverarbeitungssystem **102** empfangen. Es werden keine Annahmen gemacht, wie die Anwendung empfangen wird. Mit anderen Worten kann die Anwendung heruntergeladen, auf einem tragbaren Speichermedium (nicht gezeigt) oder auf jede andere bekannte oder noch nicht bekannte Weise empfangen werden. Einem Fachmann sollten die vielen unterschiedlichen Arten bekannt sein, wie eine Anwendung für eine Installation in einem Datenverarbeitungssystem empfangen werden kann.

[0029] Während der Verarbeitung, die zu einem Block **206** „Speichere Anwendung" gehört, wird Logik, die zu einer ausführbaren Version der Anwendung gehört, die während der zu dem Block **204** gehörenden Verarbeitung empfangen wird und bei der es sich in diesem Beispiel um einen Anwendungs-Daemon **116** (**Fig.** 1) handelt, im CRSM **112** gespeichert. Während der Verarbeitung, die zu einem Block **208** „Abrufen eines ersten Schlüssels und eines ersten Hash-Werts" gehört, werden ein Zugangsschlüssel, der einer Ressource entspricht, bei der es sich in diesem Beispiel um R_1 **136** (**Fig.** 1) handelt, und ein Hash-Code, der auf der ausführbaren Datei beruht und erzeugt wird, wenn die Anwendung kompiliert wird, abgerufen. Der Hash-Code wird üblicherweise in Verbindung mit der Anwendung übertragen. Ein Administrator, der die Anwendung installiert, würde üblicherweise Zugriff auf diese Informationen haben.

[0030] Während einer Verarbeitung, die zu einem Block **210** „Verschlüssele ersten Schlüssel mit erstem Hash-Wert" gehört, wird ein geschützter Schlüssel, bei dem es sich in diesem Beispiel um pkey **122** (**Fig.** 1) handelt, gemäß der folgenden Formel er-

zeugt: pkey = Verschlüsseln (erster Hash-Wert, erster Schlüssel). Mit anderen Worten wird pkey **122** erzeugt, indem der Schlüssel mit dem Hash-Code, der beim Kompilieren des Anwendungs-Daemon **116** erzeugt wird, zu R_1 **136** verschlüsselt wird. Während einer Verarbeitung, die zu einem Block **212** „Speichere geschützten Schlüssel" gehört, wird pkey **122**, der während einer zu dem Block **210** gehörenden Verarbeitung erzeugt wurde, in der EFS-Datei **120** (**Fig.** 1) gespeichert. Es sollte angemerkt werden, dass üblicherweise weder der ursprüngliche Schlüssel für R_1 **136** noch der Hash-Code, der mit dem Anwendungs-Daemon **116** übertragen wird, gespeichert werden. Auf diese Weise muss jeder Teilnehmer, der den Anwendungs-Daemon **116** zur Ausführung in die CPU **104** lädt, in der Lage sein, pkey **122** zu entschlüsseln, um auf R_1 **136** zugreifen zu können.

[0031] Während einer Verarbeitung, die zu dem Block **214** „Speichere Anwendungsinformationen" gehört, werden Informationen über den Anwendungs-Daemon **116** in Anwendungsdaten **152** gespeichert, zu denen Daten, die ermöglichen, dass das SASL **118** den Anwendungs-Daemon **116** mit dem pkey **122** korreliert, und eine Angabe, dass der Anwendungs-Daemon **116** gemäß dem beanspruchten Gegenstand geschützt ist, gehören, ohne jedoch darauf beschränkt zu sein. Die Steuerung geht schließlich zu einem Block **219** „Beende installieren der Anwendung" über, in dem der Prozess **200** endet.

[0032] **Fig.** 4 ist ein Ablaufplan eines Prozesses **240** „Beginne Anwendung", der Aspekte des beanspruchten Gegenstands umsetzen kann. Bei diesem Beispiel wird Logik, die zu dem Prozess **240** gehört, im CRSM **112** (**Fig.** 1) in Verbindung mit dem SASL **118** (**Fig.** 1 und **Fig.** 2) gespeichert und in einem oder mehreren Prozessoren (nicht gezeigt) der CPU **104** (**Fig.** 1) und im Datenverarbeitungssystem **102** (**Fig.** 1) ausgeführt. Der Prozess **240** kann nach einem Neubooten oder einem automatischen Neubooten des Datenverarbeitungssystems **102** ausgelöst werden oder nachdem ein bestimmtes Ereignis entweder beabsichtigt oder unbeabsichtigt bewirkt hat, dass eine vorher ausgelöste ausführbare Datei des Anwendungs-Daemon **116** (**Fig.** 1) angehalten wird. Im Allgemeinen vermeidet der Prozess **240** die Notwendigkeit eines manuellen Eingriffs, wenn der Anwendungs-Daemon **116** ausgelöst ist, und eliminiert insbesondere die Notwendigkeit, dass ein Bediener ein Passwort eingeben muss, damit die Anwendung mit Zugriff auf erforderliche verschlüsselte Dateien und Ressourcen gestartet werden kann. Die offenbarte Technologie eliminiert außerdem die Notwendigkeit, dass ungeschützte Schlüssel für derartige erforderliche Dateien und Ressourcen in dem System gespeichert werden müssen.

[0033] Der Prozess **240** beginnt in einem Block **242** „Beginne mit Starten der Anwendung" und geht sofort zu einem Block **244** „Empfange Startanforderung" über. Während der zu dem Block **244** gehörenden Verarbeitung wird eine Anforderung zum Starten einer Anwendung empfangen, bei der es sich in diesem Beispiel um den Anwendungs-Daemon **116** handelt. Wie oben erläutert kann eine derartige Anforderung von dem Datenverarbeitungssystem **102** im Fall eines Neubootens oder der Erfassung, dass eine zuvor ausgelöste Kopie des Anwendungs-Daemon **116** angehalten wurde, automatisch erzeugt werden. Während einer Verarbeitung, die zu einem Block **246** „Lade Anwendung" gehört, wird der Anwendungs-Daemon **116** zur Ausführung in die CPU **104** geladen.

[0034] Während einer Verarbeitung, die zu einem Block **248** „Ressource(n) geschützt?" gehört, erfolgt eine Feststellung, ob der Anwendungs-Daemon **116** einen Zugriff auf verschlüsselte Dateien oder Ressourcen gemäß dem beanspruchten Gegenstand fordert. Eine derartige Feststellung erfolgt anhand von Informationen, die in Verbindung mit Anwendungsdaten **152** (**Fig.** 2) gespeichert sind. Wenn festgestellt wird, dass der Anwendungs-Daemon **116** einen Zugriff auf verschlüsselte Dateien oder Ressourcen gemäß dem beanspruchten Gegenstand fordert, geht die Steuerung zum Block **250** „Erzeuge zweiten Hash-Schlüssel" über. Während der Verarbeitung, die zu dem Block **250** gehört, wird ein zweiter Hash-Code (SH) (siehe **148**, **Fig.** 2) anhand des Abbilds des Anwendungs-Daemon **116** erzeugt, das während der zu dem Block **246** gehörenden Verarbeitung geladen wurde.

[0035] Während der Verarbeitung, die zu dem Block **252** „Leite ersten Schlüssel ab" gehört, wird der Hash-Code, der während der zu dem Block **250** gehörenden Verarbeitung erzeugt wurde, als Schlüssel zum Entschlüsseln (siehe **146**, **Fig.** 2) von pkey **122** (siehe **210**, **212**, **Fig.** 3) verwendet, um anhand der folgenden Formel: SK = Entschlüsseln (SH, pkey **122**) einen zweiten Schlüssel (SK) zu erzeugen. Während der Verarbeitung, die zu einem Block **254** „Greife auf Ressource zu" gehört, wird der während der zu dem Block **252** gehörenden Verarbeitung erzeugte zweite Schlüssel verwendet, um auf R_1 **136** zuzugreifen. Es sollte angemerkt werden, dass, wenn der zweite Hash-Code mit dem ersten Hash-Code übereinstimmt (siehe **208**, **Fig.** 3) pkey **122** entschlüsselt und der ursprüngliche Schlüssel (siehe **208**, **Fig.** 3) für die entsprechenden geschützten Dateien oder Ressourcen erzeugt werden kann. Es sollte außerdem angemerkt werden, dass, wenn der Anwendungs-Daemon **116** modifiziert wurde, der zweite Hash-Code nicht mit dem ersten Hash-Code übereinstimmen und die Entschlüsselung während der zu dem Block **252** gehörenden Verarbeitung nicht den richtigen Schlüssel für die verschlüsselte Datei oder Ressource erzeugen wird.

[0036] Während der Verarbeitung, die zu einem Block **256** „Zugriff gestattet?" gehört, wird festgestellt, ob auf die geschützten Dateien oder Ressourcen zugegriffen werden konnte. Wenn das nicht der Fall ist, geht die Steuerung zu einem Block **258** „Löse Ausnahme aus" über. Während der zu dem Block **258** gehörenden Verarbeitung werden geeignete Maßnahmen getroffen. Zu derartigen Maßnahmen gehören das Übertragen einer Benachrichtigung zu einem Administrator des Datenverarbeitungssystems **102** oder dem Anwendungs-Daemon **116**, ohne jedoch darauf beschränkt zu sein. Nachdem festgestellt wurde, dass keine geschützten Dateien oder Ressourcen an der zu dem Block **248** gehörenden Verarbeitung beteiligt sind, geht die Steuerung, nachdem festgestellt wurde, dass während der zu dem Block **256** gehörenden Verarbeitung ein Zugriff gestattet oder während der zu dem Block **258** gehörenden Verarbeitung eine Ausnahme ausgelöst wurde, schließlich zu einem Block **259** „Beende Starten der Anwendung" über, in dem der Prozess **240** endet.

[0037] Die in diesem Zusammenhang verwendete Terminologie dient lediglich dem Zweck des Beschreibens bestimmter Ausführungsformen und soll die Erfindung nicht einschränken. Die hier verwendeten Singularformen „ein/e" und „der/die/das" sollen ebenfalls die Pluralformen einschließen, falls im Kontext nicht ausdrücklich anders angegeben. Es ist ferner klar, dass die Ausdrücke „aufweist" und/oder „aufweisend" bei Verwendung in dieser Patentschrift das Vorhandensein von genannten Merkmalen, Ganzzahlen, Schritten, Operationen, Elementen und/oder Komponenten angeben, jedoch nicht das Vorhandensein oder Hinzufügen eines oder mehrerer weiterer Merkmale, Ganzzahlen, Schritte, Operationen, Elemente, Komponenten und/oder Gruppen hiervon ausschließen.

[0038] Die entsprechenden Strukturen, Materialien, Handlungen und Entsprechungen aller Mittel oder Schritt-plus-Funktion-Elemente in den nachfolgenden Ansprüchen sollen jede Struktur, jedes Material oder jede Handlung zum Ausführen der Funktion in Kombination mit anderen beanspruchten Elementen, wie speziell beansprucht, enthalten. Die Beschreibung der vorliegenden Erfindung wird für Zwecke der Erläuterung und Beschreibung präsentiert und soll nicht vollständig sein oder die Erfindung auf die offenbarte Form einschränken. Viele Modifikationen und Variationen werden für Fachleute ersichtlich sein, ohne vom Schutzumfang und Gedanken der Erfindung abzuweichen. Die Ausführungsform wurde gewählt und beschrieben, um die Grundgedanken der Erfindung und die praktische Anwendung bestmöglich zu erklären und andere Fachleute zu befähigen, die Erfindung zu verstehen und auf verschiedene Ausführungsformen mit diversen Modifikationen, die für die speziell vorgesehene Verwendung geeignet sind, anzuwenden.

[0039] Der Ablaufplan und die Blockschaubilder in den Figuren veranschaulichen die Architektur, die Funktionalität und den Betrieb von möglichen Umsetzungen von Systemen, Verfahren und Computerprogrammprodukten gemäß verschiedenen Ausführungsformen der vorliegenden Erfindung. In diesem Zusammenhang kann jeder Block in dem Ablaufplan oder den Blockschaubildern ein Modul, ein Segment oder einen Abschnitt von Code darstellen, der einen oder mehrere ausführbare Befehle zum Umsetzen der spezifizierten logischen Funktion(en) aufweist. Es sollte außerdem angemerkt werden, dass bei einigen alternativen Umsetzungen die in dem Block angegebenen Funktionen nicht in der in den Figuren angegebenen Reihenfolge auftreten können. Zum Beispiel können zwei Blöcke, die nacheinander gezeigt sind, tatsächlich im Wesentlichen gleichzeitig ausgeführt werden, oder die Blöcke können gelegentlich in Abhängigkeit von der beteiligten Funktionalität in der umgekehrten Reihenfolge ausgeführt werden. Es wird außerdem angemerkt, dass jeder Block der Blockschaubilder und/oder der Ablaufplandarstellung und Kombinationen von Blöcken in den Blockschaubildern und/oder der Ablaufplandarstellung durch auf spezielle Hardware gestützte Systeme, die die spezifizierten Funktionen oder Handlungen ausführen, oder Kombinationen aus spezieller Hardware und Computerbefehlen umgesetzt werden können.

Patentansprüche

1. Verfahren, das aufweist:
Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel;
zu einem Installationszeitpunkt der Anwendung Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört;
Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen;
zu einem Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon;
Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und
Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des abgeleiteten ersten Schlüssels.

2. Verfahren nach Anspruch 1, das ferner das Speichern des geschützten ersten Schlüssels in einer Datei von Systemen verschlüsselter Dateien (EFS) aufweist.

3. Verfahren nach Anspruch 1, das ferner aufweist:
Vergleichen des ersten Hash-Codes mit dem zweiten Hash-Code; und

Verhindern eines Ladens der Anwendung als Reaktion auf eine fehlende Übereinstimmung zwischen dem ersten Hash-Code und dem zweiten Hash-Code.

4. Verfahren nach Anspruch 1, wobei es sich bei der Anwendung um einen Datenbank-Server und bei der Ressource um eine passwortgeschützte Datenbank handelt.

5. Verfahren nach Anspruch 1, wobei es sich bei der Ressource um eine passwortgeschützte Netzwerk-Datenübertragungseinheit handelt.

6. Verfahren nach Anspruch 1, wobei es sich bei der Ressource um einen LDAP-Server (Lightweight Directory Access Protocol) handelt.

7. Verfahren nach Anspruch 1, das ferner aufweist:
Erfassen, dass eine Abschaltung der Anwendung aufgetreten ist; und
Auslösen des Erzeugens des zweiten Hash-Codes, des Ableitens des ersten Schlüssels und des Zugreifens auf die Ressource als Reaktion auf das Erfassen.

8. Vorrichtung, die aufweist:
einen Prozessor;
eine computerlesbare Speichereinheit; und
Logik, die in dem computerlesbaren Medium gespeichert ist und in dem Prozessor ausgeführt wird, zum:
Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel;
zu einem Installationszeitpunkt der Anwendung Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört;
Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen;
zu einem Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon;
Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und
Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des abgeleiteten ersten Schlüssels.

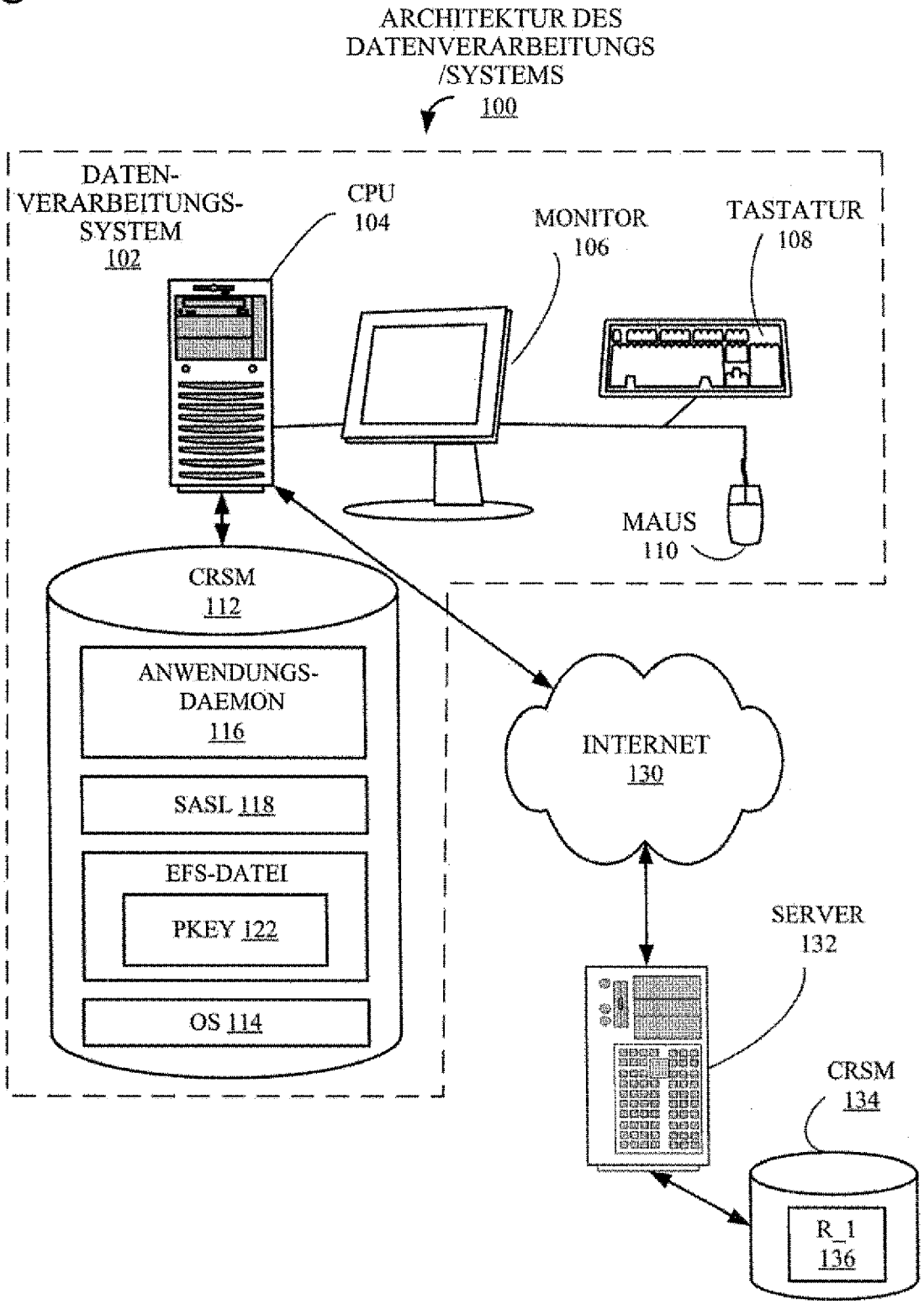
9. Computerprogrammprodukt, das aufweist:
eine computerlesbare Speichereinheit; und
Logik, die zum Ausführen in einem Prozessor in dem computerlesbaren Medium gespeichert ist, zum:
Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel;
zu einem Installationszeitpunkt der Anwendung Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört;
Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen;
zu einem Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon;
Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und
Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des abgeleiteten ersten Schlüssels.

10. Automatisierte Anwendungsstarteinheit, die aufweist:
einen Prozessor;
eine computerlesbare Speichereinheit, die mit dem Prozessor verbunden ist; und
Logik, die in dem computerlesbaren Medium gespeichert ist und in dem Prozessor ausgeführt wird, zum:
Verschlüsseln einer Ressource, die zu einer Anwendung gehört, mit einem ersten Schlüssel;
zu einem Installationszeitpunkt der Anwendung Erzeugen eines ersten Hash-Codes für einen ausführbaren Daemon, der zu der Anwendung gehört;
Verschlüsseln des ersten Schlüssels mit dem ersten Hash-Code, um einen geschützten ersten Schlüssel zu erzeugen;
zum Ladezeitpunkt einer Anwendung Erzeugen eines zweiten Hash-Codes für den ausführbaren Daemon;
Ableiten des ersten Schlüssels durch Entschlüsseln des geschützten ersten Schlüssels unter Verwendung des zweiten Hash-Codes, um einen abgeleiteten ersten Schlüssel zu erzeugen; und
Zugreifen auf die Ressource durch den ausführbaren Daemon durch Verwenden des abgeleiteten ersten Schlüssels.

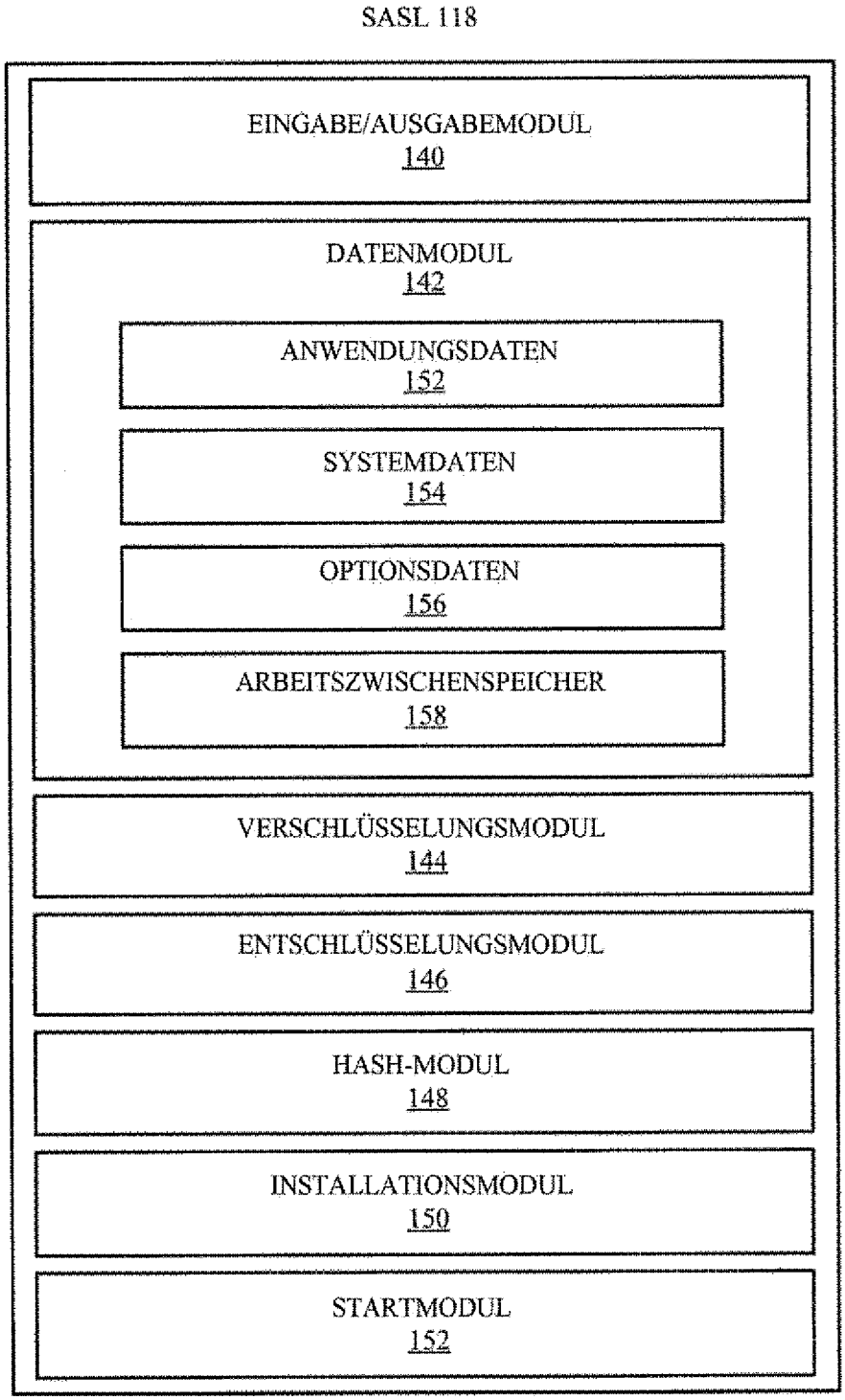
Es folgen 4 Seiten Zeichnungen

Anhängende Zeichnungen

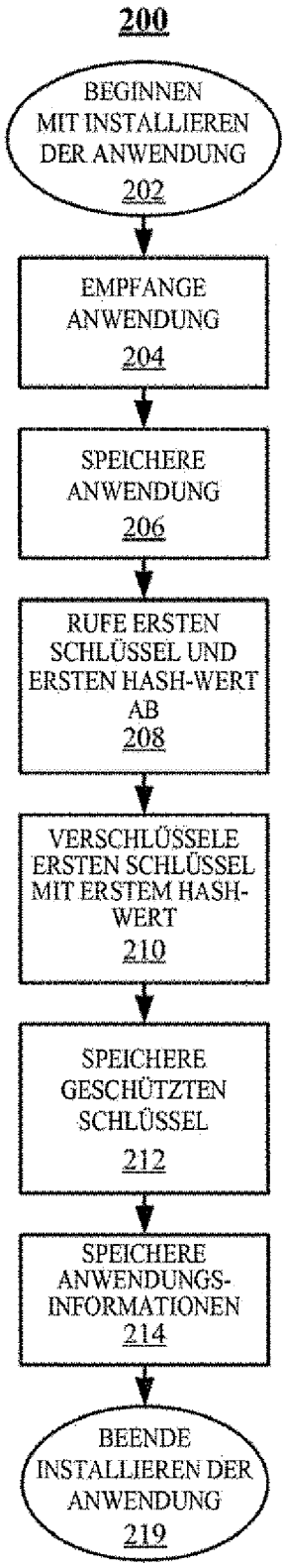
Figur 1



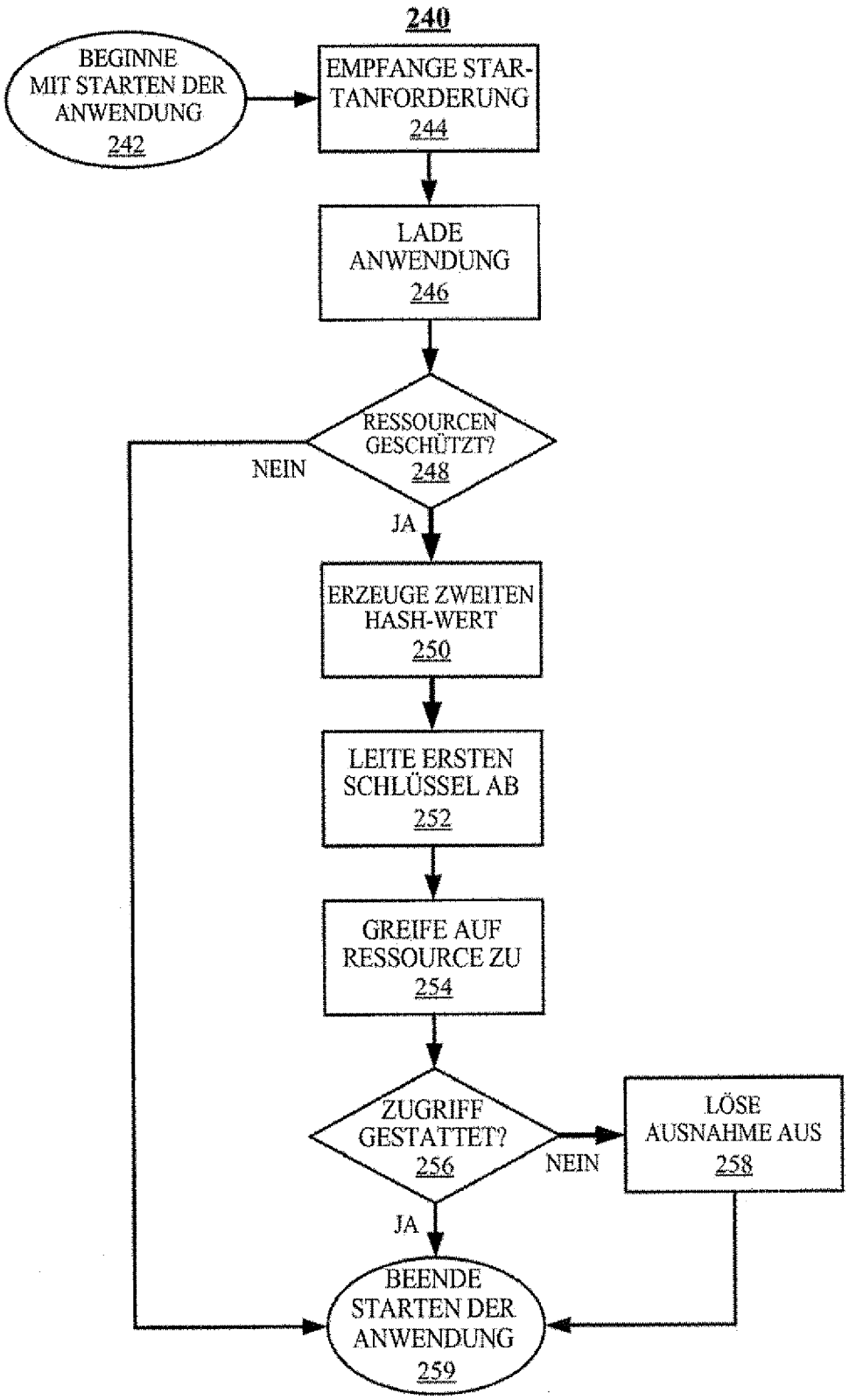
Figur 2


SASL 118
EINGABE/AUSGABEMODUL
140
DATENMODUL
142
ANWENDUNGSDATEN
152
SYSTEMDATEN
154
OPTIONSDATEN
156
ARBEITSZWISCHENSPEICHER
158
VERSCHLÜSSELUNGSMODUL
144
ENTSCHLÜSSELUNGSMODUL
146
HASH-MODUL
148
INSTALLATIONSMODUL
150
STARTMODUL
152

Figur 3


200
BEGINNEN MIT INSTALLIEREN DER ANWENDUNG
202
EMPFANGE ANWENDUNG
204
SPEICHERE ANWENDUNG
206
RUFE ERSTEN SCHLÜSSEL UND ERSTEN HASH-WERT AB
208
VERSCHLÜSSELE ERSTEN SCHLÜSSEL MIT ERSTEM HASH-WERT
210
SPEICHERE GESCHÜTZTEN SCHLÜSSEL
212
SPEICHERE ANWENDUNGS-INFORMATIONEN
214
BEENDE INSTALLIEREN DER ANWENDUNG
219

Figur 4


240
BEGINNE MIT STARTEN DER ANWENDUNG 242
EMPFANGE STARTANFORDERUNG 244
LADE ANWENDUNG 246
RESSOURCEN GESCHÜTZT? 248
NEIN
JA
ERZEUGE ZWEITEN HASH-WERT 250
LEITE ERSTEN SCHLÜSSEL AB 252
GREIFE AUF RESSOURCE ZU 254
ZUGRIFF GESTATTET? 256
NEIN
LÖSE AUSNAHME AUS 258
JA
BEENDE STARTEN DER ANWENDUNG 259