



(12)

Patentschrift

(21) Aktenzeichen: **10 2016 110 479.5**
(22) Anmeldetag: **07.06.2016**
(43) Offenlegungstag: **07.12.2017**
(45) Veröffentlichungstag
der Patenterteilung: **04.05.2023**

(51) Int Cl.: **G06F 16/13 (2019.01)**
G06F 16/901 (2019.01)
G06F 21/50 (2013.01)
G06F 11/30 (2006.01)

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:
Bundesrepublik Deutschland, vertreten durch das Bundesministerium für Wirtschaft und Energie, dieses vertreten durch den Präsidenten der Physikalischen Bundesanstalt, 38116 Braunschweig, DE; Technische Universität Berlin, 10623 Berlin, DE; Technische Universität Dortmund, 44227 Dortmund, DE

(72) Erfinder:
Peters, Daniel, 10555 Berlin, DE; Thiel, Florian, Dr., 14612 Falkensee, DE; Fischer, Johannes, Prof. Dr., 44137 Dortmund, DE; Seifert, Jean-Pierre, Prof. Dr., 10587 Berlin, DE

(74) Vertreter:
Gramm, Lins & Partner Patent- und Rechtsanwälte PartGmbH, 30173 Hannover, DE

(56) Ermittelte Stand der Technik:
siehe Folgeseiten

(54) Bezeichnung: **Verfahren und Computerprogramm zum Überprüfen der Dateisystemintegrität sowie Datenverarbeitungseinrichtung hierzu**

(57) Hauptanspruch: Verfahren zum Überprüfen der Dateisystemintegrität einer Dateiverwaltung einer Datenverarbeitungseinrichtung, wobei die Dateiverwaltung der Datenverarbeitungseinrichtung eine Mehrzahl von Dateien für die Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung enthält, gekennzeichnet durch die elektronisch ausführbaren Schritte:

- Bereitstellen einiger oder aller in der Dateiverwaltung enthaltenen Dateien in einer baumartigen Datenstruktur, bei der jeder Knoten der baumartigen Datenstruktur eine Datei der Dateiverwaltung repräsentiert,

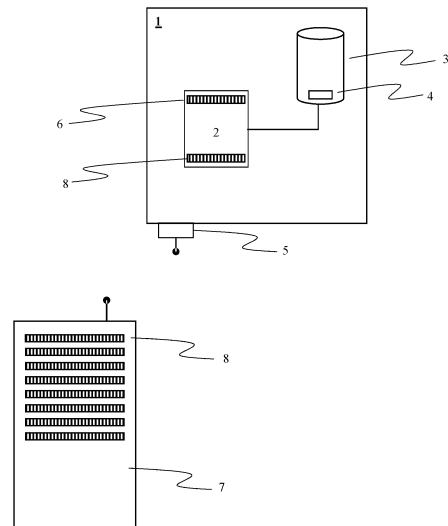
- Erstellen einer sequentiellen Datenstruktur aus der baumartigen Datenstruktur, indem

○ eine Breitensuche in der baumartigen Datenstruktur durchgeführt wird, bei der nacheinander für jede Bauebene sequentiell durch die Knoten der jeweiligen Bauebene der baumartigen Datenstruktur iteriert wird,

○ wobei für jedena Knoten ein Dateityp der durch den jeweiligen Knoten repräsentierten Datei ermittelt und eine den ermittelten Dateityp repräsentierende Typcodierung aus einer Liste von verschiedenen Codierungen ausgewählt wird, und

○ wobei die ermittelten Typcodierungen der einzelnen durch die Knoten repräsentierten Dateien nacheinander in ein sequentielles Dateityp-Datenfeld (S) der sequentiellen Datenstruktur eingefügt werden, wobei die Positionen (i) derjenigen in dem sequentiellen Dateityp-Datenfeld (S) enthaltenen Typcodierungen durch eine Anfangs- oder

Endcodierung in der sequentiellen Datenstruktur gekennzeichnet werden, deren zugrundeliegenden Dateien in der baumartigen Datenstruktur jeweils einem Anfangs- oder Endknoten einer Reihe von Knoten entsprechen, die alle einen gemeinsamen Elternknoten haben,
- Vergleichen der erstellten sequentiellen Datenstruktur oder einer mathematischen Repräsentation hiervon mit einer ...



(56) Ermittelter Stand der Technik:

JACOBSON, Guy. Space-efficient static trees and graphs. In: Foundations of Computer Science, 1989., 30th Annual Symposium on. IEEE, 1989. S. 549-554. doi: 10.1109/SFCS.1989.63533

KIM, Gene H.; SPAFFORD, Eugene H. The design and implementation of tripwire: A file system integrity checker. In: Proceedings of the 2nd ACM Conference on Computer and Communications Security. ACM, 1994. S. 18-29. doi: 10.1145/191177.191183

Beschreibung

[0001] Die Erfindung betrifft ein Verfahren zum Überprüfen der Dateisystemintegrität einer Dateiverwaltung einer Datenverarbeitungseinrichtung, wobei die Dateiverwaltung der Datenverarbeitungseinrichtung eine Mehrzahl von Dateien für die Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung enthält.

[0002] In der klassischen von-Neumann Architektur einer elektronischen Datenverarbeitungseinrichtung ist die Speicherung von Daten und Informationen von der eigentlichen Datenverarbeitung, ausgeführt meist mithilfe einer mikroprozessorgesteuerten oder mikrocontrollergesteuerten Datenverarbeitungseinheit, getrennt. Die für die Datenverarbeitung notwendigen Daten und Informationen werden hierfür in einem digitalen Speichermedium abgespeichert, wobei für die Abspeicherung der Daten und Informationen mithilfe von Bits als kleinsten Informationsträgern digitale Verwaltung notwendig ist, die das Auffinden der relevanten Daten und Informationen auf dem digitalen Speichermedium ermöglicht und somit einen sicheren und zielgerichteten Datenabruf aus dem digitalen Speichermedium sicherstellt.

[0003] In der Regel werden die Daten und Informationen auf dem digitalen Speichermedium mithilfe von logischen Datencontainern, den Dateien, abgelegt. Hierdurch lassen sich die Daten und Informationen logisch strukturiert auf dem digitalen Speichermedium ablegen, wodurch das Auffinden der benötigten Daten und Informationen erleichtert wird.

[0004] Unter einer Datei im Sinne der vorliegenden Erfindung wird dabei jede logische Datenstruktur verstanden, mit der Daten und Informationen geordnet auf dem digitalen Speichermedium abgelegt werden können. Der Begriff „Datei“ ist somit im Sinne der vorliegenden Erfindung als Oberbegriff zu verstehen und beinhaltet insbesondere Dateien als Speichercontainer für Daten und Informationen, Programmdateien zum Ausführen von Computerprogrammen, Ordner bzw. Verzeichnisse, Verweise bzw. Links auf andere Dateien (sowohl Hard- als auch Softlinks), Netzwerkpfade, blockorientierte sowie zeichenorientierte Gerätetypen. Als Dateien kommen dabei auch Sockets, Pipes sowie Dateien der sogenannten MIME-Typen. Diese Liste ist nicht abschließend.

[0005] Eine Dateiverwaltung einer Dateiverarbeitungseinrichtung organisiert nun die einzelnen Dateien auf dem entsprechenden digitalen Speichermedium und ermöglicht somit den Zugriff auf die Informationen und Daten einer Datei, die auf dem digitalen Speichermedium abgelegt ist. Die Dateiverwaltung bietet somit eine Schnittstelle zwischen dem digitalen Speichermedium einerseits und dem Benutzerzugriff andererseits, wobei der Benutzer oder ein anderes Programm von der genauen Speicherarchitektur und Speicherorganisation auf dem digitalen Speichermedium keine Kenntnis haben muss. Vielmehr wird mithilfe der Dateiverwaltung bzw. des Dateiverwaltungssystems sichergestellt, dass für jede auf dem digitalen Speichermedium abgespeicherte Datei bekannt ist, unter welcher Adresse in dem digitalen Speichermedium die entsprechenden Dateninformationen bezüglich der abzurufenden Datei hinterlegt sind.

[0006] Dabei wird in der Regel eine baumorientierte bzw. baumartige Dateistruktur geschaffen, um so eine logische Organisationsstruktur bereitzustellen, die ein leichtes Auffinden der Dateien ermöglicht. Baumorientiert bzw. baumartig meint hierbei, dass jede Datei als Knoten abgebildet werden kann, wobei jeder Knoten genau einen übergeordneten Elternknoten hat, sofern er nicht der Wurzelknoten (root) ist.

[0007] Je nach Anwendungsfall der Datenverarbeitungseinrichtung, auf der die Dateien mit Hilfe der Dateiverwaltung organisiert sind, ist es wünschenswert, wenn die Dateisystemintegrität überprüfbar ist, so dass festgestellt werden kann, ob die Dateisystemstruktur einer Dateiverwaltung einer Datenverarbeitungseinrichtung in ihrem Kern verändert wurde, um so beispielsweise Manipulationen an solchen Geräten erkennen zu können. Gerade Software von Geräten, die für amtliche Zwecke im gewerblichen Gebrauch oder im öffentlichen Interesse zum Einsatz kommen, wie beispielsweise Mess- und Steuergeräte, müssen in festgelegten Abständen auf Integrität bezüglich ihrer Software geprüft werden. Bei gesetzlich geregelten Messgeräten, wie beispielsweise Verkehrsmessgeräten, Waagen und anderen geeichten Messgeräten geschieht dies durch benannte Stellen und der Marktüberwachung (Landeseichbehörden). Die benannte Stelle überprüft die Software vor dem In-Verkehr-Bringen und die Marktüberwachung überprüft zyklisch, ob sich auf den Geräten noch dieselbe Software befindet, die von einer Konformitätsbewertungsstelle, z.B. der PTB, zugelassen wurde. Dies ist insbesondere für alle Geräte in kritischen Infrastrukturen nötig, um Manipulation und Fehlverhalten rechtzeitig zu erkennen.

[0008] Ein Hauptproblem heutiger Systeme ist dabei, dass die Überprüfung der Dateiintegrität auf dem gleichen System stattfindet, welches die Dateien auch verwaltet. Dabei wird einem System vertraut, von dem

man die Vertrauenswürdigkeit abfragen möchte, mithin das System überprüft sich also selbst. Dies birgt insofern Risiken, weil intelligente Schadsoftware die Eigenschaft besitzt, auch die Berechnung und Überprüfung der Dateisystemintegrität zu manipulieren und somit Änderungen an Dateien zu kaschieren.

[0009] Aus KIM, Gene H.; SPAFFORD, Eugene H. The design and implementation of tripwire: A file system integrity checker. In: Proceedings of the 2nd ACM Conference on Computer and Communications Security. ACM, 1994. S. 18-29. Doi: 10.1145/191177.191183 ist ein Design und eine Implementierung des UNIX Tools Tripwire bekannt, mit dem sich die Systemintegrität einer Dateistruktur überprüfen lässt.

[0010] Aus JACOBSON, Guy. Space-efficient static trees and graphs. In: Foundations of Computer Science, 1989., 30th Annual Symposium on. IEEE, 1989. S. 549-554. doi: 10.1109/SFCS. 1989.63533 ist eine mathematische Repräsentation von Graphen bekannt, bei der mithilfe einer Breitensuche der Graph sequenzielliert werden soll.

[0011] Es ist daher eine Aufgabe der vorliegenden Erfindung ein verbessertes Verfahren und Vorrichtung anzugeben, mit dem die Dateisystemintegrität einer Dateiverwaltung einer insbesondere mobilen Dateiverarbeitungseinrichtung schnell und sicher überprüft werden kann.

[0012] Die Aufgabe wird mit dem Verfahren gemäß Anspruch 1, dem Computerprogramm gemäß Anspruch 11 sowie der Datenverarbeitungseinrichtung gemäß Anspruch 12 erfindungsgemäß gelöst.

[0013] Gemäß Anspruch 1 wird ein Verfahren zum Überprüfen der Dateisystemintegrität einer Dateiverwaltung einer Datenverarbeitungseinrichtung vorgeschlagen, wobei zumindest ein Teil der Dateien oder auch alle Dateien, die zur Bereitstellung der Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung vorgesehen sind, auf der Datenverarbeitungseinrichtung in einem digitalen Speichermedium hinterlegt und abgespeichert sind. Mittels einer Dateiverwaltung können werden die Dateien organisiert, so dass die Dateiverwaltung die für die Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung notwendigen Dateien enthält. Hierunter wird insbesondere verstanden, dass die Dateiverwaltung die jeweiligen Dateien kennt und den Inhalt der Dateien adressieren kann. Die Dateiverwaltung, die auch als Dateisystem bezeichnet wird, ordnet und strukturiert somit die Dateien so, dass mithilfe des Dateinamens ein Zugriff auf die auf dem digitalen Speichermedium abgespeicherten oder abzuspeichernden Dateien möglich ist. Die Dateiverwaltung kennt somit sämtliche, auf dem digitalen Speichermedium der Dateiverarbeitungseinrichtung abgespeicherten oder abzuspeichernden Dateien, wobei hierunter auch jene Dateien zu verstehen sind, die erst bei Anforderung von einem externen Speichermedium auf die Datenverarbeitungseinrichtung geladen werden. Hierdurch lassen sich Teile der Dateien der Dateiverwaltung auf externe Speichermedien auslagern.

[0014] Unter einer Dateisystemstruktur wird des Weiteren eine Abbildung der Eigenschaften und der Organisationsstruktur der abgespeicherten Dateien verstanden, sowie sie mithilfe der Dateiverwaltung auf dem digitalen Speichermedium abgespeichert sind. Die Dateisystemstruktur beinhaltet somit nicht nur mögliche Eigenschaften von Dateien, wie beispielsweise der Dateiname, die Größe oder der Dateityp, sondern kann auch die logische Organisationsstruktur, in der die Dateien logisch strukturiert sind, enthalten. Dies beinhaltet insbesondere die Frage, welche Dateien in welchen Ordnern abgelegt oder welche Dateien auf welche Dateien verlinken.

[0015] Zur Überprüfung der Dateisystemintegrität der in der Dateiverwaltung enthaltenen Dateien der Datenverarbeitungseinrichtung werden zunächst einige oder alle in der Dateiverwaltung enthaltenen Dateien in einer baumartigen Datenstruktur bereitgestellt, bei der jeder Knoten der baumartigen Datenstruktur eine Datei der Dateiverwaltung repräsentiert. Eine solche baumartige Datenstruktur wird dabei in der Regel von der Dateiverwaltung bereitgestellt, beispielsweise in Form einer vollständigen Baumstruktur oder durch Abfrage einzelner Dateien. Da die Dateiverwaltung letztlich die vollständige Kenntnis über die abgespeicherten Dateien besitzt, ist es vorteilhaft, wenn eben auch die Dateiverwaltung die entsprechende baumartige Datenstruktur der abgespeicherten Dateien im weiteren Prozess bereitstellt.

[0016] Anschließend wird eine sequentielle Datenstruktur aus der baumartigen Datenstruktur erstellt, in dem eine Breitensuche in der baumartigen Datenstruktur durchgeführt wird, bei der nacheinander für jede Baumebene sequentiell durch die Knoten der jeweiligen Baumebene der baumartigen Datenstruktur iteriert wird. Die Breitensuche unterscheidet sich dabei von der Tiefensuche in einer baumartigen Datenstruktur dadurch, dass bei der Breitensuche nacheinander die einzelnen Knoten einer Baumebene durchlaufen werden und erst dann, wenn durch sämtliche Knoten einer Baumebene iteriert wurde, die Knoten der darunter liegenden nächsten Baumebene durchlaufen werden. Dies bedeutet auch, dass bei einer Breitensuche immer zuerst

sämtliche Elternknoten durchlaufen werden, bevor die zu einem Elternknoten gehörenden Kinderknoten bzw. Blattknoten durchlaufen werden. Dies bedeutet in der Regel auch, dass zunächst alle Geschwisterknoten eines Elternknotens durchlaufen werden, bevor die Geschwisterknoten des nächsten übergeordneten Elternknotens einer Baumebene durchlaufen werden.

[0017] Während der Breitensuche in der baumartigen Datenstruktur wird dabei für jeden Knoten, der eine jeweilige Datei repräsentiert, der Dateityp der durch den Knoten repräsentierten Datei ermittelt. Anschließend wird eine den ermittelten Dateityp repräsentierende Typcodierung aus einer Liste von verschiedenen Codierungen ausgewählt, wobei die Typcodierung des Dateityps den jeweiligen Dateityp der dem Knoten zugrunde liegenden Datei charakterisiert. So enthält beispielsweise die Liste der Codierungen für Ordner, Dateien und Links jeweils entsprechend unterschiedliche Codierungen, wodurch für jeden Knoten und der damit repräsentierenden Datei der Dateityp aus der Typcodierung ermittelbar ist.

[0018] Die so ermittelte Typcodierung wird dann in ein Dateityp-Datenfeld der zu erstellenden sequentiellen Datenstruktur eingefügt, wobei die Typcodierungen der einzelnen durch die Knoten repräsentierten Dateien nacheinander gemäß der Breitensuche in das sequentielle Dateityp-Datenfeld eingefügt werden.

[0019] Basierend auf dem Prinzip der Breitensuche bedeutet dies, dass sämtliche Typcodierungen von Dateien einer gemeinsamen Baumebene in dem sequentiellen Dateityp-Datenfeld vor sämtlichen Typcodierungen jener Dateien stehen, die in den Baumebenen tiefer angeordnet sind. Dabei wird jede Typcodierung an einer bestimmten Position in dem Dateityp-Datenfeld abgelegt, wobei sich die Position aus der Breitensuche durch die baumartige Datenstruktur ergibt. Die Typcodierungen werden dabei immer an diejenige Position eingefügt, die die erste freie Position am Ende des Dateityp-Datenfeldes darstellt. Somit werden sämtliche Typcodierungen der einzelnen durch die Knoten repräsentierten Dateien gemäß der Breitensuche nacheinander in das sequentielle Dateityp-Datenfeld eingefügt.

[0020] Schließlich werden die Positionen derjenigen in dem sequentiellen Dateityp-Datenfeld enthaltenen Typcodierungen durch eine Anfangs- oder Endcodierung in der sequentiellen Datenstruktur gekennzeichnet, deren zugrunde liegenden Dateien in der baumartigen Datenstruktur jeweils ein Anfangs- oder Endknoten einer Reihe von Knoten entsprechen, die alle einen gemeinsamen Elternknoten haben. Oder anders formuliert, die Anfangs- oder Endcodierung kennzeichnet eine Reihe von Knoten, die alle zusammen Geschwisterknoten sind und einen gemeinsamen Elternknoten haben. Hierdurch wird die organisatorische Struktur der baumartigen Datenstruktur in der sequentiellen Datenstruktur abgebildet, wodurch die Zuordnung jeder Datei innerhalb der Baumstruktur möglich wird.

[0021] Das Erstellen der sequentiellen Datenstruktur wie vorstehend beschrieben hat dabei den entscheidenden Vorteil, dass die wesentlichen Aspekte der abgespeicherten Dateien, nämlich der Dateityp sowie die organisatorische Struktur der Dateien, sehr einfach, schnell und platzeffizient erstellt und abgebildet werden können, was insbesondere auf kleinen, mobilen Endgeräten besonders vorteilhaft ist.

[0022] Um nun die Dateisystemintegrität zu überprüfen, wird die erstellte sequentielle Datenstruktur mit einer zuvor nach dem gleichen Muster erstellte sequentielle Referenz-Datenstruktur verglichen, wobei in Abhängigkeit von dem Vergleich dann die Dateisystemintegrität festgestellt oder eben nicht festgestellt werden kann.

[0023] Hierbei ist es bspw. denkbar, dass aus der erstellten sequentiellen Datenstruktur ein Hashwert als äquivalente mathematische Repräsentation mittels eines bekannten Hashalgorithmus erstellt wird, wobei aus der zuvor erstellten sequentiellen Referenz-Datenstruktur ebenfalls ein Hashwert erstellt wird. Basierend auf einem Vergleich der Hashwerte kann dann die Dateisystemintegrität festgestellt werden. Sind beide Hashwerte gleich, so hat sich die aktuelle Dateistruktur der Datenverarbeitungseinrichtung gegenüber der Dateistruktur, auf dessen Basis die sequentielle Referenz-Datenstruktur erstellt wurde, nicht verändert.

[0024] Erfindungsgemäß wird die sequentielle Datenstruktur auf der Datenverarbeitungseinrichtung mit einigen oder allen Dateien der zu überprüfenden Dateiverwaltung erstellt, wobei der Vergleich der Datenstrukturen oder ihren mathematischen Repräsentationen (Hashwert) und die Feststellung der Dateisystemintegrität in Abhängigkeit von dem Vergleich entweder auf der zu überprüfenden Datenverarbeitungseinrichtung durchgeführt wird oder auf einer mit der zu überprüfenden Datenverarbeitungseinrichtung elektronisch verbundenen zentralen Datenverarbeitungsanlage.

[0025] Im ersten Fall wird zunächst die sequentielle Datenstruktur auf der zu überprüfenden Datenverarbeitungseinrichtung erstellt, wobei die für den Vergleich der Datenstrukturen notwendige sequentielle Referenz-

Datenstruktur vor dem Vergleich auf die Datenverarbeitungseinrichtung übertragen wurde. Dies kann beispielsweise dadurch geschehen, dass bereits bei der Initialisierung der Datenverarbeitungseinrichtung die Referenz-Datenstruktur in einen nur lesbaren Speicherbereich fest eingeschrieben wird und während des Betriebes nicht veränderbar ist, so dass die Datenverarbeitungseinrichtung auf die nicht veränderbare sequentielle Referenz-Datenstruktur zum Zwecke des Vergleiches und Feststellung der Dateisystemintegrität zugreifen kann. Denkbar ist aber auch, dass bei Überprüfung der Dateisystemintegrität der Datenverarbeitungseinrichtung eine Anfrage an eine zentrale Datenverarbeitungsanlage, beispielsweise einem Zentralserver, gestellt wird, woraufhin dann die zentrale Datenverarbeitungsanlage die sequentielle Referenz-Datenstruktur oder eine mathematische Repräsentation hiervon an die Datenverarbeitungseinrichtung überträgt. Hierfür sendet die Datenverarbeitungseinrichtung beispielsweise eine eindeutige Geräteerkennung an den Zentralserver bzw. die zentrale Datenverarbeitungsanlage, woraufhin diese dann die zu der anfragenden Datenverarbeitungseinrichtung passende Referenz-Datenstruktur aus einem Speicher ausliest und an die Datenverarbeitungseinrichtung überträgt. Das Gerät kann somit selbstständig eine entsprechende Dateisystemintegritätsprüfung durchführen.

[0026] Im zweiten Fall wird die auf der zu überprüfenden Datenverarbeitungseinrichtung erstellte sequentielle Datenstruktur der Dateiverwaltung an eine zentrale Datenverarbeitungsanlage bzw. einen Zentralserver übersandt, woraufhin der Vergleich der übertragenden sequentiellen Datenstruktur der Datenverarbeitungseinrichtung mit der entsprechenden sequentiellen Referenz-Datenstruktur auf der zentralen Datenverarbeitungsanlage bzw. dem Zentralserver erfolgt. Hierdurch wird es möglich, gänzlich manipulationssicher die Dateisystemintegrität einer Datenverarbeitungseinrichtung zu überprüfen, sofern die zentrale Datenverarbeitungsanlage bzw. der Zentralserver nicht kompromittiert ist und die sequentielle Referenz-Datenstruktur ebenfalls manipuliert wurde.

[0027] Mit der vorliegenden Erfindung wird es somit möglich, die Dateisystemintegrität schnell und effizient überprüfen zu können, da der Platzbedarf der vorliegenden erfindungsgemäßen Datenstruktur äußerst gering ist und sich somit bspw. zur Übertragung an andere Geräte oder Zentralanlagen eignet. Zum Vergleich, die Übertragung der gesamten Dateistruktur der Dateiverwaltung ist in der Regel aufgrund der Größe und des Platzbedarfs aller abgespeicherten Dateien meist nicht möglich.

[0028] Vorteilhafterweise bestehen die Typcodierungen aus eindeutigen, alphanummerischen Zeichen, die jeweils den entsprechenden Dateityp codieren. So kann die Codierung der Dateitypen beispielsweise mithilfe von einstelligen Ziffern erfolgen, die jeweils einen entsprechenden Dateityp eindeutig repräsentieren. Hierdurch kann für jede Datei der Dateityp mithilfe eines einfachen 8-Bit-Zeichens codiert werden, wodurch die gesamte Dateisystemstruktur besonders platzsparend abgespeichert werden kann.

[0029] In einer weiteren vorteilhaften Ausführungsform werden die Anfangs- und Endcodierungen der jeweiligen Anfangs- und Endknoten in einen Geschwisterknoten-Datenfeld der sequentiellen Datenstruktur derart eingefügt, dass die Position einer Anfangs- oder Endcodierung eines Anfangs- oder Endknotens in dem Geschwisterknoten-Datenfeld derjenigen Position in dem Datentyp-Datenfeld entspricht, an der die Typcodierung des Dateityps der dem Anfangs- oder Endknoten zugrundeliegenden Datei eingefügt ist.

[0030] Damit referenziert die Position in dem Geschwisterknoten-Datenfeld auch gleichzeitig die semantisch korrekte Position in dem Dateityp-Datenfeld, so dass sich die entsprechenden Reihen von Geschwisterknoten schnell und effizient zuordnen lassen.

[0031] In einer vorteilhaften Ausführungsform wird für jeden Knoten zusätzlich zu dem Dateityp mindestens eine weitere Dateieigenschaft der durch den jeweiligen Knoten repräsentierten Datei ermittelt und eine den ermittelten Dateityp und Dateieigenschaft repräsentierende Typcodierung aus einer Liste von verschiedenen Codierungen ausgewählt, so dass anhand der Typcodierung nicht nur auf den Dateityp, sondern gleichzeitig auch auf eine bestimmte Dateieigenschaft geschlossen werden kann. So ist es beispielsweise denkbar, dass bei alphanummerischen Zeichen als Typcodierung eine bestimmte Anzahl von alphanummerischen Zeichen ein und denselben Dateityp repräsentieren, aber jede den Dateityp repräsentierende Typcodierung jeweils eine bestimmte Dateieigenschaft charakterisieren, wodurch nicht nur der Dateityp sondern auch die Dateieigenschaft unterscheidbar wird.

[0032] In einer weiteren vorteilhaften Ausführungsform wird überprüft, ob der ermittelte Dateityp der durch den jeweiligen Knoten repräsentierten Datei einen Verweis bzw. einen Link auf eine andere, in dem digitalen Speichermedium abgespeicherte Datei ist. Wurde festgestellt, dass es sich bei dem Dateityp der durch den jeweiligen Knoten repräsentierten Datei um einen Verweisdateityp handelt, so wird in dem Dateityp-Datenfeld

die Position der Typcodierung derjenigen zugrundeliegenden Datei, auf welche die Datei vom Dateityp verweist bzw. Link verweist, ermittelt und die ermittelte Position dann in ein Verweisziel-Datenfeld der sequentiellen Datenstruktur eingefügt wird. Hierdurch wird es möglich, einen direkten Zugriff auf sämtliche Dateien zu erhalten, auf die mithilfe einer Verweisdatei bzw. eines Links verwiesen wird.

[0033] Vorteilhaft ist hierbei, wenn die Positionen nacheinander in der Reihenfolge der Feststellung bei der Breitensuche in das Verweisziel-Datenfeld der sequentiellen Datenstruktur eingefügt werden, wodurch sich eine besonderes effiziente und platzsparende Abspeicherung der Positionen derjenigen Dateien in dem Dateityp-Datenfeld ergibt, auf die der entsprechende Link verweist. Denkbar ist aber auch, dass die Positionen in dem Verweisziel-Datenfeld an denjenigen Positionen eingefügt werden, die den einzufügenden Positionen entsprechen.

[0034] In einer weiteren vorteilhaften Ausführungsform wird für jeden Knoten zusätzlich zu dem Dateityp mindestens eine weiteren Dateieigenschaft der durch den jeweiligen Knoten repräsentierten Datei ermittelt und die ermittelte Dateieigenschaft der durch den jeweiligen Knoten repräsentierten Datei nacheinander in der Reihenfolge der Breitensuche in ein Eigenschaften-Datenfeld eingefügt. Hierdurch wird neben dem Dateityp und einer möglicherweise schon dort mit integrierten Codierung einer Dateieigenschaft es auch möglich, zusätzliche Dateieigenschaften abzuspeichern, was beispielsweise mithilfe von speziellen Dateieigenschaften-Codierungen oder im Klartext erfolgen kann.

[0035] Eine solche Dateieigenschaft kann beispielsweise der Dateiname sein, der in einem Dateinamen-Datenfeld als Eigenschaften-Datenfeld eingefügt wird. Dabei wird der Anfang oder das Ende eines zu einer Datei gehörenden Dateinamens in dem Dateinamen-Datenfeld durch eine Anfangs- oder Endcodierung in der Datenstruktur gekennzeichnet, um so immer korrekt den Dateinamen ermitteln zu können. Dies ist beispielsweise dann besonders wichtig und vorteilhaft, wenn das Dateinamen-Datenfeld ein zeichenbasiertes Datenfeld ist, bei dem an einer beliebigen Position immer nur ein beliebiges Zeichen stehen kann. Ein Dateiname, der aus mehreren Zeichen besteht, muss demnach über mehrere Positionen bzw. Felder des Dateinamen-Datenfeldes abgelegt werden, wobei es dann wichtig ist zu wissen, an welcher Position innerhalb des Dateinamen-Datenfeldes das Anfangszeichen oder das Endzeichen eines zu einer Datei gehörenden Dateinamens steht.

[0036] In einer vorteilhaften Ausführungsform hierzu wird die Anfangs- oder Endcodierung der jeweiligen Dateinamen in ein Trenn-Datenfeld derart eingefügt, dass die Position einer Anfangs- oder Endcodierung eines Dateinamens der Anfangs- oder Endposition des Dateinamens in dem zeichenbasierten Dateinamen-Datenfeld entspricht. Die Position einer Anfangs- oder Endcodierung in dem Trenn-Datenfeld entspricht dabei derjenigen Position in dem Dateinamen-Datenfeld, an dem der Enddateiname beginnt oder endet.

[0037] In einer vorteilhaften Ausführungsform werden vor dem Erstellen der sequentiellen Datenstruktur auf der zu überprüfenden Datenverarbeitungseinrichtung ein Teil der Dateien, die in der Dateiverwaltung zwar enthalten, aber noch nicht auf der Dateiverarbeitungseinrichtung abgespeichert sind, von einer mit der zu überprüfenden Datenverarbeitungseinrichtung elektronisch verbundenen zentralen Datenverarbeitungsanlage auf die zu überprüfende Datenverarbeitungseinrichtung übertragen werden. Mit anderen Worten, ein Teil der Dateien wurde auf einem externen Speichermedium ausgelagert und wird bei Bedarf auf die Datenverarbeitungseinrichtung übertragen. In der Dateiverwaltung sind diese, noch nicht abgespeicherten Dateien, vorteilhafterweise bereits enthalten, wobei lediglich deren Speicherort von dem Speicherort der übrigen Dateien verschieden ist.

[0038] Die sequentielle Datenstruktur wird nun auf der zu überprüfenden Datenverarbeitungseinrichtung in Abhängigkeit von den übertragenen Dateien und den bereits zuvor auf der Datenverarbeitungseinrichtung abgespeicherten Dateien erstellt.

[0039] Die Aufgabe wird im Übrigen auch mit einem Computerprogramm mit Programmcodemitteln gemäß Anspruch 11 gelöst, das insbesondere auf einem maschinenlesbaren Träger gespeichert ist, wobei das Computerprogramm mit den Programmcodemitteln eingerichtet ist zur Durchführung des vorstehenden Verfahrens, wenn das Computerprogramm auf einer Datenverarbeitungseinrichtung oder Datenverarbeitungsanlage abläuft.

[0040] Die Aufgabe wird im Übrigen auch mit einer Datenverarbeitungseinrichtung in einer elektronischen Datenverarbeitungseinheit, einem digitalen Speichermedium, auf dem eine Mehrzahl von Dateien abgespeichert oder abspeicherbar sind, und mit einer digitalen Dateiverwaltung, die eine Mehrzahl von Dateien für die

Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung enthält, gelöst, wobei die Datenverarbeitungseinrichtung mittels der elektronischen Datenverarbeitungseinheit zum Durchführen des vorstehend genannten Verfahrens ausgebildet ist, um die Dateisystemintegrität der Dateiverwaltung zu überprüfen.

[0041] Unter einer Datenverarbeitungseinrichtung werden insbesondere ein Computer, mobiler Computer, Rechner bzw. Rechenmaschinen verstanden. Eine Datenverarbeitungseinrichtung kann insbesondere ein mobiles Messgerät, wie bspw. ein Verkehrsmessgerät, eine Waage oder der Gleichen sein. Unter „elektronisch ausführbar“ wird insbesondere verstanden, dass die Schritte mittels einer elektronischen Datenverarbeitungseinheit, bspw. eine mikroprozessorgesteuerte bzw. mikrocontrollergesteuerte Datenverarbeitungseinheit, automatisch ausgeführt werden, wobei die elektronische Datenverarbeitungseinheit in der Regel frei programmierbar oder fest verdrahtet ist.

[0042] Die Erfindung wird anhand der beigefügten Figuren beispielhaft erläutert. Es zeigen:

Fig. 1 - schematische Darstellung einer Datenverarbeitungseinrichtung;

Fig. 2 - baumartige Datenstruktur von auf einer mit digitalem Speichermedium abgespeicherten Datei;

Fig. 3 - erfindungsgemäße Datenstruktur.

[0043] **Fig. 1** zeigt schematisch die Struktur einer Datenverarbeitungseinrichtung 1, die eine elektronische Datenverarbeitungseinheit 2 und einen damit signaltechnisch verbundenen digitalen Speicher bzw. Speichermedium 3, beispielsweise eine Festplatte, hat. Die elektronische Datenverarbeitungseinheit 2 ist dabei zum Ausführen von Computerprogrammen ausgebildet und dabei unter anderem so eingerichtet, dass sie den Zugriff auf Dateien des Speichermediums 3 realisieren kann. Das Speichermedium 3 weist dabei insbesondere eine Dateiverwaltung 4 auf, die dazu geeignet und eingerichtet ist, die entsprechenden Daten und Informationen in dem Speichermedium 3 in Dateien logisch zu strukturieren und ihre jeweiligen Adressen und Zugriffsmöglichkeiten abzuspeichern. Dadurch wird es möglich, mithilfe eines Dateinamens die zu dem jeweiligen Dateinamen betreffenden Daten und Informationen aus dem Speichermedium 3 herauszulesen bzw. Daten in Dateien einzuschreiben oder diese zu verändern.

[0044] Die elektronische Datenverarbeitungseinrichtung 1 weist des Weiteren eine Kommunikationseinheit 5 auf, um insbesondere einen drahtlosen, aber auch drahtgebundenen, Datenaustausch mit anderen Anlagen zu ermöglichen.

[0045] Die Datenverarbeitungseinheit 2 ist dabei ausgebildet, eine sequentielle Datenstruktur 6, gemäß den später noch gezeigten Vorschriften, in Abhängigkeit von dem Inhalt der Dateiverwaltung 4 zu erstellen und ggf. in dem digitalen Speichermedium 3 zwischen zu speichern.

[0046] Mithilfe der Kommunikationseinheit 5 kann die Datenverarbeitungseinrichtung 1 mit einer zentralen Datenverarbeitungsanlage 7 kommunizieren, die im Folgenden als Server bezeichnet wird. Auf dem Server befindet sich eine Reihe von sequentiellen Referenz-Datenstrukturen 8 für verschiedene Datenverarbeitungseinrichtungen 1, wobei durch eine entsprechende Anforderung, die von einer Datenverarbeitungseinrichtung 1 an den Server 7 gesendet wurde, der Server in der Lage ist, die entsprechende Referenz-Datenstruktur 8 zu ermitteln und an die anfragende Datenverarbeitungseinrichtung 1 zu übertragen. Nach der Übertragung liegt der anfragenden Datenverarbeitungseinrichtung 1 somit auch die sequentielle Referenz-Datenstruktur 8 vor, so dass durch einen Vergleich der sequentiellen Referenz-Datenstruktur 8 mit der zuvor erstellten sequentiellen Datenstruktur 6 festgestellt werden kann, ob die Dateisystemintegrität noch gegeben ist oder der Inhalt der Dateiverwaltung 4 gegenüber der Referenz verändert wurde.

[0047] Im Ausführungsbeispiel der **Fig. 1** wird davon ausgegangen, dass sich sämtliche Dateien der Dateiverwaltung 4 auf dem Speichermedium 3 befinden. Wie bereits zuvor beschrieben, ist es denkbar, dass ein Teil der Dateien zunächst auf die Datenverarbeitungseinrichtung 1 geladen werden, wenn diese auf einem externen Speichermedium ausgelagert wurden.

[0048] Selbstverständlich ist es auch denkbar, dass der Vergleich der sequentiellen Datenstruktur 6 mit der sequentiellen Referenz-Datenstruktur 8 auf den Server 7 erfolgt, wobei hierfür die Datenverarbeitungseinrichtung 1 die erstellte sequentielle Datenstruktur 6 an den Server 7 mittels der Kommunikationseinrichtung 5 überträgt.






[0049] Anhand der **Fig. 2** und **Fig. 3** soll dabei das erfindungsgemäße Verfahren zum Erstellen der sequentiellen Datenstruktur an einem Beispiel näher erläutert werden. **Fig. 2** zeigt dabei beispielhaft eine baumartige

Datenstruktur von insgesamt 18 Dateien, die mithilfe des erfindungsgemäßen Verfahrens in die sequentielle Datenstruktur gemäß **Fig. 3** gebracht werden sollen. Bei der baumartigen Datenstruktur gemäß **Fig. 2** werden die einzelnen Dateien, die auf dem Datenspeicher abgespeichert sind, in Form von Knoten einer baumartigen Datenstruktur repräsentiert. Dabei sind die einzelnen Knoten durchnummeriert, um ein besseres Verständnis der baumartigen Datenstruktur zu erhalten. Der jeweilige Dateityp wird durch ein entsprechendes Knotensymbol charakterisiert.

[0050] Der oberste Knoten 1 kennzeichnet dabei den Wurzel- bzw. Rootknoten und ist in manchen Systemen sichtbar oder auch ausgeblendet. An diesen Wurzelknoten 1 schließend sich dabei die Kindsknoten 2, 3, 4 und 5 an, die allesamt eine gemeinsame Reihe von Geschwisterknoten bilden und einen gemeinsamen Elternknoten, den Wurzelknoten 1, haben.

[0051] Die Nummerierung der Knoten stellt dabei gleichzeitig auch die Suchreihenfolge der Breitensuche durch die raumartige Datenstruktur dar.

[0052] Im Ausführungsbeispiel der **Fig. 2** wurden dabei die folgenden Dateitypen durch ihre jeweiligen Knotensymbole verwendet:

<u>Symbol</u>	<u>Dateityp</u>
	Ordner bzw. Verzeichnis
	normale Datei (Datendatei, Programmdatei)
	Verweis bzw. Link
	blockorientiertes Gerät
	zeichenorientiertes Gerät

[0053] Die einzelnen Dateitypen werden dabei mithilfe von alphanumerischen Zeichen codiert, wobei ein Verzeichnis bzw. Ordner das Zeichen 1 erhält, sofern er Dateien enthält, sonst wird ein Verzeichnis bzw. Ordner mit 0 codiert. Eine normale Datei (Datendatei, Programmdatei) wird mit einer 2 codiert, ein Link bzw. Verweis mit einer 3, ein blockorientiertes Gerät mit einer 4 und ein zeichenorientiertes Gerät mit einer 5.

[0054] Die Dateien 10, 11 und 12 stellen dabei Verweise bzw. Links auf andere Dateien dar, wobei der Link 10 auf die Datei 6, der Link 11 auf die Datei 7 und der Link 12 auf die Datei 8 verweisen.

[0055] Diese baumartige Datenstruktur, wie sie in **Fig. 2** dargestellt ist, wird von der Dateiverwaltung bereitgestellt, um nun die entsprechende sequentielle Datenstruktur aufbauen zu können. Diese baumartige Datenstruktur kann beispielsweise derart bereitgestellt werden, dass entsprechende Funktionen durch die Dateiverwaltung bereitgestellt werden, die es ermöglichen, auf die einzelnen Dateien bzw. Knoten in einer vorgegebenen Reihenfolge zugreifen zu können.

[0056] Gemäß der Breitensuche wird nun sequentiell durch die einzelnen Knoten durchgegangen, wobei hierfür die einzelnen Bauebenen nacheinander durchlaufen werden. Die erste Bauebene bildet dabei der Wurzelknoten 1, dessen zugrundeliegende Datei einen Ordner darstellt. Da der Wurzelknoten 1 mehrere Kinderknoten 2 bis 5 hat, erhält der Wurzelknoten 1 die Dateitypcodierung „1“, die sodann in das Dateityp-Datenfeld S (s. **Fig. 3**) an der letzten Position eingefügt wird. Da zuvor in das Dateityp-Datenfeld noch keine Typco-

dierungen eingefügt wurden, wird die Typcodierung „1“ des Wurzelknotens 1 an die erste Position des Dateityp-Datenfeldes eingefügt (in der Regel gekennzeichnet durch den Index $I = 0$).

[0057] Da die oberste Baumebene (nullte Ebene) keine weiteren Knoten enthält, da der Wurzelknoten 1 in der Regel alleine steht, wird nun mit der sequentiellen Suche in der darauffolgenden Baumebene 1 fortgeführt. Die Ebene 1 besteht aus den Geschwisterknoten 2, 3, 4 und 5. Begonnen wird dabei mit dem linken Knoten, in Richtung des äußersten rechten Knotens, was letztendlich die Richtung der Breitensuche in den einzelnen Ebenen entspricht.

[0058] Die durch den Knoten 2 repräsentierte Datei ist vom Dateitypverzeichnis und ist nicht leer (enthält Kinderknoten), so dass hierfür die Typcodierung „1“ gewählt wird. Diese wird dann ebenfalls in das Dateityp-Datenfeld S an der letzten Position (Index 1) eingefügt. Der nächste Knoten ist eine normale Daten- oder Programmdatei, durch die Typcodierung 2 gekennzeichnet ist und ebenfalls in das Dateityp-Datenfeld eingefügt wird.

[0059] Zu erkennen ist, dass der letzte Knoten 5 der Ebene 1 vom Dateitypverzeichnis ist und keine weiteren Dateien bzw. Knoten enthält, so dass er anstelle der Typcodierung 1 für Verzeichnis die Typcodierung 0 für Verzeichnis, leer, zugewiesen bekommt (s. Index 4).

[0060] Nachdem die Knoten der Ebene 1 vollständig durchsucht wurden, wird mit der Ebene 2 fortgesetzt, wobei hier entsprechend der die jeweiligen Knoten zugrundeliegenden Dateien der Dateityp ermittelt und die entsprechende Typcodierung ausgewählt und in das Dateityp-Datenfeld eingefügt wird. Das Einfügen geschieht dabei immer an der letzten Position, so dass der Index des Dateityp-Datenfeldes die jeweilige Position des Knotens repräsentiert (Knotennummer -1).

[0061] Parallel dazu wird ein Geschwisterknoten-Datenfeld Bs aufgebaut, was die entsprechenden Informationen bezüglich einer Reihe von gemeinsamen Geschwisterknoten enthält. Hierbei wird im Ausführungsbeispiel der **Fig. 3** für das Geschwisterknoten-Datenfeld eine Anfangscodierung verwendet, mit der der Beginn bzw. der Anfang einer Reihe von Geschwisterknoten gekennzeichnet wird, die alle einen gemeinsamen Elternknoten haben. Als Codierung wird hierfür das Zeichen „1“ verwendet, das die Anfangscodierung darstellt. Das Zeichen „0“ ist dabei als Platzhalter zu verstehen und meint, dass an dieser Position keine Anfangs- oder ggf. Endcodierung vorhanden ist.

[0062] So wird an der Position 1 eine Anfangscodierung „1“ in das Geschwisterknoten-Datenfeld eingefügt, da an der Position 1 (Index = 1) in dem Dateityp-Datenfeld ein Dateityp hinterlegt ist, dessen mit diesem Knoten korrespondierende Datei 2 den Anfang einer Reihe von Geschwisterknoten 2 bis 5 darstellt. Daher wird für diesen Knoten und seine Typcodierung entsprechend gekennzeichnet, dass es sich hierbei um den Anfang einer Reihe von Geschwisterknoten handelt bzw. dass es sich um den ersten Knoten einer Reihe von Geschwisterknoten handelt.

[0063] Die gleiche Anfangscodierung wird im Übrigen auch für den Knoten 6 (Index 5), den Knoten 10 (Index 9), den Knoten 13 (Index 12) sowie den Knoten 15 (Index 14) und den Knoten 17 (Index 16) verwendet.

[0064] Damit wurde eine Datenstruktur D aufgebaut, die für jeden Knoten einer Breitensuche der baumartigen Datenstruktur den Dateityp codiert und den Beginn einer Geschwisterknoten-Reihe kennzeichnet, wodurch sehr effizient und platzsparend eine Dateisystemstruktur aufgebaut werden kann.

[0065] Eine so erstellte Dateisystemstruktur kann dann mithilfe der Datenstruktur beispielsweise bei Inbetriebnahme bzw. Erstinbetriebnahme eines Gerätes in dem privaten Bereich des Gerätes übertragen werden, so dass jederzeit die Systemintegrität mithilfe geeigneter Prüfsoftware oder durch das Übertragen von Hash-Werten bezüglich der Datenstruktur an entfernte Geräte überprüft werden kann.

[0066] Im Ausführungsbeispiel der **Fig. 2** und **Fig. 3** wird des Weiteren ein weiteres Datenfeld während der Breitensuche oder auch anschließend, nach dem Erstellen der Datenfelder S und Bs, befüllt, das entsprechende Informationen zu Linkzielen von Links bzw. Verweisen enthält. Hierfür wird anhand des Dateityps überprüft, ob der Dateityp vom Typverweis bzw. Link ist, was bedeutet, dass die Datei vom Typverweis bzw. Link auf eine andere Datei innerhalb der baumartigen Datenstruktur verweist.

[0067] Wurde ein derartiger Knoten aufgefunden, der vom Dateitypverweis bzw. Link ist, so wird der entsprechende Knoten ermittelt, auf den der Verweis bzw. Link entsprechend verweist (Zieldatei bzw. Zielknoten).

Anschließend wird diejenige Position bzw. Indexnummer in dem Dateityp-Datenfeld in ein Verweis-Datenfeld H eingefügt, an deren Stelle eine Typcodierung hinterlegt ist, die der Zieldatei bzw. Zielknoten entspricht.

[0068] Im Ausführungsbeispiel der **Fig. 2** und **Fig. 3** verweist der Knoten 10 auf den Knoten 6, so dass zunächst an der Indexposition 9 die Typcodierung „3“ hinterlegt ist. Da diese Datei bzw. dieser Knoten, der durch die Position bzw. den Index 9 gekennzeichnet ist, auf die Zieldatei bzw. den Zielknoten 6 (Position bzw. Index 5) verweist, wird diese Position bzw. dieser Index „5“ dann ebenfalls in das Verweisziel-Datenfeld H der Reihe nach eingefügt, so dass beim Auffinden einer ersten Typcodierung vom Typverweis bzw. Link festgestellt werden kann, dass diese Datei auf den Knoten mit der Position bzw. Index 5 verweist. Äquivalent hierzu kann auch eine entsprechende Knotenreferenz hinterlegt werden, die den Knoten eindeutig referenziert, wie dies beispielsweise durch die Nummerierung im Ausführungsbeispiel der **Fig. 2** erfolgen kann. In diesem Fall würde die Zahl „6“ hinterlegt werden, da diese auf den Knoten mit der Nummer 6 (Position bzw. Index 5) verweist.

[0069] Des Weiteren werden die Dateinamen in ein spezielles Dateinamen-Datenfeld eingefügt, wobei das Dateinamen-Datenfeld dergestalt ist, dass an jeder Position genau ein alphanummerisches Zeichen stehen kann. Ein Dateiname bestehend aus mehreren alphanummerischen Zeichen besetzt somit in dem Dateinamen-Datenfeld mehrere Positionen hintereinander. Um Beginn und Ende eines solchen in einem Dateinamen-Datenfeld hinterlegten Dateinamen entsprechend erkennen zu können, wird mithilfe eines Trenn-Datenfeldes B_N der Beginn bzw. das Ende eines Dateinamens in Bezug auf das Dateinamen-Datenfeld abgespeichert. Im Ausführungsbeispiel der **Fig. 3** wird hierfür in das Trenn-Datenfeld die Position bzw. derjenige Index durch Einfügen einer Anfangscodierung „1“ markiert, deren Position bzw. Index im Dateinamen-Datenfeld das Anfangszeichen eines Dateinamens enthält. Die im Ausführungsbeispiel der **Fig. 2** gezeigten Knoten repräsentieren dabei jeweils eine Datei mit den folgenden Dateinamen:

<u>Knotennummer</u>	<u>Dateiname</u>
1	root
2	usr
3	kernel
4	desktop
5	mnt
6	game
7	browser
8	gcc
9	temp
10	play
11	browser
12	compiler
13	blk
14	char
15	blk1
16	blk2
17	char1
18	char2

[0070] Gemäß dessen wird so das Dateinamen-Datenfeld N sowie das Trenn-Datenfeld B_N befüllt, sowie in **Fig. 3** gezeigt ist.

[0071] Mithilfe der Datenfelder S und Bs kann man zu jedem Knoten traversieren. Die Funktionen „parent“ und „child“ sind dabei wie folgt:

- $\text{child}(n, i) = \text{select_1}(B; \text{rank_1}(S; n)) + i - 1$, falls $S[n] = 1$

- $\text{patent}(n) = \text{select_1}(S; \text{rank_1}(B; n))$

[0072] Über H kann auch noch abgefragt werden, ob ein Knoten ein Link ist ($S[n] = 3$), bzw. weitere Links besitzt und welche Knotennummer bzw. Indexposition diese haben:

- $\text{get_link}(n, i) = \text{select_3}(S; \text{select_n}(H; i))$
- $\text{get_orig}(n) = H[\text{rank_3}(S; n)]$, falls $S[n] = 3$

[0073] Die Funktionen „select“ sowie „rank“ definieren sich dabei für einen String S der Länge $|S| = n$ Folgendes:

- $\text{rank_a}(S, p)$: Mit $p \leq n$, gibt sie Anzahl von Zeichen a bis Position p in S aus.
- $\text{select_a}(S, n)$: Gibt die Position des n-ten Zeichens a in S aus.

[0074] Zusätzlich können über die Funktionen „rank“ und „select“ folgende Operationen direkt ausgeführt werden:

- Bestimmung der Anzahl von Knoten eines Ordners und deren Auflistung
- Bestimmung der Anzahl bestimmter Dateitypen in einem Ordner und deren Auflistung
- Bestimmung der Anzahl bestimmter Dateitypen im gesamten Dateisystem und deren Auflistung

[0075] Mit einer solchen Datenstruktur wird es somit möglich, die Dateisystemstruktur so zu indizieren, dass beispielsweise basierend auf den Linkzielen bzw. Vennreiszielen schnell und effizient die entsprechenden Verweisdateien, die auf die entsprechenden Verweisziele verweisen, aufgefunden werden können. Außerdem lässt sich eine derartige sequentielle Datenstruktur besonders platzsparend abspeichern, wodurch eine Übertragung der Dateisystemstruktur an andere Geräte möglich wird, um beispielsweise die Dateisystemintegrität zu überprüfen. Hierfür ist es beispielsweise denkbar, dass nach Aufbau der Datenstruktur und der erstellten Dateisystemstruktur ein gesamter Hash-Wert über alle in den Datenfeldern enthaltenen Daten gebildet wird, der dann übertragen wird, um ihn dann mit bereits vorherigen gebildeten Hash-Werten zu vergleichen. Denkbar ist aber auch, dass lediglich die Dateinamen signiert übermittelt werden, die sich aus dem Dateinamen-Datenfeld und dem Trenn-Datenfeld ergeben.

[0076] Es ist auch denkbar, dass für jede Datei ein Hash-Wert berechnet wird, der dann in ein Hash-Datenfeld eingetragen wird und entsprechend übermittelt wird. Hierbei werden sämtliche in der Datenstruktur zu einem Knoten bzw. einer Datei bekannten Informationen herangezogen. Denkbar ist aber auch, dass die Dateinamen ausgelassen werden, um entsprechend Platz bei der Übertragung zu sparen.

[0077] Alternativ ist auch denkbar, dass nur eine kleine Anzahl von vordefinierten Dateien betrachtet wird, von denen dann basierend auf der erstellten Datenstruktur und Dateisystemstruktur der Hash-Wert gebildet wird, um noch mehr Platz zu sparen. Hierbei ist es denkbar, dass einer Hash-Wert-Ermittlungseinheit zuvor eine Liste von Dateinummern übergeben wird von denen dann der entsprechende Hash-Wert basierend auf der Dateisystemstruktur gebildet werden soll.

[0078] Als Hash-Algorithmen kommen dabei beispielsweise einfache Checksummen wie CRC-16 bis hin zu sicheren Hash-Algorithmen wie SHA-2 in Frage.

[0079] Denkbar ist aber auch, dass die Dateien zum Betrieb des Dateiverarbeitungsgerätes erst bei Inbetriebnahme von einem entfernten Datenspeicher geladen werden. Um hier die Dateisystemintegrität der geladenen Dateien bzw. der geladenen Dateisysteme zu gewährleisten ist es denkbar, dass auf jedem Gerät die vorliegende Dateisystemstruktur in einer Datenstruktur oder auch die Hash-Werte der korrekten Dateisystemstruktur abgespeichert werden. Wird das Gerät nun in Betrieb genommen, werden die Dateien des Dateisystems auf das Endgerät geladen. Anschließend wird mithilfe des vorgestellten Verfahrens die Dateisystemstruktur und Datenstruktur erstellt und anschließend ein entsprechender Hash-Wert gebildet, der dann mit dem bereits auf dem Gerät hinterlegten Hash-Wert verglichen wird. Stimmen beide überein, konnte die Integrität positiv überprüft werden.

Patentansprüche

1. Verfahren zum Überprüfen der Dateisystemintegrität einer Dateiverwaltung einer Datenverarbeitungseinrichtung, wobei die Dateiverwaltung der Datenverarbeitungseinrichtung eine Mehrzahl von Dateien für

die Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung enthält, **gekennzeichnet durch** die elektronisch ausführbaren Schritte:

- Bereitstellen einiger oder aller in der Dateiverwaltung enthaltenen Dateien in einer baumartigen Datenstruktur, bei der jeder Knoten der baumartigen Datenstruktur eine Datei der Dateiverwaltung repräsentiert,
- Erstellen einer sequentiellen Datenstruktur aus der baumartigen Datenstruktur, indem
 - eine Breitensuche in der baumartigen Datenstruktur durchgeführt wird, bei der nacheinander für jede Baumebene sequentiell durch die Knoten der jeweiligen Baumebene der baumartigen Datenstruktur iteriert wird,
 - wobei für jedena Knoten ein Dateityp der durch den jeweiligen Knoten repräsentierten Datei ermittelt und eine den ermittelten Dateityp repräsentierende Typcodierung aus einer Liste von verschiedenen Codierungen ausgewählt wird, und
 - wobei die ermittelten Typcodierungen der einzelnen durch die Knoten repräsentierten Dateien nacheinander in ein sequentielles Dateityp-Datenfeld (S) der sequentiellen Datenstruktur eingefügt werden, wobei die Positionen (i) derjenigen in dem sequentiellen Dateityp-Datenfeld (S) enthaltenen Typcodierungen durch eine Anfangs- oder Endcodierung in der sequentiellen Datenstruktur gekennzeichnet werden, deren zugrundeliegenden Dateien in der baumartigen Datenstruktur jeweils einem Anfangs- oder Endknoten einer Reihe von Knoten entsprechen, die alle einen gemeinsamen Elternknoten haben,
- Vergleichen der erstellten sequentiellen Datenstruktur oder einer mathematischen Repräsentation hiervon mit einer zuvor nach dem gleichen Muster erstellten sequentiellen Referenz-Datenstruktur oder einer mathematischen Repräsentation hiervon, und
- Feststellen der Dateisystemintegrität in Abhängigkeit von dem Vergleich, wobei die sequentielle Datenstruktur auf der die zu überprüfende Dateiverwaltung aufweisende Datenverarbeitungseinrichtung mit einigen oder allen Dateien der Dateiverwaltung erstellt wird, wobei der Vergleich der Datenstrukturen oder ihrer mathematischen Repräsentationen und die Feststellung der Dateisystemintegrität in Abhängigkeit von dem Vergleich entweder
- auf der zu überprüfenden Datenverarbeitungseinrichtung durchgeführt wird, wobei hierfür die sequentielle Referenz-Datenstruktur zuvor auf die Datenverarbeitungseinrichtung übertragen wurde, oder
- auf einer mit der zu überprüfenden Datenverarbeitungseinrichtung elektronisch verbundenen zentralen Datenverarbeitungsanlage durchgeführt wird, wobei hierfür die erstellte sequentielle Datenstruktur zuvor auf die zentrale Datenverarbeitungsanlage übertragen wurde.

2. Verfahren nach Anspruch 1, **dadurch gekennzeichnet**, dass bei der Erstellung der sequentiellen Datenstruktur für jeden Dateityp ein eindeutiges alphanumerisches Zeichen, welches den jeweiligen Dateityp repräsentiert, als Typcodierung ausgewählt wird.

3. Verfahren nach Anspruch 1 oder 2, **dadurch gekennzeichnet**, dass bei der Erstellung der sequentiellen Datenstruktur die Anfangs- oder Endcodierungen der jeweiligen Anfangs- oder Endknoten in ein Geschwisterknoten-Datenfeld (Bs) der sequentiellen Datenstruktur derart eingefügt werden, dass die Position (j) einer Anfangs- oder Endcodierung eines Anfangs- oder Endknotens in dem Geschwisterknoten-Datenfeld (Bs) derjenigen Position (i) in dem Dateityp-Datenfeld (S) entspricht, an der die Typcodierung des Dateityps der dem Anfangs- oder Endknoten zugrundeliegenden Datei eingefügt ist.

4. Verfahren nach einem der vorhergehenden Ansprüche, **dadurch gekennzeichnet**, dass bei der Erstellung der sequentiellen Datenstruktur für jeden Knoten zusätzlich zu dem Dateityp mindestens eine weitere Dateieigenschaft der durch den jeweiligen Knoten repräsentierten Datei ermittelt und eine den ermittelten Dateityp und Dateieigenschaft repräsentierende Typcodierung aus einer Liste von verschiedenen Codierungen ausgewählt und in das Dateityp-Datenfeld eingefügt wird.

5. Verfahren nach einem der vorhergehenden Ansprüche, **dadurch gekennzeichnet**, dass bei der Erstellung der sequentiellen Datenstruktur überprüft wird, ob der ermittelte Dateityp der durch den jeweiligen Knoten repräsentierten Datei ein Verweis auf eine andere, in der Dateiverwaltung enthaltenen Datei ist, wobei bei Feststellung eines Verweisdateityps dann in dem Dateityp-Datenfeld die Position der Typcodierung derjenigen zugrundeliegenden Datei, auf welche die Datei vom Verweisdateityp verweist, ermittelt und die ermittelte Position in ein Verweisziel-Datenfeld (H) der sequentiellen Datenstruktur eingefügt wird.

6. Verfahren nach Anspruch 5, **dadurch gekennzeichnet**, dass die Positionen (i) nacheinander in der Reihenfolge der Feststellung bei der Breitensuche in das Verweisziel-Datenfeld (H) der sequentiellen Datenstruktur eingefügt werden oder dass die Positionen in dem Verweisziel-Datenfeld (H) an denjenigen Positionen eingefügt werden, die den einzufügenden Positionen entsprechen.

7. Verfahren nach einem der vorhergehenden Ansprüche, **dadurch gekennzeichnet**, dass bei der Erstellung der sequentiellen Datenstruktur für jeden Knoten zusätzlich zu dem Dateityp mindestens eine weitere Dateieigenschaft der durch den jeweiligen Knoten repräsentierten Datei ermittelt und die ermittelten Dateieigenschaften der durch den jeweiligen Knoten repräsentierten Dateien nacheinander in der Reihenfolge der Breitensuche in ein Eigenschaften-Datenfeld (P) der sequentiellen Datenstruktur eingefügt werden.

8. Verfahren nach Anspruch 7, **dadurch gekennzeichnet**, dass als zusätzliche Dateieigenschaft der jeweilige Dateiname der durch den Knoten repräsentierten Datei ermittelt wird, wobei die Dateinamen nacheinander in der Reihenfolge der Breitensuche in ein zeichenbasiertes Dateinamen-Datenfeld (N) der sequentiellen Datenstruktur als Eigenschaften-Datenfeld (P) eingefügt werden, wobei der Anfang oder das Ende eines zu einer Datei gehörenden Dateinamens in dem Dateinamen-Datenfeld (N) durch eine Anfangs- oder Endcodierung in der sequentiellen Datenstruktur gekennzeichnet werden.

9. Verfahren nach Anspruch 8, **dadurch gekennzeichnet**, dass die Anfangs- oder Endcodierung der jeweiligen Dateinamen in ein Trenn-Datenfeld (B_N) derart eingefügt werden, dass die Position (j) einer Anfangs- oder Endcodierung eines Dateinamens der Anfangs- oder Endposition (i) des Dateinamens in dem zeichenbasierten Dateinamen-Datenfeld (N) entspricht.

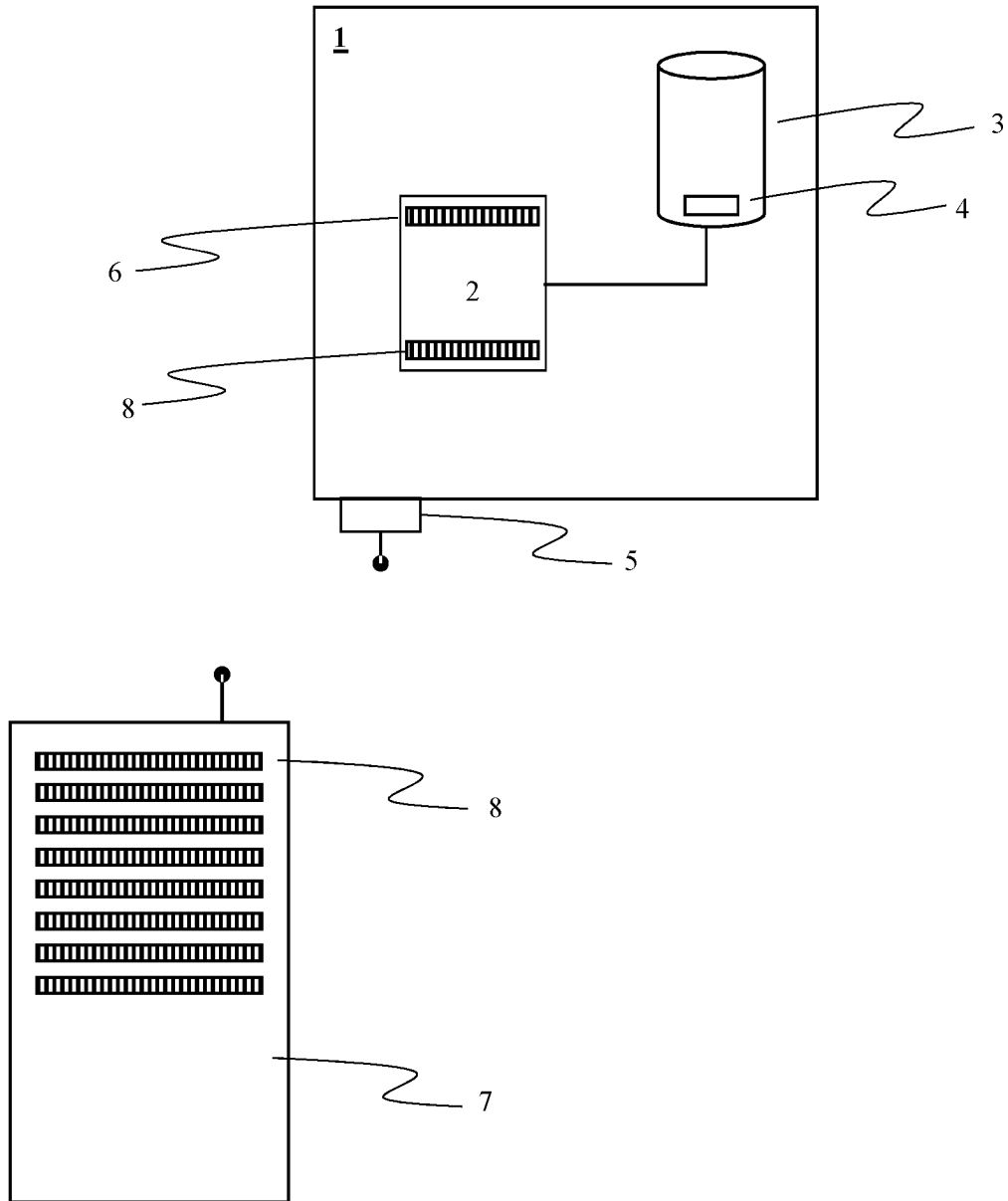
10. Verfahren nach einem der vorhergehenden Ansprüche, **dadurch gekennzeichnet**, dass vor dem Erstellen der sequentiellen Datenstruktur auf der zu überprüfenden Datenverarbeitungseinrichtung ein Teil der Dateien, die in der Dateiverwaltung enthalten und nicht auf der Datenverarbeitungseinrichtung abgespeichert sind, von einer mit der zu überprüfenden Datenverarbeitungseinrichtung elektronisch verbundenen zentralen Datenverarbeitungsanlage auf die zu überprüfende Datenverarbeitungseinrichtung übertragen werden, wobei die sequentielle Datenstruktur auf der zu überprüfenden Datenverarbeitungseinrichtung in Abhängigkeit von den vor der Übertragung bereits vorhandenen Dateien und den übertragenen Dateien erstellt wird.

11. Computerprogramm mit Programmcodemitteln, insbesondere gespeichert auf einem maschinenlesbaren Träger, eingerichtet zur Durchführung des Verfahrens nach einem der Ansprüche 1 bis 10, wenn das Computerprogramm auf einer elektronischen Datenverarbeitungseinrichtung und/oder Datenverarbeitungsanlage ausgeführt wird.

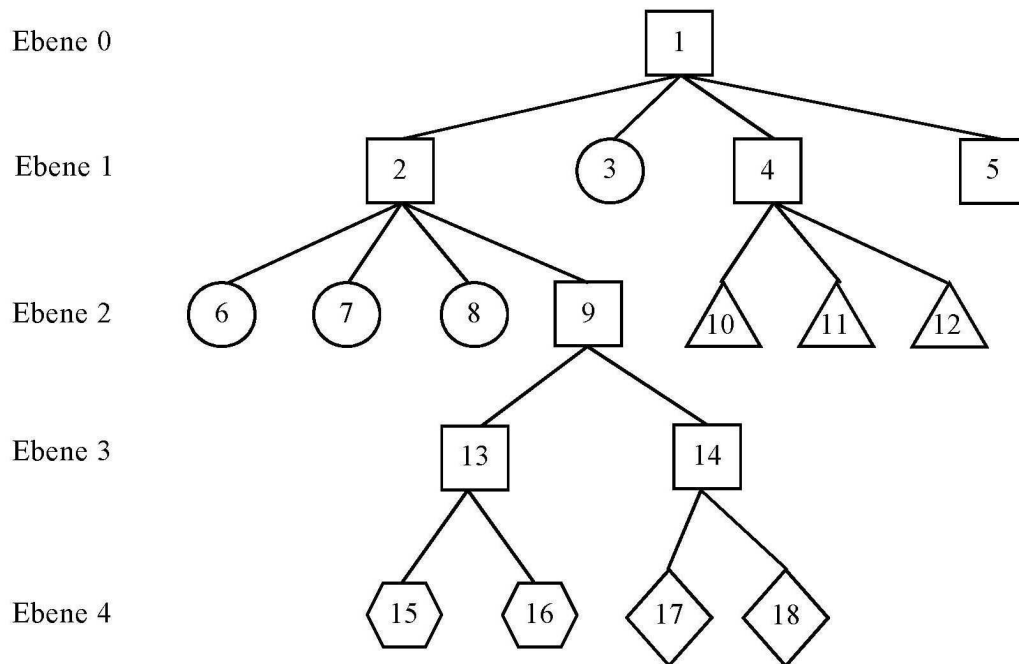
12. Datenverarbeitungseinrichtung mit einer elektronischen Datenverarbeitungseinheit, einem digitalen Speichermedium, auf dem eine Mehrzahl von Dateien abgespeichert oder abspeicherbar sind, und mit einer digitalen Dateiverwaltung, die eine Mehrzahl von Dateien für die Datenverarbeitungsfunktion der Datenverarbeitungseinrichtung enthält, **dadurch gekennzeichnet**, dass die Datenverarbeitungseinrichtung mittels der elektronischen Datenverarbeitungseinheit zum Durchführen des Verfahrens nach einem der Ansprüche 1 bis 10 eingerichtet ist, um die Dateisystemintegrität der die Dateien enthaltenden Dateiverwaltung zu überprüfen.

Es folgen 3 Seiten Zeichnungen

Anhängende Zeichnungen



Figur 1



Figur 2

Knoten	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
S	1	1	2	1	0	2	2	2	1	3	3	3	1	1	4	4	5	5
B _s	0	1	0	0	0	1	0	0	0	1	0	0	1	0	1	0	1	0

Index	0	1	2
H	5	6	7

Index	0	1	2	3	4	5	6	7	8	9	10	11	...	79	80	81	82	83
N	r	o	o	t	u	s	r	k	e	r	n	e	...	c	h	a	r	2
B _N	1	0	0	0	1	0	0	1	0	1	0	0	...	1	0	0	0	0

Figur 3