Daten sicher löschen

Stefan Schumacher
Stefan.Schumacher@Kaishakunin.com

KAISHAKUNIN.COM
IT-Sicherheitsberatung

Veröffentlicht in der GUUG UpTimes März 2007.

Literaturverzeichnis

Schumacher, Stefan (2007). „Daten sicher löschen“. Deutsch. In: *UpTimes* 1, Seiten 7–16. ISSN: 1860-7683. URL: http://kaishakunin.com/publ/guug-uptimes-loeschen.pdf (besucht am 07. 11. 2009).

```
@article{Schumacher-ut-loeschen,
    author = {Stefan Schumacher},
    title = {Daten sicher löschen},
    year = {2007},
    volume = "1",
    publisher = {German Unix User Group},
    journaltitle="UpTimes",
    issn="1860-7683",
    language = {german},
    url={http://kaishakunin.com/publ/guug-uptimes-loeschen.pdf},
    urldate="2009-11-07",
    pages="7\,--\,16",
}
```

Daten sicher löschen

Stefan Schumacher, <stefan@net-tex.de>

Die Hauptaufgabe eines Systemadministrators sind Schutz und Verfügbarmachung von Daten. Doch Daten müssen teilweise wieder sicher vernichtet werden. Dies ist beispielsweise der Fall, wenn alte Rechner oder Datenträger ausgesondert werden oder gesetzliche Aufbewahrungsfristen überschritten werden. In diesem Artikel wird gezeigt, welche Methoden zur sicheren Datenlöschung existieren, was bei den verschiedenen Datenträgern zu beachten ist und wie man die Datenvernichtung bereits im Vorfeld plant.

Wie Daten gespeichert werden

Um Daten vor unbefugtem Zugriff zu schützen, kann man kryptographische Methoden einsetzen. So kann man, wie in (Schumacher, 2006b) beschrieben, CGD unter NetBSD einsetzen um Dateisysteme zu verschlüsseln. Aber Dateien müssen nicht nur aufbewahrt und gesichert, sondern auch gelöscht werden. Wenn es sich dabei um sensible Daten handelt, sind spezielle Löschverfahren notwendig, die die Daten unwiederbringlich Löschen.

Wie man Daten vernichtet, hängt vom Speichermedium bzw. der verwendeten Speichertechnologie ab. Für elektromagnetische Datenträger (Festplatten, Disketten, Magnetbänder) gibt es drei Möglichkeiten, die physikalische Vernichtung mittels Curie-Temperatur oder Magnetfeld, oder die vernichtungsfreie Löschung der Daten durch Überschreiben.

Um diese Verfahren zu verstehen, muss man wissen, wie elektromagnetische Datenträger prinzipiell funktionieren. Dateien werden als Bitfolge abgespeichert, die physikalisch auf der Festplatte durch magnetische Flußwechsel gespeichert werden. Magnete bzw. ferromagnetische Stoffe haben einige physikalische Eigenschaften, die für dieses Thema relevant sind. Diese Grundlagen werden in jedem Einführungsbuch in die Elektrotechnik oder Nachrichtentechnik, wie beispielsweise (Steinbuch, 1967, Kap. 2.3 *Magnetische Bauelemente*), (Lindner†, Bauer† und Lehmann, 1999, Kap. 2 *Elektrische und Magnetische Felder*) oder dem für Informatiker bzw. Administratoren hervorragend geeignetem (Messmer, 1998, Kapitel 30.1, *Grundlagen magnetischer Datenaufzeichnung – Ferromagnetismus und Induktion*), vertieft dargelegt, sodaß ich hier lediglich einen kurzen Überblick geben werde.

In ferromagnetischen Stoffen, wie Eisen, Nickel oder Cobalt, gibt es mikroskopisch kleine, so genannte *Weißsche Bezirke* oder *Domänen*. Diese Bezirke sind vollständig magnetisiert, im Normalzustand aber nicht gleich ausgerichtet. Dadurch, das der Magnetismus der Bezirke quasi kreuz und quer im Stoff wirkt, ist der Stoff nach außen an sich nicht magnetisch. Allerdings können die Bezirke gleichgerichtet werden, indem man sie einem äußeren Magnetfeld aussetzt. Das Magnetfeld richtet alle Bezirke in derselben Richtung aus, nämlich der Flußrichtung des äußeren Magnetfeldes. Da nun alle Weißschen Bezirke in derselben Richtung ausgerichtet sind, wirkt der Elementarmagnetismus des Stoffes und der Körper ist nach außen magnetisch. Im Physikunterricht wurde dies oft demonstriert, in dem ein einfacher Eisennagel mehrfach mit einem Tafelmagneten in derselben Richtung bestrichen wurde. Der Nagel wurde magnetisiert und wirkte für kurze Zeit selbst wie ein Magnet. Die Weißschen Bezirke ferromagnetischer Stoffe bleiben unterschiedlich lange gleichgerichtet, der Körper also unterschiedlich lange magnetisch. Dieses ≫Verbleiben≪ des äußeren Magnetismus im Körper bezeichnet man als *Remanenz*[6]. Die *Sättigungsmagnetisierung* gibt an, welche Feldstärke notwendig ist um alle Weißschen Bezirke auszurichten.

Da elektromagnetische Datenträger ihre Daten lange speichern müssen, verwendet man hier spezielle Stoffe mit hoher Remanenz. Dies ist natürlich schön, wenn man Daten speichern will, allerdings problematisch, wenn man die Daten sicher löschen möchte. Aufgrund der Remanenz der Datenträger kann man auch nach mehrfachem Überschreiben der Sektoren noch vergangene Ausrichtungen mit teuren Laboreinrichtungen rekonstruieren. Dies ermöglicht es, bereits gelöschte und sogar überschriebene Daten wiederherzustellen.

Remanente Weißsche Bezirke können durch ein entgegengesetztes Magnetfeld in die andere Richtung ausgerichtet werden. Die dazu notwendige Stärke des äußeren Magnetfeldes wird als *Koerzitivfeldstärke* bezeichnet. Legt man ein äußeres Magnetfeld an, das sich einmal komplett dreht, werden auch die einzelnen Bezirke mitgedreht. Dies ist quasi so, als würde man einen Kompass nehmen und mit einem Magneten die Nadel nach Norden ausrichten. Dann führt man den Magneten am Kompass entlang und dreht die Nadel von Nord über Ost und Süd nach Westen und wieder nach Norden. Den Prozess des kompletten Herumdrehens um 360° bezeichnet man als *Hysterese* bzw. *Hysteresekurve*. Diese Kurve ist ebenfalls wieder Material-spezifisch, ermöglicht aber

[6] von lateinisch: *remanere*, verbleiben

sehr gute Aussagen über das magnetische Verhalten des Stoffes. Die Abbildung 1 zeigt eine Freihandskizze einer Hysteresekurve. In ihr sind die Feldstärke des äußeren Magnetfeldes und die Magnetisierung des Stoffes an den Achsen abgetragen. Außerdem wurden die Remanenz und die Koerzitivität eingezeichnet.

Um Daten auf einem elektromagnetischen Datenträger abzulegen, richtet ein Schreib-/Lesekopf die Weißschen Bezirke aus. Ein logisches Bit muß also immer mindestens so groß wie ein Weißscher Bezirk sein, in der Praxis besteht es aber aus mehreren zusammengefassten Bezirken. Die benötigte Feldstärke, die der Schreib-/Lesekopf zum Ausrichten aufwenden muß, entspricht der Koerzitivfeldstärke des Materials. Die Magnetisierung der Weißschen Bezirke des Datenträgers entspricht der Remanenz. Unterschiedlich ausgerichtete Weißsche Bezirke können sich gegenseitig beeinflussen, daher muß'mit steigender Kapazität der Datenträger die Remanenz des verwendeten Materials ebenfalls steigen.

Ein interessantes Phänomen magnetischer Stoffe ist die *Bit-Verschiebung* (engl. *bit shifting*). Dabei wandern die Weißschen Bezirke, die für die Aufzeichnung verwendet werden, mehrere Mikrometer auf der Platte herum, die Lesespuren verschieben sich also. Technisch kann man diesem Problem einfach begegnen. Entweder werden die inneren Spuren auf einem Zylinder weiter oder man baut eine so genannte Vorkompensation ein. Dabei verwendet man einen Controller, der den Lesekopf die Verschiebung der Bits ausgleichen lässt. Interessant ist dieses Phänomen für Löschverfahren durch Überschreiben.

Ebenfalls von Bedeutung für das gewählte Thema ist die so genannte Curie-Temperatur. Auch sie ist eine spezifische Eigenschaft ferromagnetischer Stoffe und bezeichnet die Temperatur, bei der schlagartig alle ferromagnetischen Eigenschaften verschwinden. Setzt man einen Datenträger der Curie-Temperatur aus, verschwindet die Ausrichtung der Weißschen Bezirke – und damit auch alle Daten. Für die Eisen-Kobalt-Legierung $Fe_{65}Co_{35}$ liegt die Curie-Temperatur bei 920°C und die Sättigungsfeldstärke bei satten 2,45 Tesla. Zum Vergleich: Kernspintomographen arbeiten mit ca. 1 Tesla, das Erdmagnetfeld hat gerade einmal 50 Mikro-Tesla.

Um die Packungsdichte auf Festplatten zu vergrößern, wurden die Codierungsverfahren optimiert. Es existieren inzwischen verschiedene Verfahren, um durch magnetische Flußwechsel logische Bitfolgen zu repräsentieren. Einige Optimierungen vollzogen sich auf physikalisch-elektrotechnischer Ebene, andere in der Codierungstheorie der Nachrichtentechnik. Es existieren beispielsweise verschiedene Verfahren, Bitfolgen zu codieren, sodaß die Anzahl magnetischer Flußwechsel reduziert werden kann. Bekannt und am besten dokumentiert sind die FM- und MFM-Formate – was auch daran liegt das sie für Disketten entwickelt wurden und vergleichsweise alt sind.

Im FM-Format werden Bits in einer so genannten *Bit-Zelle* abgelegt. Dies ist ein definierter Bereich von Weißschen Bezirken, die insgesamt meist 3 Mikrometer groß sind. Eine solche Bit-Zelle wird halbiert in ein so genanntes Takt-Bit und ein Daten-Bit. Das Takt-Bit wird immer gesetzt, sodaß ein Daten-Bit quasi immer zwischen zwei logischen Einsen eingekeilt ist. Ein Daten-Bit repräsentiert dann je nach Ausrichtung eine logische Null oder Eins. Um die Daten auszuwerten, benötigt man lediglich einen Zähler. Jedes ungerade Bit ist ein Takt-Bit und immer gesetzt. Jedes gerade Bit ist ein Daten-Bit und damit entweder eine Null oder eine Eins. Allerdings ist dieses Verfahren recht primitiv und die Datenmenge je Fläche nicht sehr groß. Das MFM-Verfahren verbessert die Datendichte, indem die Taktung geändert wird. Statt in jeder Bitzelle ein Takt-Bit zu setzen, wird es nur gesetzt, wenn das vorige und das folgende Daten-Bit Null gesetzt ist. *Zwei* Daten-Nullen in Folge werden also nur noch durch *ein* Takt-Bit repräsentiert. Damit reduziert sich der Platzbedarf und die Packungsdichte wächst. Das verwendete Codierungsformat ist von Bedeutung, wenn man Daten durch Überschreiben mit statischen Bitmustern löschen will.

Weiterführende Informationen sowohl zu Speicherverfahren als auch zu Codierungstypen finden Sie in (Messmer, 1998).

Das Betriebssystem selbst (hier NetBSD) verwendet ein Dateisystem, um Daten abzulegen. Das heißt, die physikalischen und elektrotechnischen Vorgänge der Datenspeicherung bleiben vor dem Betriebssystem und damit auch dem Anwender verborgen. Trotzdem muß das Betriebssystem Daten und *Metadaten* organisieren. Mit Daten bezeichnet man die eigentlichen Nutzdaten, also beispielsweise einen Brief, der als LaTeX-Datei abgespeichert wird. Metadaten bezeichnen »Daten über den Daten« – also Dinge wie Eigentümer der Datei, Zugriffsrechte, letzte Änderung und so weiter. Das Dateisystem organisiert die Ablage der Dateien in den *logischen* Blöcken des Dateisystems und den *physikalischen* Blöcken des Datenträgers. Für gewöhnlich wird beim Erzeugen des Dateisystems eine logische Blockgröße angegeben, meist sind dies 16Kb oder mehr. In diese logischen Blöcke werden Dateien geschrieben. Die Datei wird dazu je nach ihrer Größe in die benötigten Blöcke verteilt. Eine Datei von 100Kb Größe wird auf einem Dateisystem mit 16Kb Blockgröße auf $100/16 = 6,25$ Blöcke verteilt. Da nur ganze Blöcke angesteuert werden können, wird die Datei in 7 Blöcke geschrieben. Der letzte Block enthält dann nur die restlich verbleibenden 4Kb der Datei. In den frei bleibenden 12Kb des 7. Blockes können noch die Daten einer Datei liegen, die vorher hier abgelegt war. Diese Reste nennt man *File Slacks* bzw. Restblöcke. Auch sie sind von Bedeutung für das sichere Löschen, da sich unter Umständen in diesen Restblöcken Fragmente vertraulicher Dateien finden können. Im schlimmsten Fall ist ein solcher Block 64Kb groß und mit nur einem einzelnen »neuen« Byte belegt, dann lassen sich noch 65535 Byte »alte« Daten rekonstruieren.

Neben der Aufteilung in logische Blöcke sorgt das Dateisystem auch für die Zuordnung der belegten Blöcke zu den Dateien sowie für die Aufteilung der logischen Blöcke auf die physikalischen Blöcke der Datenträger. In den klassischen BSD-Dateisystemen (LFS, FFS) werden Indexknoten (*Index Node*) bzw. Inodes verwendet, um Dateien im Dateisystem zu organisieren. In ihnen finden sich die oben erwähnten Metadaten (Besitzer, Zugriffsrechte, Zugriffszeiten etc.) sowie eine Liste der belegten logischen Dateisystemblöcke. Soll eine

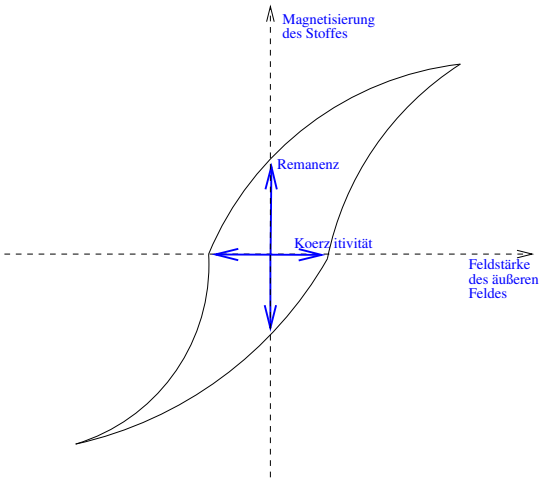

Abbildung 1: Freihandskizze einer Hysteresekurve

Datei gelesen oder geschrieben werden, wird über entsprechende Prozeduren (ffs_read() und ufs_bmap() für das BSD FFS) aus dem Inode die Liste der belegten logischen Blöcke ausgelesen. Die Identifikatoren der logischen Blöcke werden in die Physikalischen übersetzt. Anschließend werden die benötigten Blöcke vom Datenträger angefordert. Soll eine Datei gelöscht werden, Wird lediglich der verwendete Inode und die belegten logischen Blöcke als leer markiert. Die eigentlichen Daten in den Blöcken verbleiben auf dem Datenträger und können beispielsweise mit fsdb(8) rekonstruiert werden. Dazu muss man lediglich von Hand den entsprechenden Inode wieder aufbauen. Sollen Daten unwiederbringlich gelöscht werden, sind spezielle Löschverfahren notwendig, die in den folgenden Kapiteln erläutert werden. Weitere Informationen zu Dateisystem finden Sie im ≫Daemon-Book≪ (McKusick, Bostic, Karels und Quarterman, 1999, Kap. 7, ≫Local Filesystems≪ und Kap. 8, ≫Local Filestores≪), das den Aufbau der 4.4BSD-Dateisysteme (LFS, FFS, MFS) sehr detailliert beschreibt.

Physikalische Datenvernichtung

Ein sehr sicherer Weg der Datenvernichtung ist die physikalische Vernichtung des Datenträgers. Wenn das Medium oder Laufwerk nicht mehr funktioniert, kann es auch nicht mehr ausgelesen werden. Für verschiedene Laufwerks- bzw. Datenträgertypen gibt es unterschiedliche Arten und Wege, Daten zu vernichten.

Eine Warnung vorneweg: ich beschreibe den Einsatz von Feuer, Strom und mechanischer Gewalt, um Datenträger zu vernichten. Wenn Sie von diesen Dingen keine Ahnung haben, lassen Sie die Finger davon! Wenn Sie Datenträger mechanisch zerstören wollen, achten Sie auf Ihren Schutz! Tragen Sie zumindest ordentliche Arbeitshandschuhe und eine Schutzbrille! Ein umherfliegender Splitter einer Festplatten-Scheibe kann problemlos einen Augapfel zerstören.

Magnetische Datenträger

Wie in Kapitel beschrieben, speichern magnetische Datenträger ihre Daten in dem sie kleine magnetische Bereiche magnetisieren. Verliert der Datenträger nun seinen Magnetismus, sind schlagartig alle Daten unwiderruflich vernichtet.

Für jeden Magneten existiert als spezifische Eigenschaft die Curie-Temperatur. Bei der Curie-Temperatur verliert jeder Magnet schlagartig alle magnetischen Eigenschaften. Die Daten sind also definitiv und unwiederbringlich vernichtet. Einziger Nachteil ist die Höhe der Curie-Temperatur. Bei handelsüblichen Festplatten liegt sie bei ca. 1100 Kelvin bzw. $800°C$, ist also mit dem heimischen Backofen schwer zu erreichen. Hat man jedoch direkten Zugang zu einem Hochofen, Walzwerk oder Müllverbrennungsanlagen, kann man diesen Weg der Datenvernichtung wählen.

Eine zweite Möglichkeit der physikalischen Datenvernichtung ist der so genannte Degaußer. In ihm wird ein starkes Magnetfeld mit einer Flußdich-

te von mindestens 1 Tesla aufgebaut.

Dieses starke Magnetfeld wird um den Datenträger herum aufgebaut und durchläuft die Hystereseschleife (rotiert also mindestens einmal rundherum) und richtet somit *alle* Weißschen Bezirke neu aus. Damit ist jegliche Form von Daten in den Magnet-Partikeln vernichtet. Es gibt auch keine Daten in irgendwelchen Nebenspuren oder Remanenz bereits gelöschter Dateien. Dieses System ist definitiv sicher, zerstört aber die Festplatte. Denn das starke Magnetfeld vernichtet auch die Steuerelektronik und Servospuren der Festplatten.

Problematisch ist bei beiden Verfahren, das jeder Datensektor des Mediums erreicht werden muß. Das Magnetfeld muss jede Datenscheibe komplett durchfluten bzw. die Curie-Temperatur mehrere Minuten an jedem Punkt jeder Datenplatte anliegen. Da die Fesplattengehäuse schirmen, öffnen Sie die Gehäuse und bearbeiten Sie nur die Datenplatten im Degaußer bzw. im Hochofen.

Einen derartigen Degaußer kann man von speziellen Firmen mieten. Selbstbaulösungen mit etwas gewickeltem Kupferdraht sind zwar möglich, der Konsum aller MacGyver-Folgen qualifiziert aber noch nicht zur Elektro-Fachkraft. Lassen Sie also unbedingt die Finger von derartigen Experimenten!

Nachteil beider Verfahren ist die Zerstörung der Datenträger und ihre nicht-triviale Beschaffung. Zumindest ein Degaußer läßt sich jedoch gegen Entgelt mieten. Vorteil ist die hohe Sicherheit der Verfahren. Außerdem funktionieren sie auch mit defekten Platten, beispielsweise nach einem Head-Crash oder mit defekter Elektronik.

Optische Datenträger

Optische Datenträger speichern ihre Daten in einer Trägerschicht, die von einem Laser abgetastet wird. Wird diese Trägerschicht physikalisch vernichtet, sind auch die Daten weg. Die verbreitetsten optischen Datenträger, (wieder-)beschreibbare CDs und DVDs, bestehen prinzipiell aus drei Schichten: der unteren, durchsichtigen aber stabilen, Schutzschicht, der mittleren Trägerschicht und der oberen Deckschicht.

Eine CD lässt sich am einfachsten vernichten, wenn man die obere Deckschicht, also daß was normalerweise bedruckt ist oder beschrieben werden kann, zerstört. Die darunter liegende Trägerschicht oxidiert in der Regel sofort bzw. wird durch die mechanische Einwirkung ebenfalls vernichtet. Die Trägerschicht sieht aus wie hauchdünne Alu-Folie und zerfällt sehr schnell in sehr kleine Teile. Diese verteilen sich gerne in der Gegend und bevorzugt in Teppiche. Dort kann man ihnen nur sehr schwer mit dem Staubsauger zu Leibe rücken. Zerstören Sie also eine CD in einer Umgebung, in der dies nicht stört oder benutzen Sie einen großen Müllbeutel oder eine Abfalltonne als Hülle. Tragen Sie dabei Arbeitshandschuhe und schützen Sie ihre Augen! Sie können den CDs oder DVDs mit einem stabilen Schraubenzieher, robusten Messer oder auch einem Akkuschrauber bzw. einer Bohrmaschine mit Metallbohrer zu Leibe rücken. Entfernen Sie die Trägerschicht gründlich und zerbrechen Sie die CD danach.

In einem Handwerksbetrieb haben wir ein einfaches System entwickelt, um massenhaft CDs zu zerstören. Wir haben eine simple Spindel aus einem Holzbrett und einer 15mm-Gewindestange gebaut. Auf die Spindel wurden die CDs aufgestapelt und von oben mit einer großen Unterlegscheibe und einer Flügelmutter befestigt. Die gesamte Spindel wurde unter einer Standbohrmaschine sicher eingespannt und mehrfach mit einem 12mm Metallbohrer durchbohrt.

Weitere Möglichkeiten sind Häcksler bzw. Aktenvernichter, die auch CDs zerhacken können. Normales Zerbrechen der CD in größere Teile reicht nicht aus, um Spezialfirmen davon abzuhalten, Teile der CD wieder auszulesen.

Physikalische Vernichtung lässt sich auch durch Ätzen der CD erreichen. Hierzu muss die CD in eine Säure oder Ähnliches getaucht werden. Der Umgang mit Säuren und dergleichen ist aber nicht ungefährlich und daher hier nicht weiter von Bedeutung.

Ebenfalls nicht wirksam ist ein Zerkratzen der Unterseite der CD, da derartige Kratzer mit Füllmasse korrigiert werden können. Es muß die Trägerschicht des Mediums vernichtet werden. Dies lässt sich am besten über die Oberseite erreichen.

Wirksam, aber nicht empfehlenswert, ist die Mikrowelle. Durch die Mikrowellenstrahlung wird innerhalb von Millisekunden die Trägerschicht effizient und nicht-rekonstruierbar vernichtet. Dabei entstehen aber giftige und umweltschädliche Dämpfe sowie eine erhöhte Brandgefahr.

Magneto-Optische Datenträger

MO-Medien sind aufgrund der verwendete Speicherverfahren und technischen Umsetzungen vergleichsweise robust. Magneto-Optische Datenträger *schreiben* Daten magnetisch, *lesen* diese dann aber wieder optisch über Laser aus. Soll ein Datum auf dem Datenträger geschrieben werden, wird die Oberfläche von Laser über die Curie-Temperatur erhitzt. Damit verschwindet die Magnetisierung der Oberfläche und nach Herunterkühlen unter die Curie-Temperatur wird mit dem Magnetkopf das Datum geschrieben. Weitere Informationen zu MO-Laufwerken finden sich in (Messmer, 1998, Kapitel 31.8.4, *Magnetooptische Laufwerke*).

Da das Erhitzen der magnetischen Schicht über die Curie-Temperatur heraus jeglichen Magnetismus entfernt, reicht ein einmaliges Überschreiben der *gesamten* Platte aus.

Flash-EEPROM

Flash-EEPROMs werden in USB-Sticks oder Compact-Flash-Karten verwendet. Sie verwenden keinen Magnetismus um Daten abzulegen, sondern so genannte ≫Floating Gates≪ der Feldeffekt-Transistoren (FETs). Vereinfacht gesprochen ist das Floating Gate ein Leiter zwischen zwei Punkten (Source und Drain, Quelle und Abfluß genannt). Durch Anlegen einer Spannung kann der Leiter blockiert werden, eine gleich hohe, aber negative Spannung öffnet den Leiter wieder. Je nachdem ob der Leiter offen oder blockiert ist, wird eine logische 1 oder 0 repräsentiert. Ein FET ist also im Prinzip nichts anderes als ein sehr kleiner Schalter.

Daten können erst überschrieben werden, nachdem die alten Daten gelöscht wurden. Die Löschoperation kann dabei nur bei bestimmten EEPROMs auf Byte-Ebene erfolgen;

meist geschieht dies Sektor-weise, woher auch der Name *Flash* rührt. Da es bei den Feldeffekt-Transistoren nicht möglich ist, vorige Belegungen zu rekonstruieren, genügt ein einfaches Überschreiben der Daten mit einem beliebigen Bit-Muster. Praktisch lässt sich dies am einfachsten mit einem `dd if=/dev/zero of=/dev/sd0d` realisieren.

Beachten Sie aber, daß ein Flash-Speicher systembedingt nur eine begrenzte Anzahl von Lösch-Operationen durchführen kann. Dies rührt daher, daß bei jedem Durchtunneln der Oxid-Schicht des Floating-Gates diese Schicht beschädigt wird. Zur Zeit geben die Hersteller Zyklen von 100.000 bis 1.000.000 Löschvorgängen an.

Zerstörungsfreie Datenvernichtung

Um Daten auf magnetischen Datenträgern, wie Festplatten, Disketten oder Bändern, sicher zu löschen, müssen die Blockinhalte der Datei überschrieben werden. Dazu gibt es verschiedene Standards und Methoden, die unterschiedliche Sicherheitsstufen bieten.

Problematisch sind hierbei allerdings grundsätzliche Probleme: Die bereits erwähnten Datei-Reste in Blöcken sowie die Existenz nicht bekannter Kopien der Daten. Das Betriebssystem kann eine Datei unter anderem in den Swap-Speicher ausgelagert haben, sodaß sich die Daten auf der Swappartition befinden. Ebenso kann die Datei im temporären Verzeichnis /tmp abgelegt werden. Daher kann man zwar dafür sorgen, das die bekannte Datei sicher gelöscht wird, nicht jedoch alle anderweitig existierenden Kopien der Daten. Außerdem können nur Dateien überschrieben werden, die noch existieren. Bereits unsicher gelöschte Dateien können nicht ohne Weiteres überschrieben werden.

Außerdem funktioniert Überschreiben nicht, wenn das Dateisystem die Blockgröße bei jedem Speichervorgang neu zuordnet. Dies tut beispielsweise LFS. Um Daten sicher zu überschreiben, muß der Schreib-Cache der Platte deaktiviert werden. Verwenden Sie dazu unter NetBSD dkctl(8) in der Form `dkctl` *GERÄT* `setcache none`. Leider schalten nicht alle IDE-Platten den Schreib-Cache ab, obwohl dies implementiert sein müsste. SCSI-Platten können generell ihren Schreib-Cache abschalten.

Ein weiterer Kritikpunkt ist die Abhängigkeit der statischen Bitmuster vom Speicherverfahren der Festplatte. Eine Platte mit FM benötigt andere Bitmuster als MFM. Inzwischen gibt es eine Vielzahl unterschiedlicher Verfahren – auch proprietäre und daher unbekannte. Bisher existiert kein spezifisches Schema der Bitfolgen für die neueren Verfahren wie EEPRML, MTR oder Trellis, da sie kaum oder gar nicht öffentlich dokumentiert sind. Hier hilft nur der Einsatz von Zufallsdaten, die das Signal der Daten verrauschen.

Alles in allem sind Überschreibeverfahren im Dateisystem kritisch zu beurteilen. Sie sind nicht hinreichen, wenn der Datenträger ausgesondert wird. Sie sind nur dann nützlich, wenn der Datenträger weiter im Besitz bleibt und möglichst verschlüsselnde Dateisysteme ingesetzt werden.

Einfaches Überschreiben mit Nullen

Beim einfachen Überschreiben des Datenträgers wird jeder Block mit Nullen überschrieben. Damit schützt man sich zwar vor Programmen, die die Daten wiederherzustellen versuchen, nicht jedoch vor anderen Maßnahmen. Durch das gleichmäßige Bitmuster lassen sich die vorher gespeicherten Datenmuster relativ einfach wieder rekonstruieren. Dies wird zwar schwerer, je höher die Datendichte (und damit Kapazität) der Festplatte ist, ist aber nicht unmöglich.

Diese Methode ist nicht empfehlenswert.

US DoD 5220.22-M (E) und (ECE)

Ein vom US DoD (United States Department of Defense, US-amerikanisches Verteidigungsministerium) herausgegebener Standard. In 3 Durchläufen werden die Daten erst mit $0xFF$, dann mit $0x00$ und abschließend mit Zufallsdaten überschrieben.

Eine Erweiterung des (E)-Standards ist die (ECE)-Version, mit insgesamt 7 Durchläufen. Zuerst und zuletzt werden die 3 Durchläufe der (E)-Version ($0xFF$, $0x00$, Zufallsdaten) geschrieben. Als 4. Durchlauf werden wiederum Zufallsdaten geschrieben.

Die Zufallsdaten erhöhen die Sicherheit des (ECE)-Verfahrens, schützen jedoch nicht vor aufwendigen Laboranalysen. Außerdem werden beide Standards im (National Security Agency, 2001) nicht *im* Dateisystem eingesetzt, sondern direkt auf den Datenträger geschrieben.

Weiterhin wurde befohlen, daß das US-Militär mittels Überschreiben keine Daten löschen darf, die als *Geheim* oder *Streng Geheim* eingestuft wurden. Derartige Daten dürfen nur über Degaußer (siehe) vernichtet werden.

Richtlinie zum Geheimschutz von Verschlusssachen beim Einsatz von Informationstechnik

Die Richtlinie zum Geheimschutz von Verschlusssachen beim Einsatz von Informationstechnik des Bundesamtes für Sicherheit in der Informationstechnik spezifiziert 7 Durchläufe, je dreimal $0xFF$ und $0x00$ im Wechsel und abschließend ein Durchlauf mit $0xAA$.

Aufgrund der statischen Bitmuster nicht empfehlenswert.

Bruce Schneier

Bruce Schneiers Algorithmus aus (Schneier, 1997) überschreibt die Daten erst mit $0xFF$, $0x00$ und 5-mal mit im streng kryptographischen Sinne zufälligen Daten. Streng zufällige Daten sind jedoch aus signaltheoretischer Sicht nicht zwingend erforderlich, da die zu löschenden Daten »lediglich« durch unregelmäßige Signale verrauscht werden sollen. Hierzu genügen auch pseudo-zufällige Daten wie aus /dev/urandom generiert oder aus Anwendungsdateien wie Videos oder MP3-Dateien.

Peter Gutmann

Peter Gutmann ist natürlich kein Standard, sondern Autor von (Gutmann, 1996). In diesem Artikel analysiert er trefflich die Speichermethoden auf magnetischen Datenträgern. Sein Vorschlag zum sicheren Löschen ist ein 35-faches Überschreiben der Daten.

Hierbei werden in den ersten und letzten 4 Läufen Zufallsdaten geschrieben, ansonsten verwendet er spezielle Bitmuster. Diese speziellen Bitmuster sind an die Speicherverfahren der Festplatten (RLL, MFM) angepasst. Ziel ist es hierbei, mithilfe der Bitmuster das rekonstruierbare Signal der Nutzdaten zu verrauschen. Dabei werden beispielsweise erst die Folge 01010101 und danach das Komplement dazu, 10101010, geschrieben. Damit ist sichergestellt, daß jedes ≫Bit≪ einmal gedreht wurde.

Die *Idee* des Standards kann als sicher betrachtet werden. Allerdings ist die *Implementierung* durch die Einführung neuer Codierungsstandards veraltet, da nur MFM, (1,7)RLL und (2,7)RLL-kodierte Festplatten betrachtet werden. Anhand stochastischer Berechnungen (siehe (Reinke, 2004)) kann man davon ausgehen, daß 33 Schreibvorgänge mit zufälligen Daten mit einer Wahrscheinlichkeit von 99% jeden Bit der Festplatte zweimal ≫umschalten≪, die Nutzdaten also sehr sicher von der Platte schrubben. Sieben Durchläufe reichen aus, um jedes Bit der Platte einmal umzuorientieren. Abbildung 3 zeigt ein Skript, das eine Festplatte 33-mal mit Zufallsdaten überschreibt. Dies kann mit den unten beschriebenen Programmen wipe(1) und dd(1) oder neb-wip(1) problemlos realisiert werden.

Programme zur Datenvernichtung

Im folgenden werde ich einige Programme vorstellen, die Daten durch mehrfaches Überschreiben löschen. Diese Verfahrensweise muß aber kritisch betrachtet werden, da es einige Probleme zu beachten gilt. Überschreibeverfahren funktionieren nicht, wenn das zugrunde liegende Dateisystem die Blockgröße dynamisch alloziert. Es ist schlicht nicht möglich beim Überschreiben genau die Blöcke zu treffen in denen die Nutzdaten liegen.

Empfehlenswert sind die angegebenen Löschprogramme im Dateisystem nur, wenn der Datenträger weiterhin unter physikalischer Kontrolle bleibt. Soll er ausgesondert werden, sind zerstörende Maßnahmen sinnvoller oder zumindest wie in Kapitel beschrieben der Einsatz von dd(1), um direkt auf die Platte zu schreiben.

rm(1)

NetBSDs rm(1) kann mit der Option `-P` zu löschende Dateien mit $0xFF$, $0x00$ und Zufallsdaten überschreiben. Damit können Daten gelöscht werden, die vor einfacher Wiederherstellung durch Software geschützt werden sollen. Der Standard schützt nicht vor Laboranalysen. Sicheres Löschen von Dateien im Dateisystem ist in NetBSD mit Bordmitteln nicht zu bewerkstelligen. Lediglich komplette Datenträger oder zumindest Partition lassen sich mit Zufallsdaten schrubben.

destroy(1)

In Pkgsrc findet sich das Programm destroy(1) von Shane Kinney. Destroy überschreibt Dateien je Durchlauf einmal mit Null und einmal mit Zufallsdaten. Es können 1 bis 20 Durchläufe durchgeführt werden. Dieses Verfahren kann bei hinreichend vielen Durchläufen (mindestens 7) als sicher angesehen werden.

wipe(1)

Wipe von Tom Vier ist als pkgsrc/sysutils/wipe installierbar. Es löscht komplette Verzeichnisbäume und ist wesentlich flexibler konfigurierbar als Destroy. Wichtige Optionen sind:

- -I unterbindet Bestätigungsanfrage an den Benutzer
- -r löscht rekursiv Verzeichnisse
- -s schaltet Fortschrittsanzeige aus
- -v zeigt Fortschrittsanzeige für jede Datei an
- -d löscht Dateien nach dem Überschreiben
- -D behält Dateien nach dem Überschreiben
- -n löscht spezielle Dateien (Fifos, Links, etc.)
- -N überspringt spezielle Dateien (Fifos, Links, etc.)
- -k oder -K sperrt Dateien während des Überschreibens bzw. sperrt sie nicht
- -o oder -O schreibt die Bitmuster an STDOUT oder in eine Datei
- -p x wiederholt alle Läufe x-mal
- -x x führt x Durchläufe mit Zufallsdaten durch, gilt pro Lauf, der mit -p festgelegt wurde, -p4 -x2 führt $4 * 2 = 8$ Zufallsläufe durch.
- -b überschreibt einmal mit angegebenem Byte
- -l x Sicherheitsstufe
  - 0 Der Pseudozufallsgenerator wird nur einmal initialisiert
  - 1 Der Pseudozufallsgenerator wird für jede Datei initialisiert
  - 2 Der Pseudozufallsgenerator wird für jeden Durchlauf initialisiert
- -a oder -A überschreibt Daten bis der Plattenplatz verbraucht ist. bzw. tut dies nicht
- -z oder -Z überschreibt Dateien einmal mit Nullen oder deaktiviert diesen Modus
- -t oder -T aktiviert oder deaktiviert statische Durchläufe
- -B und -S setzt die Anzahl und Größe der Sektoren eines Block-Gerätes
- -C maximale Dateipuffergröße

Standardmäßig läuft wipe mit den Optionen `-ZdntVAkO -S512 -C4096 -l1 -x8 -p1`, d.h. keine Nullen, löscht Dateien und spezielle Dateien, mit statischen Durchläufen, Fortschrittsanzeige, sperrt dabei die Dateien, schreibt in eine Datei mit Blockgröße von 512, Dateipuffergröße von 4096, der Pseudozufallsgenerator wird für jede Datei initialisiert, es wird acht-mal mit Zufallsdaten überschrieben und das ganze Muster einmal durchlaufen.

Wipe kann auch als Quelle für andere Prozesse verwendet werden: So kann man mit `wipe -T -x1 -o 20971520 | dd of=cont2 bs=20971520` 20971520 Bytes, also 20MB, Pseudozufallsdaten generieren und mit dd(1) in eine Datei schreiben lassen. Mit Wipe als Quelle dauert der Prozess knapp 1,5 Sekunden, mit /dev/urandom 26 Sekunden.

dd(1)

Wie in Kapitel beschrieben, haben Überschreibe-Verfahren einige grundsätzliche Probleme. Sie funktionieren nicht auf Dateisystemen mit dynamischer Größenanpassung, können nicht-bekannte Kopien nicht überschreiben und erwischen keine Datei-Reste in Blöcken.

Möchte man sicherstellen, das alle Daten auf einer Festplatte – oder zumindest einer Partition – überschrieben werden, muß man zu des Administrators Superwaffe – dd(1)

– greifen. Man kann dd verwenden, um direkt auf Raw-Geräte zu schreiben, also unterhalb des Dateisystems. Ein simples `dd if=/dev/wd0h of=/tmp/wd0h.dd` schreibt den *gesamten* Inhalt der Partition wd0h in eine Datei. Umgekehrt kann man mit `dd if=/tmp/container of=/dev/wd0h` eine Datei auf die Partition schreiben lassen. Mithilfe der beiden Geräte `/dev/zero` und `/dev/urandom` kann man eine unendliche Folge von Nullen bzw. Pseudozufallsdaten erzeugen lassen und auf eine Partition oder Festplatte schreiben lassen. Mit den Befehlen `dd if=/dev/zero of=/dev/wd0h ; dd if=/dev/urandom of=/dev/wd0h` wird die Partition wd0h zuerst mit Nullen und dann mit Zufallsdaten überschrieben. Dabei wird *jeder* Block der Partition, auf den die Festplatte Zugriff hat, überschrieben. Ein derartiges Verfahren ist von der Technik her das sicherste, da die Daten direkt über das Raw-Gerät überschrieben werden. Es müssen lediglich hinreichend viele Durchläufe benutzt werden, wie beispielsweise in Abbildung 2 gezeigt wird. Hierbei wird eine Schleife eingesetzt, die 5-mal durchlaufen wird. In der Schleife wird die gesamte zweite IDE-Festplatte (*wd1* ist die zweite Platte, die Partition *d* die gesamte Platte), dreimal mittels dd(1) beschrieben. Zuerst mit Nullen aus /dev/zero, dann mit Pseudozufallsdaten aus /dev/urandom und abschließend mit wipe(1) als Pseudozufallsgenerator. Die Option bs=2m setzt die Blockgröße auf 2 Megabyte. Dies erhöht den Schreibdurchsatz auf die Festplatte erheblich, als Wert sollte hier immer die Größe des Platten-Caches verwendet werden. Die Option progress gibt alle 50 Blöcke (also 50 * 2m = 100MB) einen kleinen Punkt als Fortschrittsanzeige aus. In der letzten Zeile wird das Programm wipe(1) als Zufallsgenerator verwendet, da es wesentlich schneller ist als /dev/urandom. Die Option `-o 2097152` setzt zwar die Größe der zu erzeugenden Zufallsdaten auf 2097152 Byte, also genau 2MB, aber die Option `-a` überlädt `-o` und generiert solange Daten bis das Gerät voll ist und dd(1) abbricht. Die meisten Festplatten lassen sich schneller schrubben wenn beide Programme mit einer Blockgröße von 2MB arbeiten.

Abbildung 3 zeigt ein einfaches Shellscript, das insgesamt 33-mal eine Festplatte mit Zufallsdaten überschreibt. Dazu wird wipe(1) als Quelle für Zufallsdaten verwendet. Mit der Option `-o 2097152` wird wipe(1) angewiesen, 2MB große Blöcke zu schreiben. Dies erhöht den Durchsatz dramatisch gegenüber anderen Einstellungen. Auf meinem »IBM Thinkpad X24«-Laptop steigt der Schreibdurchsatz von 6MB auf 10MB, wenn die Blockgröße auf 2MB gesetzt wird. Der einzustellende Wert hängt jedoch von der verwendeten Festplatte ab, sodaß man ein paar Blockgrößen ausprobieren sollte. Das Programm dd(1) gibt beim Beenden immer die durchschnittlich geschriebenen Bytes je Sekunde aus, sodaß sich etwas Experimentieren lohnt. Außerdem wird im Skript der Schreibcache jeder Platte abgeschaltet und zusätzlich nach jedem Durchlauf auf die Platte geschrieben. Die Maßnahme ist notwendig, da nicht alle IDE-Platten den Schreibcache wirklich abschalten.

neb-wipe(1)

NetBSD-Wipe ist über Pkgsrc-WIP erhältlich. Es schreibt direkt auf ein Raw-Gerät. Dazu kann es Gutmann-Bitmuster verwenden oder Zufallsdaten erzeugen. Es können beliebig viele Durchläufe geschrieben werden. Der verwendete Pseudozufallszahlengenerator ist wesentlich schneller als /dev/urandom. Mit `neb-wipe -r 33 /dev/sd0d` überschreibt man die erste SCSI-Platte 33-mal mit Zufallsdaten.

buffer(1)

Buffer ist ein Zwischenpuffer, der für Magnetbänder entworfen wurde. Der Eingabefluß wird geglättet und zwischengespeichert, so daß das Bandlaufwerk nicht ständig vor- und zurückspulen muß, falls gelesene Daten ausbleiben.

Anstatt dd(1) sollte man buffer(1) verwenden, um direkt auf Bänder zu schreiben, insbesondere wenn erst Zufallsdaten generiert werden müssen. Bei schnellen Bändern empfiehlt es sich, die Zufallsdaten erst in eine Datei zu schreiben und diese dann auf das Band zu bringen.

Mit `wipe -T -x1 -o 20971520 -a | dd of=wipedatei bs=1m count=2048 ; buffer -i wipedatei -o /dev/nrst0` wird zuerst eine 2GB große Datei mit Zufallsdaten generiert und anschließend mit buffer auf das erste Bandlaufwerk geschrieben.

Die NetBSD/Schrubber Live-CD

Um eine Festplatte komplett zu putzen, benötigt man ein System mit bestimmten Werkzeugen. Zu diesem Zwecke habe ich eine NetBSD/Live-CD erstellt, die auf einem minimalistischen NetBSD 3.1 basiert und alle notwendigen Werkzeuge mitbringt.

Mit der CD kann man einen beliebigen PC booten und zum Platten schrubben einsetzen. Einzige Voraussetzung ist ein bootfähiges CDROM-Laufwerk und unterstützte IDE- bzw. SCSI-Controller, was aber meist der Fall ist. Das System ist sehr minimalistisch, es gibt keine Manpages, keine einkompilierten Netzwerkkarten und sonstige Hardware, die nicht benötigt wird. Im Kernel sind alle verfügbaren IDE- und SCSI-Controller aktiviert, sodaß nahezu jedes Massenspeichergerät angesprochen werden kann.

Auf der CD finden sich neben der Allzweckwaffe dd(1) auch die oben vorgestellten Programm Wipe und Neb-Wipe, sowie das Shell-Skript `wipe-33.sh`. Es verwendet Wipe, um 33-mal alle übergebenen Partitionen mit Zufallsdaten zu überschreiben. Die Anzahl der Durchläufe kann mit der Umgebungsvariable `DURCHLAEUFE` festgelegt werden; die zu überschreibenden Partitionen werden als Argument übergeben. Das Skript prüft alle Geräte auf Existenz und gibt ggf. eine Warnmeldung aus. Falls das Gerät existiert, wird der Schreibcache mit dkctl(8) deaktiviert und eine Schleife mit der entsprechenden Anzahl der Durchläufe gestartet. Nach jedem Durchlauf wird mit dkctl(8) der Schreibcache zum synchronisieren gezwungen. Dies ist leider notwendig, da einige IDE-Platten den Schreibcache nicht abschalten können. Mit diesem Behelf wird zumindest am Ende eines Durchlaufs dafür gesorgt, das der Cache auf die Platte geschrieben wird. Das Skript gibt einige Statusmeldungen nach jedem Durchlauf aus, um den Fortschritt anzuzeigen.

```
# for i in ‘seq 1 5‘
> do
> dd if=/dev/zero of=/dev/wd1d bs=2m progress=50
> dd if=/dev/urandom of=/dev/wd1d bs=2m progress=50
> wipe -T -x 1 -o 2097152 -a | dd of=/dev/wd1d bs=2m
> done
```

Abbildung 2: Eine Partition mit dd(1) überschreiben

```
#!/bin/sh

if [ ! $DURCHLAEUFE ]; then
        DURCHLAEUFE=33
fi

for i in $@
do
        if [ ! -b /dev/$i ]; then
                printf "\nDas Geraet /dev/$i existiert nicht!\n\n"
        else
                /sbin/dkctl $i setcache r

                for j in ‘seq 1 $DURCHLAEUFE‘
                do
                        printf "\n$j. Durchlauf auf Platte /dev/$i"

                        /usr/pkg/bin/wipe -T -x1 -o 2097152 -a | \
                        /bin/dd of=/dev/$i bs=2m

                        /sbin/dkctl $i synccache force
                done
        fi
done
```

Abbildung 3: Eine Partition 33-mal mit wipe(1) und dd(1) überschreiben

Mit `export DURCHLAEUFE=7 && ./wipe-33.sh wd0d sd0d` wird die erste IDE- und die erste SCSI-Platte komplett in 7 Durchläufen geschrubbt.

Die CD lässt sich über das Passwort-lose Root-Konto benutzen. Es werden 8 Terminals initialisiert, die über STRG+ALT+F[1-8] erreicht werden können. Die Terminals F4-F8 sind mit 50 statt 25 Textzeilen initialisiert, was allerdings auf einigen sehr alten Grafikkarten (z.B. ATI) zu Problemen führt. Das erste Terminal (STRG+ALT+F1), auf dem die Bootmeldungen erscheinen, ist die Konsole. Dort schlagen auch die Kernel-Meldungen an den Syslogd auf. Von daher ist es ratsam, ein anderes Terminal zum Arbeiten zu verwenden.

Vorbeugende Maßnahmen

Verschlüsselung

Die einfachste Maßnahme, um die Datenvernichtung zu vereinfachen, ist die Datensparsamkeit. Daten die nie gespeichert wurden, müssen auch nie gelöscht werden. Da nicht alle sensiblen Daten nicht gespeichert werden können, ist der Einsatz verschlüsselnder Dateisysteme geboten. Damit wird der direkte Zugriff auf die Daten verhindert. NetBSD bietet mit CGD und CFS zwei Varianten um Dateisysteme zu verschlüsseln, mit CGD können gesamte Partitionen – auch /tmp und die Swap-Partition – verschlüsselt werden. Die Artikel (Schumacher, 2004) und (Schumacher, 2006b) beschreiben die Funktionsweise verschlüsselnder Dateisysteme sowie den Einsatz von CGD und CFS unter NetBSD.

Auch wenn Daten verschlüsselt wurden, müssen sie sicher gelöscht werden. Dies gilt insbesondere für die Passwort-Hashes oder Schlüssel der Dateisysteme. Außerdem kann eine Festplatte oder ein Magnetband 30 Jahre halten – und niemand kann garantieren, das die eingesetzten kryptographischen Verfahren dann noch sicher sind. Eine adäquate Vorgehensweise ist hier das mehrfache Überschreiben der Festplatte mit Zufallsdaten.

Daten organisieren

Wenn Sie große Datenmengen verwalten müssen, organisieren Sie die Datenstruktur entsprechend der Schutzstufen. Legen Sie eindeutige Kriterien zur Bestimmung der Schutzstufe fest und teilen Sie beispielsweise 5 Schutzstufen (0 – nicht schutzwürdig, 4 – Streng Geheim) ein. Legen Sie anschließend fest, wie die Schutz-

stufen zu löschen sind (0 – einmaliges Überschreiben Zufallsdaten; 4 – sicheres Löschen der Datenträger durch mehrfaches Überschreiben und anschließende physikalische Vernichtung der Datenträger durch Degaußer). Setzen Sie von vornherein ein Verschlüsselungsgebot für alle Daten ab Stufe 2 durch. Verschlüsseln Sie ebenfalls alle Sicherungs-, Arbeits- oder sonstigen Kopien der Daten. Verbieten Sie in Ihrer Sicherheitsrichtlinie, die geschützten Daten anderweitig abzulegen als erlaubt. Beachten Sie ebenfalls Medienbrüche (Ausdrucke). Tragen Sie Sorge, das auch Ausdrucke von geschützten Dateien geschützt und entsprechend vernichtet werden. Ebenfalls hilfreich ist es, die Schutzstufe im Dateinamen zu hinterlegen, sodaß eine Datei beispielsweise `PersAkte-Müller.SS4.txt` heißt. Ebenfalls hilfreich ist insbesondere in großen Organisationen eine Markierung der Datenträger, die ihre Schutzstufe wiedergibt. Eine rote ≫S4≪ mit Folienstift auf die Festplatte gemalt oder auf einem hitzebeständigen, resistenten Etikett dürfte ausreichen.

Medienbrüche beachten

Überprüfen Sie alle Kopierer und Drucker auf Festplatten – größere Modelle für den Netzwerkeinsatz haben oft eingebaute Festplatten. Diese müssen natürlich ebenso sicher gelöscht werden wie die restlichen Datenträger.

Außerdem sollten müssen Sie Medienbrüche beachten, das heißt neben den Festplatten gegebenenfalls auch Ausdrucke vernichten. Hierfür gibt es Aktenvernichter, die idealerweise $1mm^2$ kleine Schnitze erzeugen.

Wenn Sie einen externen Dienstleister mit der Vernichtung Ihrer Datenträger betrauen ist es empfehlenswert, die Datenträger vorher durch mehrfaches Überschreiben zu löschen. Es sind auch schon in Sicherheitsunternehmen Datenträger gestohlen worden.

Datensicherung und Datenvernichtung

Auch sensible Daten müssen gesichert werden. Dazu bedarf es unbedingt des Einsatzes von Verschlüsselung. Verschlüsseln Sie ausnahmslos *jede* Kopie der sensiblen Daten – auch jene die Sie in Ihrer geschützten Umgebung aufbewahren.

Richten Sie gegebenenfalls verschiedene Sicherungskreise ein. Dies ist notwendig, wenn Sie Daten haben, deren Lagerfrist kürzer ist als Ihre gängige Archivierungsdauer. Der Gesetzgeber schreibt beispielsweise Aufbewahrungsfristen von maximal 2 Jahren für Einträge in Personalakten vor. Wenn Sie eine Personalakte mit Eintrag 5 Jahre archivieren, kann der betreffende Mitarbeiter Sie verklagen und wird problemlos gewinnen. Sorgen Sie daher dafür, das Daten mit Verfallsfrist entsprechend gesondert gesichert und aufbewahrt werden. Überwachen Sie die Verfallsdaten im Zuge Ihrer regelmäßigen Inventur der Sicherungsmedien! Markieren Sie sensible Medien oder Medien mit Verfallsdatum eindeutig. Dies kann mit bunten Etiketten, einem einfachen Folienschreiber oder über Softwarelabel (beispielsweise dump -L 'SS4 2009-01-31') geschehen.

Mein Handbuch zur Datensicherung (Schumacher, 2006a) widmet sich diesem Thema ausführlich und beschreibt, wie verschiedene Programme (Bacula oder Amanda) konfiguriert werden müssen, um verschiedene Sicherungskreise einzurichten.

Fazit

Sicheres Datenlöschen ist mit etwas Aufwand möglich. Selbst nichtzerstörende Verfahren wie mehrfaches Überschreiben bietet Schutz gegen übliche Laboranalysen. Das bedeutet, das mindestens 7-mal, besser noch 33-mal, Zufallsdaten auf das Raw-Gerät geschrieben werden müssen.

Sind die Daten so wertvoll, daß sie ausführliche und teure Laboranalysen rechtfertigen, hilft nur die physikalische Vernichtung mittels angemietetem Degaußer oder Curie-Temperatur, falls man darauf Zugriff hat. Man kann die Datenvernichtung auch an externe Dienstleister abtreten. Beachten Sie aber, das es dann zu unüberwachbaren Sicherheitsproblemen kommen kann – überschreiben Sie die Datenträger also sicherheitshalber!

Eine Richtlinie zum sicheren Löschen ist in größeren Organisationen zwingend erforderlich. Bewerten Sie anfallende Daten bezüglich ihrer Schutzstufe und legen Sie passende Löschverfahren fest. Markieren Sie sensible Datenträger und berücksichtigen Sie sensible Daten und ihre Verfallsdaten in der Sicherungsstrategie.

Stefan Schumacher beschäftigt sich in seiner Freizeit mit japanischen Kampfkünsten sowie mit NetBSD und PostgreSQL. Seine persönlichen Webseiten sind unter www.nettex.de bzw. www.cryptomancer.de erreichbar. Seine PGP-Schlüssel-ID ist 0xB3FBAE33.

Literaturverzeichnis

[Gutmann 1996] GUTMANN, Peter: Secure Deletion of Data from Magnetic and Solid-State Memory. In: *USENIX Security Symposium Proceedings*. San Jose, California : USENIX, July 1996, S. 77–90. – URL http://www.cs.auckland.ac.nz/~pgut001/pubs/secure_del.html. – Zugriffsdatum: Jan. 2007

[Lindner† u. a. 1999] LINDNER†, Studiendirektor H. ; BAUER†, Dr. H. ; LEHMANN, Prof. Dr. C.: *Taschenbuch der Elektrotechnik und Elektronik*. 7. völlig neubearbeitete Auflage. Leipzig : Fachbuchverlag Leipzig, 1999. – ISBN 3-446-21056-3

[McKusick u. a. 1999] MCKUSICK, Marshall K. ; BOSTIC, Keith ; KARELS, Michael J. ; QUARTERMAN, John S.: *The Design and the Implementation of the 4.4BSD Operating System*. 1. Auflage. Readin, Massachusetts : Addison-Wesley, 1999. – ISBN 0-201-54979-4

[Messmer 1998] MESSMER, Hans P.: *Das PC-Hardware-Buch*. 1. Nachdruck der 5. Auflage von 1997. Bonn : Addison-Wesley, 1998. – ISBN 3-8273-1302-3

[Reinke 2004] REINKE, Dr. T.: Sicheres Löschen magnetischer Datenträger. (2004). – URL http://fhh.hamburg.de/stadt/Aktuell/weitere-einrichtungen/datenschutzbe%auftragter/informationsmaterial/informationstechnik/sicheres-loeschen-pdf,prop%erty=source.pdf. – Zugriffsdatum: Jan. 2007

[Schneier 1997] SCHNEIER, Bruce: *Applied Cryptography*. Addison-Wesley, 1997. – ISBN 0-471-11709-9

[Schumacher 2004] SCHUMACHER, Stefan: Einführung in kryptographische Methoden. (2004). – URL http://cryptomancer.ath.cx/~stefan/21c3/21c3-kryptograhie-paper.pdf. – Zugriffsdatum: Okt. 2006

[Schumacher 2006a] SCHUMACHER, Stefan: Methoden zur Datensicherung – Strategien und Techniken für NetBSD. (2006), S. 154. – URL http://www.net-tex.de/backup.pdf

[Schumacher 2006b] SCHUMACHER, Stefan: Verschlüsselte Dateisysteme für NetBSD. In: *GUUG UpTimes Dezember 2006* (2006). – URL http://www.net-tex.de/netbsd/advocacy/guug-uptimes-cgd_cfs.pdf. – Zugriffsdatum: Jan. 2007

[Steinbuch 1967] STEINBUCH, Herausgeber Dr.-Ing. K.: *Taschenbuch der Nachrichtenverarbeitung*. 2. überarbeitete Auflage von 1968. Berlin/Heidelberg : Springer-Verlag, 1967. – nur noch antiquarisch verfügbar

[U.S. DoD 5220.22-M 2001] NATIONAL SECURITY AGENCY: *U.S. DoD 5220.22-M National Industrial Security Program Operating Manual (≫NISPOM≪)*. National Security Agency, 2001. – URL http://tscm.com/NISPOMSU.html. – Zugriffsdatum: Feb. 2007