Verschlüsselung von Laufwerken

FreeBSD bietet durch GEOM ausgezeichnete Möglichkeiten, Daten vor unberechtigten Zugriffen zu schützen. Wenn das Betriebssystem läuft, schützen Zugriffsrechte und vorgeschriebene Zugriffskontrollen (MAC) die Daten. Die traditionelle Zugriffskontrollen des Betriebssystems schützen allerdings nicht vor einem Angreifer, der Zugriff auf den Rechner hat. Der Angreifer kann eine Festplatte einfach in ein anderes System einbauen und dort die Daten analysieren. Bedenken Sie bitte, wie schnell z.B. ein Laptop mit wichtigen Daten gestohlen werden kann.

Die für FreeBSD verfügbaren kryptografischen GEOM-Klassen gbde und geli sind in der Lage, Daten auf Dateisystemen auch vor hoch motivierten Angreifern zu schützen, die über erhebliche Mittel verfügen. Dieser Schutz ist unabhängig von der Art und Weise, durch die ein Angreifer Zugang zu einer Festplatte oder zu einem Rechner erlangt hat, solange die Daten nicht entschlüsselt eingebunden sind (der Rechner also rebootet werden muss oder abgeschaltet ist). Im Gegensatz zu anderen Systemen, die auf einzelnen Dateien oder innerhalb von Dateicontainern arbeiten, verschlüsseln gbde und geli unterhalb des Dateisystems transparent das gesamte Laufwerk.

Features die beide Systeme gemeinsam haben sind u.a.:

- entkoppelte und damit ohne Neuverschlüsselung wechselbare Passphrase
- mehrere, unabhängige Schlüssel (geli: 2, gbde: 4)
- Verwendung eines zufälligen, nicht gespeicherten Schlüssels
- Passphrase und/oder Schlüsseldatei als Zugriffsschutz

Nachfolgend Unterschiede zwischen den Klassen:

Verschlüsselung: gdbe

Nur AES wird als möglicher Algorithmus unterstützt, hierbei wird ein Multikey-Setup verwendet. Die Keysektoren werden mit AES256 geschützt, jeder andere Sektor mit einem eigenen AES128 Schlüssel.

Es werden Zonen innerhalb des Laufwerkes definiert, um logisch zusammenhängende Daten über das gesamte physikalische Laufwerk zu verstreuen.

Unterstützt Passphrase und Passphrase/Schlüsseldatei, aber keine nur mit einer Schlüsseldatei angelegte Konfiguration.

Verschlüsselung: geli

Unterstützung des crypto(9)-Frameworks - verfügt das System über kryptografische Hardware, wird diese von geli automatisch verwendet.

Unterstützung verschiedener Algorithmen mit definierbarer Schlüssellänge (derzeit AES, Blowfish sowie 3DES) in einem Single-Key Setup.

Eine einfache Möglichkeit, auch die root-Partition zu verschlüsseln wenn der Kernel von einem externen Medium geladen wird.

Durch einfache Sektor-zu-Sektor-Verschlüsselung mit einem Key und ohne Spreading sehr schnell.

Optional Verifikation der Datenintegrität durch HMAC.

Swap (Auslagerungsspeicher) verschlüsseln

Auch die Verschlüsselung des Swap dient dem Schutz sensitiver Informationen. Wenn z.B. eine Anwendung ein Passwort verlangt und FreeBSD lagert Daten auf die Festplatte aus, um im Hauptspeicher Platz für andere Anwendungen zu schaffen, kann es passieren, dass Ihr Passwort im Klartext in den Swap geschrieben wird. Sie können den Swap sowohl mit geli als auch gdbe wirksam verschlüsseln - sie müssen nur die fstab verändern.

Weitere Informationen

Wir empfehlen wir Ihnen einen Blick in das exzellente FreeBSD Handbuch:
http://www.freebsd.org/doc/de/books/handbook

Weitere ausführliche Quellen finden Sie hier:

http://phk.freebsd.dk/pubs/bsdcan-04.slides.geomtut.pdf

http://phk.freebsd.dk/pubs/bsdcan-04.slides.geom.pdf

http://phk.freebsd.dk/pubs/bsdcan-04.slides.gbde.pdf

http://events.ccc.de/congress/2005/fahrplan/attachments/586-paper_Complete_Hard_Disk_Encryption.pdf

http://wikitest.freebsd.org/gvirstor

FreeBSD GEOM - Deutsch

FreeBSD

GEOM

Einführung

GEOM ist ein modulares und objektorientiertes Framework im FreeBSD Kernel, über das seit FreeBSD 5.0 sämtliche Zugriffe auf Speichermedien verarbeitet werden. Es sitzt zwischen dem Dateisystem und den beiden I/O-Subsystemen CAM und ATA und abstrahiert die Geometrie des eigentlich verwendeten Mediums. Der vorliegende Flyer soll Ihnen in kurzer Form einige der nahezu unbegrenzten Möglichkeiten von GEOM aufzeigen und stellt die wichtigsten GEOM-Klassen vor.

Das GEOM-Framework im Überblick

Ausserhalb akademischer Kreise stellt der interessante Part an GEOM die zur Verfügung gestellten Klassen und die über sie erreichbare Funktionalität dar. Wir beschränken uns daher im folgenden auf eine Beschreibung der, wie wir finden, wichtigsten und interessantesten Klassen und vernachlässigen komplett wie das *geometry layer* (GEOM) innerhalb des Kernels funktioniert. Dies ist im Dæmon-Book mehr als ausreichend erläutert.

RAID 0 - gstripe

Bei RAID 0 handelt es sich, wie der Name schon sagt, um ein RAID mit null Redundanz. RAID 0 bietet unter Umständen gesteigerte Transferraten, indem mehrere Festplatten zusammengeschlossen und Schreiboperationen auf allen parallel durchgeführt werden (engl. *striping* - „in Streifen zerlegen“). Fällt eine der Festplatten aus, können ohne deren Teildaten die Nutzdaten nicht mehr vollständig rekonstruiert werden. RAID 0 ist daher nur in Anwendungen zu empfehlen, bei denen Datensicherheit keine Rolle spielt oder durch eine geeignete Form von Datensicherung anderweitig gewährleistet wird.

Die Anwendung von gstripe ist simpel: Laden das Moduls (kldload geom_stripe), mountpoint anlegen, neuen Datenträger erzeugen (gstripe label), Partitionstabelle mit bsdlabel schreiben, Dateisystem mit newfs anlegen. Anmerkungen: Der FreeBSD-installer "sysinstall" ist noch nicht in der Lage GEOM-Datenträger anzulegen; booten von gstripe-Laufwerken ist nicht möglich.

RAID 1 - gmirror

Ein RAID-1-Array muss aus mindestens zwei Festplatten bestehen, die exakt die gleichen Daten enthalten (engl. *mirror* - *Spiegel*). RAID 1 bietet die volle Redundanz der gespeicherten Daten, während die Kapazität des Arrays so groß ist wie die kleinste beteiligte Festplatte. Fällt eine der Platten aus, können die anderen weiterhin die Daten liefern. RAID 1 bietet eine hohe Ausfallsicherheit. Zum Totalverlust der Daten führt nur der Ausfall aller Platten.

Die Anwendung von gmirror ist sehr einfach analog zu gstripe, ist bei root-Partitionen problemlos möglich und ein Ersatz einer ausgefallenen Platte sind einfach zu bewerkstelligen. Alle Operationen wie Fehlererkennung, Erkennung ausgefallener Platten und Wiederherstellung des RAID 1 laufen automatisch ab.

RAID 3 - graid3

RAID 3 ist ein RAID 0 mit Redundanz, welche auf einer zusätzlichen dedizierten Festplatte gespeichert wird. Da ein paralleles und kein unabhängiges Verfahren verwandt wird, stellt diese Paritätsplatte im Gegensatz zu RAID 4 keinen Flaschenhals dar, eine Verteilung der Paritätsdaten wie bei RAID 5 ist daher unnötig.

Die Bedienung ist analog zu gstripe und gmirror ebenfalls sehr einfach und alle Operationen erfolgen automatisch. Die Paritätsinformationen werden auf der letzten Platte gespeichert. Die Anzahl der Platten berechnet sich nach der Formel (`2^n + 1`), das heisst 3, 5, 9, 17 usw.

Kombinierte RAID-Level

Die Klassen gstripe, gmirror und graid3 können nahezu beliebig miteinander kombiniert werden. Hierdurch sind dann die RAID-Level 0+1, 10, 0+3 und 30 erreichbar.

RAID 5 - gvinum

RAID 5 bietet sowohl gesteigerten Datendurchsatz beim Lesen von Daten als auch Redundanz bei relativ geringen Kosten und ist dadurch die beliebteste RAID-Variante. Sowohl die Nutzdaten als auch die Paritätsinformationen werden über alle Platten verteilt. Da die Paritätsinformationen beim Lesen nicht benötigt werden, stehen alle Platten zum parallelen Zugriff zur Verfügung. Dieser (theoretische) Vorteil greift allerdings nicht bei kleinen Dateien, erst bei größeren Dateien tritt eine nennenswerte Beschleunigung ein. Bei RAID 5 ist die Datensicherheit des Arrays beim Ausfall von maximal einer Platte gewährleistet.

Netzwerkexport - Geomgate

ggate ist eine GEOM-Klasse, welche das Exportieren von Laufwerken über das Netzwerk ermöglicht. Im Gegensatz zu Protokollen wie NFS dürfen diese auf dem Server nicht gemountet sein und können auch nicht von mehreren Clients gleichzeitig verwendet werden.

Die gesamte Dateisystemlogik wird vom Client bearbeitet, über das Netz werden nur die reinen I/O-Anfragen an die Hardware übermittelt.

Storage Virtualisation Layer - Gvirstor

gvirstor ist eine GEOM-Klasse, die unabhängig von den darunterliegenden Geräten oder deren Gebrauch ein virtuelles device definiert. Dies ist bei Serverkonsolidierungen nützlich. Man kann beliebig große virtuelle Geräte definieren (im Terabyte-Bereich z.B.) und mit beliebig vielen und grossen physischen Datenträgern (jeweils Festplatten und/oder GEOM-devices inklusive RAID-sets im Gigabytebereich) füllen. Der Speicherplatz wird in kleine Stücke aufgeteilt (z.B. zu je 4 MB) und bei Bedarf alloziert. Ist der Speicherplatz erschöpft, können beliebige weitere Festplatten/Geom-devices einfach und flexibel hinzugefügt werden. gvirstor ist ein wichtiger Baustein für einen LVM-Manager und ist ein ladbares Kernelmodul mit einem zugehörigen Utility und später optionaler GUI. Besondere Beachtung erfuhr die leichte Administrierbarkeit und das Monitoring der verfügbaren physischen Datenträger. gvirstor ist in der abschliessenden Testphase und wird bald in FreeBSD integriert sein.

Journaling - gjournal

UFS 2, das Standarddateisystem von FreeBSD, kennt kein Journaling, sondern sog. "Softupdates". Unter Softupdates versteht man eine Erweiterung von UFS, welche die Konsistenz der Metadaten ohne Jornaling mit möglichst geringen Einbussen der Schreibgeschwindigkeit erzielt. Dies wird erreicht, indem ganze Blöcke von zu schreibenden Metadaten, z.B. für ein gesamtes Verzeichnis, im Arbeitsspeicher gehalten, bearbeitet und in einer bestimmten Reihenfolge sortiert auf die Platte geschrieben werden.

Nach einem Crash Neustart (Stromausfall, Systemabsturz) sieht es so aus, als hätte das Schreiben der Metadaten entweder *ganz* oder *gar nicht* stattgefunden. Mit anderen Worten, Meta-Updates sind *atomar* (unteilbar). Das Dateisystem ist immer in einem konsistenten Zustand. Der einzige mögliche Fehler, der dabei auftreten kann, sind Blöcke im Dateisystem, die noch als belegt markiert sind, obwohl sie bereits frei sein sollten. Aus diesem Grund ist eine Prüfung des Dateisystems auf seine Blockbelegung notwendig bei jedem Systemstart. Um die Zeit zum Starten zu Verkürzen, findet diese Prüfung des Dateisystems im Hintergrund statt (bgfsck). Bei sehr großen Plattensystemen kann dies dennoch störend werden.

Daher wurde eine mittlerweile ausgereifte journaling GEOM-Klasse entwickelt, welche voraussichtlich in FreeBSD 6.3 Einzug halten wird (momentan current). gjournal arbeitet unterhalb des Dateisystem-Layers auf einem oder zwei beliebigen Datenträgern, ist dateisystemunabhängig und es ist autonom von einem etwaigen Schreibcache einer Festplatte. Zudem ist es flexibel kombinierbar mit anderen GEOM-Klassen und schneller als Softupdates, vor allem bei kleinen Dateien durch Schreiboptimierung der Metadaten. Wird es z.B. auf gmirror oder graid3 aufgesetzt, entfällt nach einem Crash sogar die Synchronisation, da die Daten konsistent bleiben.

Festplattenkomprimierung: geom_uzip

Das geom_uzip framework erlaubt Ihnen einen lesenden Zugriff auf komprimierte Disk Images. Sie können mit wenig CPU-Bedarf pro Lesevorgang große Mengen an Platz auf ihren Festplatten sparen. geom_uzip durch Daten im Geom-label komprimierte Images die durch mkuzip(8) erzeugt wurden und stellt sie dem Kernel als Ramdisk mit md(4) zur Verfügung. geom_uzip erzeugt ein eindeutiges *md#.uzip* device für jedes Image, welches dann vom FreeBSD Kernel benutzt wird, um auf das Disk Image zuzugreifen. Schreiboperationen sind allerdings nicht möglich.