Das plattformübergreifende Magazin für alle IT-Administratoren

ADMIN

Netzwerk & Securi

FreeBSD

Auf DVD:

Die Linux-Alternative:

SICHER VERBUNDEN

IPSEC UND SSH IN HETEROGENEN NETZEN

LINUX BEQUEM

Was bietet der Univention Corporate Server als Virtualisierungsplattform?

ZFS-Storage

NAS-Eigenbau auf FreeBSD-Fundament

Spurensuche

Netzwerkmitschnitt für forensische Analysen

OpenNMS

Große Netzwerke stets sicher im Blick behalten

So funktionieren SSD-Speicher

Frische Chips

Solid State Drives werden von Tag zu Tag populärer. Die kleinen Performance-Wunder versprechen eine deutlich höhere I/O-Performance als herkömmliche Festplatten – und das bei niedrigerem Stromverbrauch. Wie SSDs funktionieren und wann sich deren Einsatz lohnt, verrät dieser Artikel. Werner Fischer

SSDs haben mit normalen Festplatten nur eine Gemeinsamkeit: Beide Medien speichern Daten. Intern arbeiten SSDs allerdings vollkommen anders als Festplatten. Während Festplatten die Daten auf mehreren Magnetscheiben speichern, nutzen SSDs mehrere Flash-Chips – meist bis zu zehn Stück – für die Datenspeicherung. Da bei SSDs keine mechanischen Teile wie bei Festplatten bewegt werden, sind vor allem zufällig verteilte Datenzugriffe deutlich schneller.

Festplatte im Vergleich

Um die Performance von SSDs mit Festplatten zu vergleichen, lohnt zuvor ein kurzer Blick auf den Aufbau herkömmlicher Festplatten. Diese speichern die Daten auf mehreren rotierenden Magnetscheiben. Die Daten werden dabei mit Schreib-/Leseköpfen auf die Scheiben geschrieben beziehungsweise von diesen gelesen.

Soll nun ein zufälliger Sektor von der Festplatte gelesen werden, muss die Festplatte den Schreib-/Lesekopf an die richtige Stelle bewegen (Seek Time) und dann darauf warten, dass der gewünschte Sektor auf der rotierenden Scheibe unter dem Schreib-/Lesekopf vorbeifährt (Latency Time). Die Latency Time beträgt dabei im Mittel die Dauer, welche für eine halbe Umdrehung notwendig ist. Die Summe aus Seek Time und Latency Time ergibt die Random Access Time (mittlere Zugriffzeit). Diese beträgt je nach Festplatte und deren Drehzahl zwischen fünf und 15 Millisekunden, was 200 respektive 66 IOPS (I/O-Operationen pro Sekunde) entspricht.

SSDs erreichen im Vergleich dazu mehrere tausend zufällig verteilte IOPS. Bei kontinuierlichen Datentransfers kann die Festplatte allerdings deutlich besser mithalten. Sie liefert dabei je nach Typ zwischen 60 und 150 MByte/s. SSDs können zwar zu Beginn über 250 MByte/s liefern. Mit höherem Füllstand der SSD sinkt diese Rate jedoch. Wie sehr die Datenrate zurückgeht, hängt dabei von den Algorithmen im SSD-Controller ab.

Aufbau

Die kleinste Einheit in einem Flash-Chip einer SSD ist die Speicherzelle. Je nach Typ speichert eine einzelne Speicherzelle ein oder mehrere Bits:

- SLC (Single Level Cell) 1 Bit
- MLC (Multi Level Cell) 2 Bits
- TLC (Triple Level Cell) 3 Bits

Eine SLC unterscheidet zwei Ladungszustände, eine MLC vier und eine TLC acht. Die angelegte Spannung beim Schreiben auf eine Speicherzelle ist allerdings fix. Daher muss bei einer MLC für einen Schreibvorgang bis zu vier Mal Spannung angelegt werden, um in den höchsten Ladungszustand zu kommen. Bei einer SLC reichen maximal zwei solcher Vorgänge – die Schreibperformance ist damit höher. TLCs erfordern bis zu acht Mal Spannung und werden daher für SSDs nicht verwendet. Aufgrund ihrer hohen Datendichte und höheren Kapazität werden TLCs aber für USB-Sticks und SD-Karten genutzt. Dazu reicht ihre Performance aus.

Bei jedem Anlegen von Spannung kommt es zu einer kleinen Abnützung der Isolationsschicht der Speicherzelle. Da für einen einzelnen Schreibvorgang auf eine MLC öfter Spannung angelegt wird als bei einer SLC, verträgt die MLC weniger Schreibvorgänge (sogenannte Program/Erase Cycles oder p/e-Zyklen). MLCs haben meist eine Lebensdauer zwischen 10 000 und 30 000 p/e-Zyklen, SLCs rund 100 000 p/e-Zyklen.

Mehrere Speicherzellen bilden eine Page. Sie ist die kleinste Struktur, die vom SSD-Controller gelesen oder beschrie-

ben werden kann. Allerdings kann der SSD-Controller den Inhalt einer bereits beschriebenen Page nicht löschen oder verändern. Eine Page ist in der Regel vier Kibibyte (KiB) groß (= 4096 Bytes). Mit MLCs entspricht das 16 384 Speicherzellen. Künftige Flash Chips mit einer Fertigungsstrukturbreite von 25 nm haben Pages mit acht KiB Größe.

Mehrere Pages sind in Blöcken zusammengefasst. Aktuell besteht ein solcher Block aus 128 Pages und fasst damit 512 KiB an Daten. Blöcke der künftigen 25nm-Fertigung nutzen 256 Pages à acht KiB, in Summe also 2 MiB. Ein Block ist die kleinste Einheit, die der SSD-Controller löschen kann, indem er eine Löschspannung am gesamten Block anlegt. Erst nach dem Löschen eines Blocks kann der SSD-Controller die einzelnen Pages wieder neu beschreiben.

1024 Blöcke bilden eine Plane. Vier Planes sind wiederum auf einem Die untergebracht, wie Abbildung 1 zeigt. Ein Die ist dabei etwa so groß wie ein Fingernagel (167 mm^2). In der Produktion werden circa 200 bis 300 Dies aus einem Wafer gewonnen. Je nach gewünschter Kapazität landen ein bis acht Dies in einem Thin small-outline package (TSOP) – jene schwarzen Chips, wie sie auch auf Speichermodulen vorhanden sind. Eine SSD besteht schließlich aus bis zu zehn TSOPs und einem SSD-Controller.

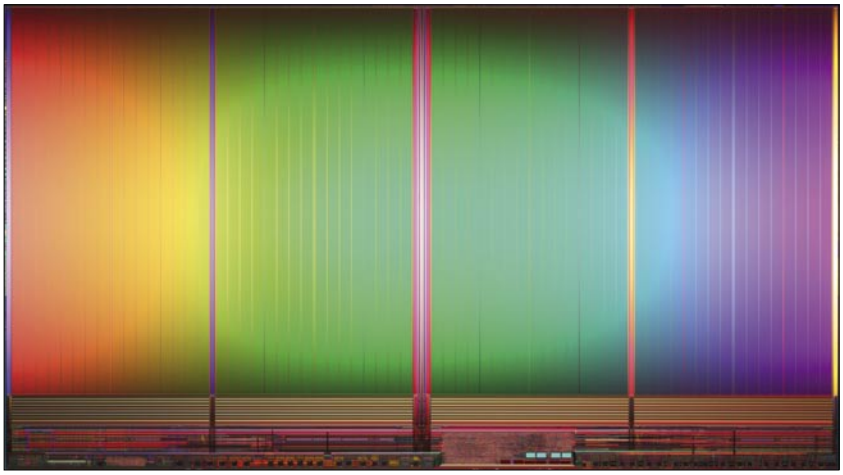

Abbildung 1: Die eines Flash-Chips mit vier Planes. (Quelle: Intel)

Schreibtechniken

Die geschilderte interne Funktionsweise einer SSD erfordert spezielle Schreibtechniken um sowohl eine hohe Performance als auch eine lange Lebensdauer der SSD zu erreichen. Das erstmalige Beschreiben einer neuen SSD ist dabei noch einfach. Alle Blöcke (und somit auch alle Pages) sind gelöscht. Neue Daten schreibt der SSD-Controller direkt auf die entsprechenden Pages.

Schwieriger wird es, sobald bereits vorhandene Daten verändert werden. Ohne zusätzliche Mechanismen müsste der SSD-Controller sämtliche Pages eines Blocks zuerst in einen Cache einlesen, anschließend den gesamten Block löschen, um abschließend die zwischengespeicherten Pages versehen mit den Änderungen wieder in den Block zu schreiben. Die Änderung eines einzelnen Bits würde so das Lesen, Löschen und Neuschreiben von 512 KiB an Daten verursachen.

Das vermeidet der SSD-Controller mit einem einfachen Trick. Wenn er Daten einer Page verändert, schreibt er dazu einfach die neuen Daten in eine andere Page, die noch gelöscht ist. In einer internen Zuordnungstabelle vermerkt der Controller dabei, dass die Daten der logischen LBA-Adresse A nun nicht mehr in Page X, sondern in Page Y zu finden sind. Die ursprüngliche Page X markiert er als ungültig. Damit diese Methode funktioniert, benötigt der SSD-Controller zusätzliche Speicherkapazität. Ansonsten könnte er auf einer vollgeschriebenen SSD keine Daten mehr verändern. Daher hat jede SSD eine sogenannte Spare Area. Typischerweise beträgt die Größe dieser Spare Area zwischen sieben und 27 Prozent der Nennkapazität einer SSD. Eine 160 GB SSD kann etwa tatsächlich über 172 GByte verfügen, 12 GByte versteckt die SSD aber vor dem Betriebssystem und nutzt diese Datenmenge als Spare Area.

Mit fortschreitender Nutzungsdauer verändern sich immer mehr vorhandene Daten. Mehr und mehr Pages markiert der SSD-Controller als ungültig. Einzelne Blöcke enthalten nur mehr 40 bis 60 Prozent echte Daten, die restlichen Pages darin sind ungültig und können vorerst nicht weiter genutzt werden. Bevor nun die Spare Area zur Neige geht, räumt der SSD-Controller auf. Bei dieser Garbage Collection kopiert der SSD-Controller die noch gültigen Pages eines Blocks in einen freien Block, der damit nur teilweise gefüllt ist. Die restlichen Pages dieses neuen Blocks bleiben noch unbeschrieben und können für weitere Schreiboperationen genützt werden. Den ursprünglichen Block löscht der SSD-Controller danach. Somit sind alle Pages dieses gelöschten Blocks wieder beschreibbar. Abbildung 2 zeigt ein solches Beispiel.

Wie eingangs erwähnt, ist die Anzahl der möglichen Schreibvorgänge (p/e-Zyklen) pro Speicherzelle begrenzt. Sollen nun neue Daten gespeichert werden, schreibt der SSD-Controller daher immer zuerst auf Pages, die noch wenig abgenützt sind. Dieser Vorgang wird als Dynamic Wear Leveling (dynamische Abnutzungsverteilung) bezeichnet. Dynamisch deshalb, weil nur neue oder veränderte Daten verteilt werden.

Diese Methode erhöht die Lebensdauer der SSD. Daten, die nur einmal geschrieben werden und sich danach nicht mehr ändern, bleiben damit aber auf ihren Pages. Static Wear Leveling geht daher einen Schritt weiter und verschiebt auch solche Daten periodisch auf andere Pages, die schon mehr abgenützt sind. Damit werden tatsächlich alle Pages einer SSD gleichmäßig abgenützt und die Lebensdauer der SSD steigt weiter.

Write Amplification

Durch die geschilderten Algorithmen werden einmal geschriebene Daten, selbst wenn sie nicht verändert werden, in andere Pages kopiert (etwa durch Garbage Collection oder Static Wear Leveling). Ein geschriebenes Byte kann also mit der Zeit vom SSD-Controller durchaus mehrfach kopiert werden und somit zu

mehreren Schreibvorgängen führen. Der Faktor, wie oft es statistisch zu einem solchen Kopiervorgang kommt, wird als Write Amplification bezeichnet.

Eine hohe Write Amplification führt naturgemäß zu einer höheren Abnutzung und damit geringeren Lebensdauer. Manche SSD-Controller (etwa von Sandforce) versuchen, durch eine Komprimierung der Daten im SSD-Controller die nötigen Schreibvorgänge zu reduzieren, um damit die Abnutzung zu reduzieren. Eine weitere Möglichkeit ist die Vergrößerung der Spare Area. Dadurch sinkt zwar die nutzbare Kapazität, da aber seltener eine Garbage Collection erforderlich ist, steigt dennoch die Lebensdauer der SSD.

Lebensdauer

Die Lebensdauer von SSDs ist durch die Lebensdauer der Speicherzellen begrenzt. Mit fortschreitender Nutzung können vermehrt Blöcke ausfallen. Wobei ein „Ausfall" dabei relativ ist. Je mehr p/e-Zyklen stattfinden, umso länger dauert das Löschen eines Blocks. Überschreitet diese Zeit zum Löschen einen bestimmten Schwellwert, markiert der SSD-Controller diesen Block als „Bad Block" und nutzt stattdessen einen Spare Block aus der Spare Area. Es kommt dabei also zu keinem Datenverlust. Es sinkt nur die Anzahl der Spare-Blöcke. Einzelne Bitfehler können zwar auftreten (raw bit error rate, RBER), werden aber durch ECC-Mechanismen korrigiert. Erst bei zu vielen Bitfehlern greift ECC nicht mehr, und es kommt zu einem unkorrigierbaren Fehler (uncorrectable bit error rate, UBER).

Damit künftig die Lebensdauer von unterschiedlichen SSDs einfach vergleichbar sind, hat die JDEC Solid State Technology Association die beiden Standards JESD218 (SSD Requirements and Endurance Test Method) und JESD219 (SSD Endurance Workloads) verabschiedet. Damit können Hersteller die Lebensdauer ihrer SSDs in TBW (Terabytes written, geschriebene Terabytes) angeben. Einige SSDs geben auch schon jetzt per SMART Auskunft über die verbleibende Lebensdauer (Media Wearout Indicator) [1]. Generell ist die zu erwartende Lebensdauer von SSDs bei normalen Schreibmengen durchaus mit jenen von Festplatten vergleichbar, oft sogar höher, da bei es bei SSDs zu keinen mechanischen Ausfällen (wie etwa Headcrashes bei Festplatten) kommt.

Einsatzgebiete

SSDs eignen sich aufgrund ihrer Eigenschaften für zahlreiche Einsatzgebiete. Die folgende Auflistung ist nach aufsteigenden Investitionskosten gegliedert.

Bereits eine SSD mit geringerer Kapazität zwischen 40 und 80 GByte bringt bei einem Einzelplatz-PC deutliche Produktivitätsvorteile. Bei geringen Investitionskosten dient sie als Installationsmedium für das Betriebssystem und die Anwendungsprogramme neben einer normalen Festplatte, die Benutzerdaten enthält. Das Hochfahren des Rechners geht damit deutlich schneller. Auch die einzelnen Programme starten rascher, das Arbeiten am PC ist flüssiger.

SSDs mit höheren Kapazitäten ab 160 GByte können die bisherige Festplatte in einem Rechner vollständig ersetzen. Das erhöht zwar die Kosten, beschleunigt aber dafür auch den Zugriff auf die Daten des Benutzers. Weitere Vorteile durch den Wegfall einer herkömmlichen Festplatte sind der Wegfall des Betriebsgeräusches der Festplatte und der deutlich geringere Stromverbrauch. Bei Notebooks lässt sich damit die Akkulaufzeit um 20 bis 30 Minuten erhöhen. Auch im Serverumfeld lassen sich SSDs sinnvoll einsetzen. Vor allem bei Datenbanken mit vielen zufälligen I/O-Zugriffen spielen SSDs ihre Vorteile aus. Eine Spiegelung von SSDs mit einem RAID 1 schützt vor Datenverlust beim Ausfall einer einzelnen SSD. Manche RAID-Controller unterstützen dabei sogar ein RAID 1 bestehend aus einer SSD und einer herkömmlichen Festplatte, wobei der RAID-Controller dabei Lesezugriffe im Normalbetrieb ausschließlich an die SSD schickt.

SSDs können darüber hinaus auch als beschleunigender Lesecache in einem RAID-Verbund mit normalen Festplatten genutzt werden. Adaptec-Controller mit Maxcache-Unterstützung legen häufig gelesene Datenbereiche zusätzlich auf SSD ab [2]. Um diese Daten später erneut zu lesen, wird direkt von den SSD gelesen und nicht von den RAID-Festplatten. Im Gegensatz zum normalen Cache des RAID-Controllers hat dieser SSD-Cache eine deutlich höhere Kapazität und bleibt auch bei einem Reboot erhalten.

Fazit

SSDs sind zwar nach wie vor mit höheren Einstiegskosten verbunden. Sie zahlen sich aber durch ihre hohe I/O Performance oft sehr schnell aus. Entscheidend für eine gute Performance ist aber die Qualität, wie gut der Hersteller die Algorithmen im SSD-Controller implementiert hat.

Im nächsten ADMIN zeigt der zweite Teil dieser Artikelserie, wie sich die SSD-Performance mit AHCI, ATA TRIM, einer vergrößerten Spare Area und durch ein korrektes Partition Alignment optimieren lässt. *(ofr)* ■

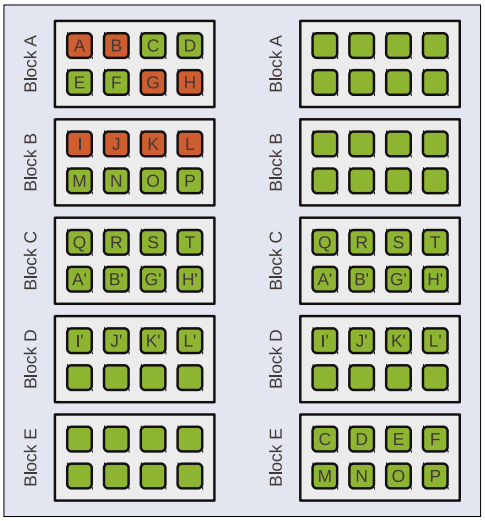

Abbildung 2: Viele Pages sind ungültig, weil deren Daten verändert und woanders abgespeichert wurden. Die Garbage Collection macht die ungültigen Bereiche wieder nutzbar.

Infos:

[1] SSD-SMART-Analyse: [http://www.thomas-krenn.com/SSD-SMART-Analyse]

[2] Maxcache-Support: [http://www.thomas-krenn.com/maxCache]

Der Autor

Werner Fischer ist seit 2005 Technology Specialist bei der Thomas-Krenn.AG und Chefredakteur des Thomas Krenn Wikis. Seine Arbeitsschwerpunkte liegen in den Bereichen Hardware-Monitoring, Virtualisierung, I/O Performance und Hochverfügbarkeit.

SSD-Performance optimieren

Schnelligkeit ist keine Hexerei

Nachdem ein Artikel im letzten ADMIN-Magazin erläutert hat, wie SSDs funktionieren und wann sich deren Einsatz lohnt, geht es diesmal um die Optimierung der SSD-Performance. Werner Fischer

SSDs werden so wie viele Festplatten per SATA-Schnittstelle mit dem Rechner verbunden. Sofern der Rechner halbwegs aktuell ist, erkennt er die SSD beim Hochfahren und kann sie sofort verwenden. Wer die optimale Performance erreichen will, für den lohnt es sich allerdings, vor dem Einsatz ein paar Einstellungen zu überprüfen.

AHCI aktivieren

Der erste Schritt auf dem Weg zu einer optimalen I/O-Performance der SSD besteht darin, den Advanced Host Controller Interface-Modus (AHCI) im BIOS zu aktivieren. Er ermöglicht im Gegensatz zum IDE-Modus Native Command Queuing (NCQ). Damit bekommt die SSD immer gleich mehrere I/O-Anfragen parallel vom Betriebssystem und muss nicht nach jeder einzelnen Abfrage auf die nächste warten. Die Pipeline der SSD bleibt damit voll, der SSD-Controller kann durchgängig I/O-Anfragen abarbeiten, die Performance steigt.

Neben NCQ bietet der AHCI-Modus darüber hinaus auch noch das Device Initiated Power Management (DIPM) der SATA-Schnittstelle, das den Stromverbrauch von SSDs im Idle-Betrieb deutlich minimiert. Das erhöht zwar nicht die Performance, steigert aber etwa die Akkulaufzeit bei einem Laptop. Sowohl der Microsoft MSAHCI-Treiber als auch Linux ab Kernel 2.6.24 unterstützen DIPM. Der MSAHCI-Treiber nutzt DIPM standardmäßig nur im Power Saver Modus. Mit dem Windows Tool »powercfg« lässt sich die Nutzung von DIPM auch in anderen Modi aktivieren. Unter Linux kann DIPM über das Sysfs aktiviert werden (Listing 1). Als dritten Vorteil ermöglicht AHCI, Laufwerke im Betrieb per Hot-Plug zu tauschen.

Spare Area vergrößern

Wie bereits im ersten Artikel dieser Serie erwähnt, kann ein SSD Controller die Daten von einer einmalig beschriebenen Page (die aus mehreren Speicherzellen besteht) nicht verändern. Vor einem erneuten Beschreiben dieser Page müsste der Controller den gesamten Block löschen, in dem sich die Page befindet. Um dies zu vermeiden, schreibt der Controller in einem solchen Fall die geänderten Daten einfach in eine andere, bisher unbenutzte Page und aktualisiert seine interne Zuordnungstabelle entsprechend (Abbildung 1).

Sind auf der SSD zu Beginn nur wenige Daten gespeichert, gibt es noch ausreichend freie Pages. Darüber hinaus hat jede SSD eine Spare Area, deren Pages der SSD Controller ebenfalls für diese Zwecke nutzt. Bevor die freien Pages ganz zur Neige gehen, räumt der SSD Controller mit seinem Garbage Collector auf. Er kopiert dabei verstreute belegte Pages aus verschiedenen Blöcken in noch freie Pages. Die dadurch freigeschaufelten Blöcke kann er nun löschen. Damit stehen ihm in der Summe wieder mehr unbenutzte Pages zur Verfügung. Diese Garbage Collection kostet jedoch Zeit und damit Performance. Außerdem erhöht sie durch das interne Kopieren die Anzahl der Schreibvorgänge auf die einzelnen Speicherzellen (die Write Amplification steigt). Somit sinkt auch die Lebensdauer dieser Speicherzellen. Ein einfacher Trick verhindert häufige Auf-

Listing 1: DIPM-Konfiguration unter Linux

```
01 root@ubuntu-10-10:~# hdparm -I /dev/sda | grep Device-initiated
02                Device-initiated interface power management
03 root@ubuntu-10-10:~# echo min_power > /sys/class/scsi_host/host0/link_power_management_policy
04 root@ubuntu-10-10:~# hdparm -I /dev/sda | grep Device-initiated
05           *    Device-initiated interface power management
```

rufe der Garbage Collection: Werden bei der erstmaligen Nutzung der SSD etwa nur 90 Prozent des verfügbaren Datenbereiches partitioniert (Over-Provisioning), erfolgen auf die restlichen zehn Prozent niemals Schreibzugriffe. Die entsprechenden Pages bleiben ungenutzt, der SSD Controller kann diese Pages somit wie Pages aus der Spare Area nutzen. Untersuchungen von Intel zeigen, dass bei einer solchen Einsparung von zehn Prozent der Datenmenge die Random-I/O-Performance auf das Zweieinhalbfache steigt und sich die Lebensdauer der SSD mehr als verdoppelt (Abbildung 2).

Secure Erase

Wie bereits erläutert kann ein SSD Controller die Daten von einer einmalig beschriebenen Page nicht verändern, sondern muss zuvor den gesamten Block mit mehreren Pages löschen. Für die Wiederverwendung einer zuvor bereits benutzten SSD ist es daher sinnvoll, vor dem Einsatz alle Blöcke der SSD zu löschen. Bei den meisten SSDs klappt dies mit einem einfachen Secure Erase [2]. Ein Secure Erase soll laut ATA-Spezifikation das sichere Löschen aller gespeicherten Daten eines Datenträgers garantieren. Bei den meisten SSDs, die Secure Erase unterstützen, führt dies zum physischen Löschen aller Blöcke der SSD. Die SSD ist dann wieder mit der ursprünglichen optimalen Performance nutzbar, da alle Pages direkt beschrieben werden können. Bei einigen neueren SSDs ist das Secure Erase allerdings anders implementiert. Diese SSDs verschlüsseln automatisch alle geschriebenen Daten. Bei einem Secure Erase wird dann einfach der Schlüssel sicher gelöscht – die Daten können damit nicht mehr entschlüsselt werden, sind aber noch physisch vorhanden. Bei solchen SSDs werden somit nicht alle Blöcke der SSD gelöscht. In diesem Fall müssen die Blöcke per TRIM gelöscht werden, um für die neue Verwendung der SSD die optimale Performance zu bekommen. Windows 7 führt ein solches TRIM bei der Formatierung automatisch durch. Unter Linux bietet »hdparm« dazu zwar die »--trim-sector-ranges«-Option, die Manpage von hdparm 9.37 rät allerdings nach wie vor von ihrer Verwendung ab.

Partition Alignment

Unter Partition Alignment versteht man das Ausrichten von Partitionen an bestimmten Grenzen eines Datenträgers [3]. Ein korrektes Partition Alignment gewährleistet eine optimale Performance bei Datenzugriffen. Speziell bei SSDs (mit internen Page-Größen von beispielsweise 4.096 oder 8.192 Bytes), Festplatten mit 4 KiB-Sektoren (4.096 Bytes) und RAID-

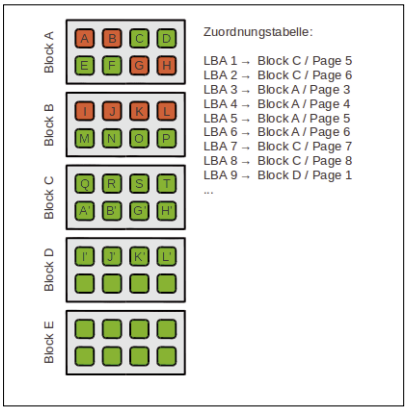

Abbildung 1: Viele Pages (rot markiert) sind ungültig, weil deren Daten verändert und woanders abgespeichert wurden. Der SSD Controller aktualisiert seine interne Zuordnungstabelle entsprechend.

Volumes führt eine fehlerhafte Ausrichtung von Partitionen zu einer verminderten Performance [4].

In der Vergangenheit begann die erste Partition stets auf LBA-Adresse 63 (entspricht dem 64. Sektor), um kompatibel zu DOS und zur alten CHS-Adressierung (Cylinder/Head/Sector) zu bleiben. Die Größe eines solchen (logischen) Sektors beträgt 512 Byte. Bei normalen Festplatten (mit einer physischen Sektorgröße von 512 Byte) bringt das keine Nachteile. Neuere Festplatten mit einer physischen Sektorgröße von 4.096 Byte (4 KiB) emulieren zwar nach außen hin eine Sektorgröße von 512 Byte, arbeiten intern aber mit 4.096 Byte. Und auch SSDs arbeiten mit einer Pagegröße von 4 KiB beziehungsweise 8 KiB. Bei diesen neuen Festplatten und SSDs ist eine solche Partitionierung beginnend bei LBA-Adresse 63 daher sehr problematisch.

Formatiert der Benutzer eine solche Partition mit einem Dateisystem mit einer

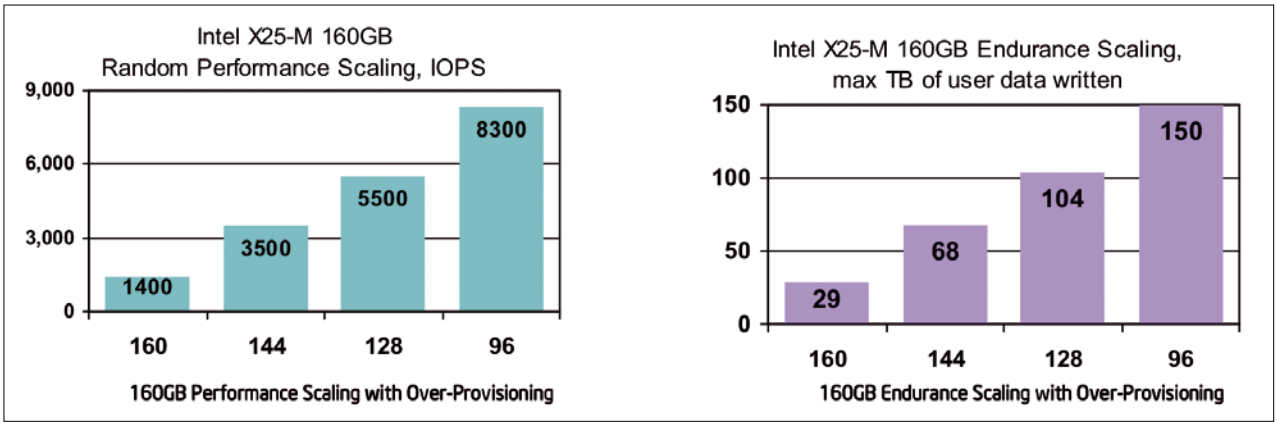

Abbildung 2: Eine etwas vergrößerte Spare Area hat positive Auswirkungen. Nutzt man nur 144GB einer 160GB X25-M SSD, steigen I/O Performance und Lebensdauer.

typischen Blockgröße von 4 KiB, passen die 4-KiB-Dateisystem-Blöcke nicht direkt in die 4 KiB oder 8 KiB großen Pages der SSD (Abbildung 3). Beim Schreiben eines einzelnen 4-KiB-Dateisystem-Blockes müssen dann zwei 4 KiB Pages verändert werden. Erschwerend kommt dabei hinzu, dass die jeweiligen 512-Byte-Sektoren erhalten bleiben müssen – es kommt damit zu einem Read/Modify/Write. Die Folge ist eine bis zu 25fach schlechtere Schreibperformance bei kleinen Dateizugriffen, wie Analysen von IBM zeigen [5].

Um diese Probleme zu vermeiden, empfiehlt sich ein Alignment auf 1 MiB – damit ist man auf lange Sicht auf der sicheren Seite. Mit der aktuellen Adressierung in 512 Byte großen logischen Sektoren entspricht das 2048 Sektoren (Abbildung 4). Neuere Windows-Versionen (Windows Vista, Windows 7, Windows Server 2008) führen bei Partitionen größer 4 GiB ein solches Alignment auf 1 MiB durch. Kleinere Partitionen richten diese Windows-Versionen auf 64 KiB aus. Ältere Versionen (Windows XP, Windows Server 2003) benötigen ein manuelles Alignment. Aktuelle Linux-Distributionen verwenden ebenfalls ein Alignment von 1 MiB bei der Installation. Beim späteren Einrichten von Partitionen mit »fdisk« sind dessen Optionen »c« (Deaktiveren des DOS-Kompatibilitätsmodus) und »u« (verwendet Sektoren statt Zylinder als Einheiten) nötig, um ein korrektes Alignment zu erhalten.

Zur Kontrolle zeigt »fdisk -l -u« die Startsektoren aller Partitionen. Sind diese Sektoren durch 2048 teilbar, sind alle Partitionen korrekt ausgerichtet (Listing 2). Neue LVM-Versionen (ab Version 2.02.73) verwenden dank eines Patches von Mike Snitzer ebenfalls ein Alignment von 1 MiB [6].

Listing 2: Partition-Alignment mit »fdisk«

```
01 root@ubuntu-10-10:~# fdisk -l -u /dev/sda
02
03 Disk /dev/sda: 160.0 GB, 160041885696 bytes
04 255 heads, 63 sectors/track, 19457 cylinders, total 312581808 sectors
05 Units = sectors of 1 * 512 = 512 bytes
06 Sector size (logical/physical): 512 bytes / 512 bytes
07 I/O size (minimum/optimal): 512 bytes / 512 bytes
08 Disk identifier: 0x9349dd6c
09
10    Device Boot      Start         End      Blocks   Id  System
11 /dev/sda1   *        2048     2457599     1227776    7  HPFS/NTFS
12 Partition 1 does not end on cylinder boundary.
13 /dev/sda2         2457600    61051349    29296875    7  HPFS/NTFS
14 Partition 2 does not end on cylinder boundary.
15 /dev/sda3        61052928    80582655     9764864   83  Linux
16 Partition 3 does not end on cylinder boundary.
17 /dev/sda4        80582656   312580095   115998720   83  Linux
18 Partition 4 does not end on cylinder boundary.
```

ATA TRIM

Eine weitere Funktion zur Steigerung der Performance sowie der Lebensdauer ist ATA TRIM. Mit dem TRIM-Kommando teilt das Betriebssystem der SSD mit, welche Datenbereiche (etwa die Datenbereiche einer gelöschten Datei) es nicht mehr benötigt. Der SSD-Controller kann damit betroffene Blöcke löschen, was ähnlich wie eine vergrößerte Spare Area die Performance und Haltbarkeit der SSD steigern soll. Damit ATA TRIM funktioniert, muss es von der SSD, vom Betriebssystem und vom Dateisystem unterstützt werden. Windows 7 mit NTFS oder Linux ab Kernel 2.6.33 mit Ext4 mit Discard-Option erfüllen diese Anforderungen.

Ab Kernel 2.6.38 unterstützen Ext4 und XFS zudem das zeitversetzte Batched Discard, das bei SSDs mit langsamer TRIM-Funktion wichtig ist. Über den tatsächlichen Performance-Gewinn aufgrund von ATA TRIM gibt es unterschiedliche Aussagen.

Tatsache ist, dass ATA TRIM nur für einzelne SSDs genutzt werden kann. RAID-Controller unterstützen diese Funktion nicht. Eine etwas vergrößerte Spare-Area durch Nutzung von 90 Prozent der SSD-Kapazität für das RAID-Volume sollte in diesem Fall ein Fehlen der TRIM-Funktion ausgleichen. *(jcb)* ■

Infos:

[1] Over-provisioning an Intel SSD : [http://cache-www.intel.com/cd/00/00/45/95/459555_459555.pdf]

[2] Secure Erase: [http://www.thomas-krenn.com/Secure-Erase]

[3] Partition Alignment: [http://www.thomas-krenn.com/Alignment]

[4] Ben Martin, RAID-Systeme unter Linux optimal konfigurieren, ADMIN 02/2011, S. 80

[5] Linux on 4KB-sector disks: [http://www.ibm.com/developerworks/linux/library/l-4kb-sector-disks]

[6] LVM Alignment Patch: [http://www.redhat.com/archives/lvm-devel/2010-August/msg00035.html]

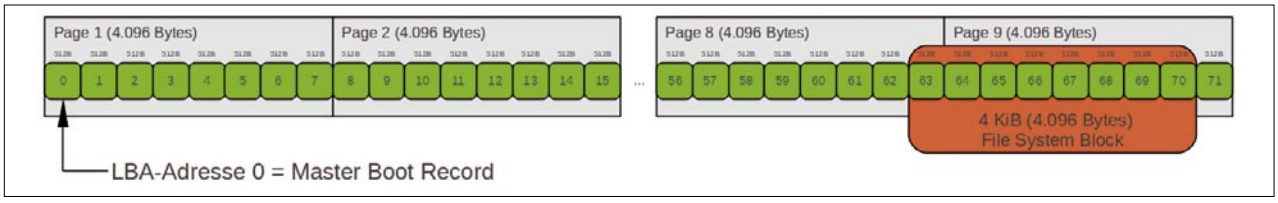

Abbildung 3: DOS-kompatible Partitionen, die bei LBA Adresse 63 beginnen, führen zu erheblichen Performance-Nachteilen bei Datenzugriffen.

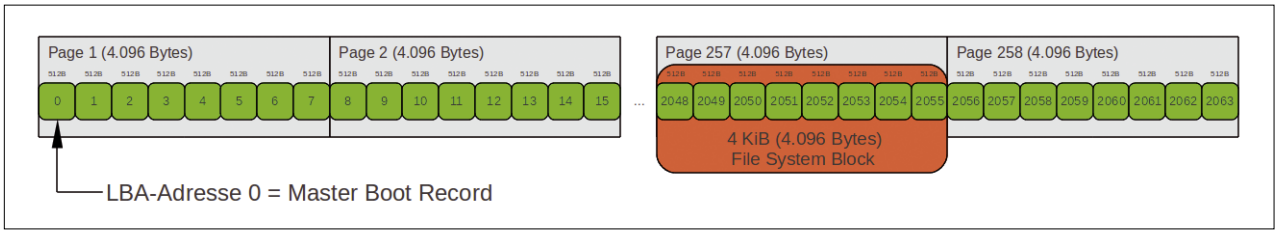

Abbildung 4: Eine korrekt ausgerichtete Partition bringt optimale Performance bei Lese- und Schreiboperationen.

IN KÜRZE VERFÜGBAR!

DIE NEUE INTEL® SOLID-STATE DRIVE 320 / 510 SERIES

JETZT WERDEN THOMAS KRENN SERVER NOCH SCHNELLER!

Die Highlights:

- Bis zu 600GB Speicherkapazität
- Geringerer Stromverbrauch als HDDs
- Bis zu 300x mehr Random-IOPS als HDDs
- Flash-Chips mit 25nm Technologie (320 Series)
- Erweiterte Power-Loss Data Protection
- Random-IO Lese-Performance bis zu 39.500 IOPS (320 Series)

Profitipps:

Alles über SSDs im Thomas Krenn Wiki
http://www.thomas-krenn.com/Intel-SSDs

Thomas-Krenn.AG®
Die Server-Experten

TK SERVER

Flächendeckendes Händler- und Servicenetz in der Schweiz:
www.thomas-krenn.com/ch

www.thomas-krenn.com EU: +49 (0) 8551 9150-0 · AT: +43 (0) 7282 20797-3600 · CH: +41 (0) 848 207970 **Made in Germany!**