

LSB - der Standard für den Pinguin

[Jana Jaeger](#)

Die Welt hat einen weiteren Standard mehr - braucht sie ihn? Das werden sich vielleicht seit dem 29. Juni 2001 viele fragen. Standards sind in der Unixwelt nichts sonderlich Ungewöhnliches und es gibt sie gleich bündelweise. Nicht immer haben solche Standards das hehre Ziel (die Welt einfacher zu machen) erreicht und trugen durch Unsicherheiten in ihrer Umsetzung eher dazu bei, die Programmiererwelt zu verwirren. In diesem Zusammenhang existiert ein wunderschönes, wenn auch sehr zynisches Zitat von Andrew S. Tanenbaum, dem Vater von MINIX:

"The nice thing about standards is that there are so many of them to chose from"
Das Schöne an Standards ist, daß es so viele davon zum Aussuchen gibt.

Dieser Artikel möchte das Geheimnis um den Sinn, Zweck und Inhalt des LSB ein wenig lüften und belegen, daß LSB nicht einer unter vielen Standards sein wird, die einfach in der Versenkung verschwinden.

Einer für alle - alle für einen

Ein paar Worte allgemein zum LSB (Linux Standard Base), bevor es an die Einzelheiten geht. Grob gesagt, losgelöst von allen technischen Einzelheiten, will LSB die gemeinsame Basis bilden, an der sich Distributoren, Entwickler und Firmen, die auf Linux aufsetzende Software entwickeln (engl. *Independent Software Vendors, ISV*), orientieren können. Warum dies so wichtig ist und welche Seite wie vom diesem Standardwerk profitiert, dazu später.

Mittlerweile ist Linux schon länger nicht mehr dieses "X-Disketten & 1 Nerd"-Phänomen, das es früher einmal war. Immer mehr Softwarefirmen oder einzelne Entwickler haben Blut geleckt und wollen Software für Linux entwickeln (und auch verkaufen).

Damit tut sich aber schon sehr schnell ein Problem auf: Sobald die eigenen Programme jetzt aber mit einem Linux (will sagen: einer Distribution) laufen sollen, muß man entweder sich auf ein System festlegen oder für seine Software für jedes Linux einzeln kompilieren. Unterschiedliche Linuxanbieter verwenden unterschiedliche Versionen von Systembibliotheken (glibc, z.B.), die unterschiedliche Versionen oder Implementierungen mancher Subroutinen verwenden, und verstecken ihre Dateien, Programme und Initialisierungsskripte teilweise an unterschiedlichen Orten.

Die logische Folge davon ist, daß viele der ISVs aus Effizienzgründen ihre Software nur für die paar wenigen großen Distributionen oder nur eine einzelne davon liefern. Auf lange Sicht würde ein solches Verhalten die bunte Welt der Linuxdistributionen relativ schnell ausdünnen, weil es für manche dann eben nicht heißt: "Firma XYZ hat ihre Software für unser Linux zertifiziert". Früher oder später würden die Geldsummen, um die es bei solchen Aktionen mittlerweile geht, den einen oder anderen ruinieren.

Eines der Ziele von LSB ist mit, diese Gebundenheit an einige wenige Distributionen aufzuheben und es den ISVs zu ermöglichen, LSB-konforme Software zu vertreiben, die auf allen LSB-konformen Systemen problemlos läuft.

Von einer solchen Regelung der Verhältnisse würde jeder profitieren, der auch nur im entferntesten mit Linux bzw. Software dafür zu tun hat:

- **der Benutzer:** es spielt keine Rolle mehr, welches Linux/welche Distribution er verwendet. Er kann zwischen ihnen Software austauschen/dazuinstallieren, wie immer er möchte. LSB-konforme Pakete kennen keine Abhängigkeiten mehr, als die von anderen LSB-Paketen.
- **der ISV:** Software, die für ein LSB-konformes Linux zertifiziert wurde, wird ohne jegliches Problem auch auf anderen LSB-konformen Linuxen einsetzbar sein. Der ISV ist in keiner Weise an die Unterstützung einer bestimmten Distribution gebunden.
- **der Entwickler:** es reicht, Software so zu schreiben, daß sie den Anforderungen des LSB genügt, Anpassungen für einzelne Distributionen werden hinfällig, so diese den LSB unterstützen.

Die LSB-Initiative begann Anfang 1998 mit der ursprünglichen Idee von namentlich Bruce Perens und anderen, eine offene und nichtkommerzielle Referenzdistribution zu schaffen. Die Vielzahl der auf dem Markt befindlichen Distributionen ließ die Gefahr einer Aufspaltung in viele inkompatible Zweige real werden. Auf lange Sicht würde sich die Linuxdistributionen mit einer solchen Politik selbst den Nachschub abschneiden, weil die Entwicklungsgemeinde sowohl auf kommerzieller Seite (ISV) als auch auf nichtkommerzieller Seite die Orientierung verlieren würde, wie sie ihre Anwendungen denn nun zu entwickeln haben. Mit dem "papiernen" Standard will das LSB-Team eine Orientierungshilfe geben, die Referenzimplementierung sollte von der Distributorenseite leicht in ihre eigentliche Distribution zu integrieren sein und den Entwicklern eine lauffähige Referenzplattform bieten. Mittels spezieller Testprogramme, den "Test Suites", kann die Konformität zum Standard sowohl die Distributionen als auch die Applikationen betreffend überprüft werden. Das erste Announcement des LSB-Projekts vom Mai 1998 faßt seine (ehrgeizigen) Ziele zusammen.

Die geistigen Väter der LSB finden sich in den Reihen hochkarätiger Entwickler und Vordenker der Open Source Community (Linus Torvalds, Jon "Maddog" Hall, Bruce Perens etc.) und einiger Distributoren. Im Laufe der Zeit stießen auch Vertreter großer ISVs (Hewlett Packard, IBM, Oracle, ...) dazu, die das Projekt teils mit Knowhow teils mit finanzieller Unterstützung weitertragen.

Der Blick nach innen

Wie werden jetzt die hehren Ziele, von denen oben die Rede war umgesetzt, welche Elemente muß ein solcher Standard enthalten, wenn er erfolgreich sein will und was ist davon mit LSB 1.0 bereits umgesetzt?

Ein LSB-konformes System muß von der Wiege bis zur Bahre resp. vom Booten bis zu den verwendeten Routinen normiert sein, d.h. langfristig müssen alle der nun folgenden Punkte umgesetzt werden, damit Applikationen auf allen LSB-Linuxen laufen können.

Shared Libraries (glibc, pthreads, ncurses, etc.)

Unterschiedliche Linuxdistributionen verwenden unterschiedliche Systembibliotheken. So kann das Linux von A eine jüngere Version enthalten als B und C wiederum um Meilen von einem gemeinsamen Nenner abweichen. Liefert ein ISV nun eine neue Software aus, kann es sein, daß diese auf A ohne Probleme läuft, auf B aber überhaupt nicht. Das Problem in diesem Fall: Der ISV hat seine Software zur Verwendung mit der Version von A optimiert - leider sind Anwendungen zwar in der Lage, mit aktuelleren Bibliotheken zu arbeiten, aber um die Rückwärtskompatibilität ist es schlecht bestellt.

Der Königsweg: Man definiere eine bestimmte Version als Standard, gebe ihr einen sprechenden Namen, beispielsweise `ld-1sb.so.1` und habe somit einen festen Bezugspunkt für die ISVs.

Denn wenn nun diese Bibliothek in allen LSB-konformen Distributionen enthalten ist, werden alle Applikationen, die diese Standardbibliothek brauchen, auch laufen. Das bedeutet nicht, daß die

Distributionen nicht auch parallel andere Versionen dieser Bibliothek enthalten können, mit denen besondere, für eine Distribution typische Software laufen kann.

Systembefehle

Ein weiterer Knackpunkt für ein Referenzsystem sind die verwendeten Systembefehle. Es muß eine genaue Auflistung aller Befehle sowie ihrer Unterbringung im Dateisystem erstellt werden, da nur dann die Programmierer den letztendlichen Überblick behalten, welche Routine sie überhaupt in ihren Programmen oder Skripten aufrufen dürfen.

LSB definiert daher einen Mindestsatz an allgemeinen UNIX-Befehlen (man, cat, grep, etc.), die ein LSB-konformes System beherrschen muß (hier ein [Überblick](#)). Damit geht der nächste wichtige Punkt einher:

Dateisystem Hierarchie (Filesystem Hierarchy)

Was nützt es dem besten System, wenn es standardgemäß zusammengestellte Bibliotheken und Systembefehle umfaßt, diese aber an allen möglichen Orten in dem Dateisystem verbirgt? Damit alle Programme und Skripte auf einer standardkonformen Distributionen "wissen", an welcher Stelle welche Datei zu finden ist, umfaßt LSB den Filesystem Hierarchy Standard (FHS) in der Version 2.2. Der FHS ist eine Spezifikation, die schon vor Zeiten der LSB für die Organisation der meisten Distributionen verwendet wurde. Die gesamte aktuelle Version liegt unter www.pathname.com, auch erreichbar über www.freestandards.org.

Ablauf bei Initialisierung des Systems/Initskripten

Ein System wird aus Sicht der ISVs nicht erst interessant, wenn es fertig initialisiert ist. Besonders die Hersteller großer Datenbankanwendungen müssen zum Beispiel wissen, welche grundlegenden Dienste (z.B. Netzwerk) ab welchem Zeitpunkt verfügbar sind. Datenbanken greifen teilweise schon auf einer sehr elementaren Ebene des Bootprozesses auf ein System zu. Deshalb setzt sich LSB sehr genau damit auseinander, wie die Initialisierungsdateien/-skripte beschaffen sein müssen, wie sie aufeinander abfolgen und wo im System sie liegen, (siehe oben FHS).

Hierbei ist es wichtig, daß jedes System die Skripte über eine definierte Schnittstelle richtig versteht und umsetzt. Es muß aber nicht notwendigerweise bedeuten, daß der Runlevel 5 des LSB auch der Runlevel 5 einer Distribution ist. Wichtig ist allein, **daß** die vom LSB vorgesehene Funktionalität vorhanden ist, nicht **wie**.

Umsetzen des POSIX.1 Standards (mit glibc Erweiterungen)

Um die Programmierer von Linux Software nicht darüber im Unklaren zu lassen, welche Funktionen anwendbar ist, bedient sich LSB des POSIX (Portable Operating System Interface) Standards, dem allgemein für UNIX und UNIX-artige Systeme wichtigsten Standard.

Sockets

Schon von "Kindesbeinen" an ist Linux als ein sehr netzwerkbewußtes System konzipiert. Die Art, wie das System mit dem Internet und Netzwerken allgemein kommuniziert, ist ein weiterer wichtiger Bereich, den LSB abdeckt. Hierbei bleibt LSB recht nah an den schon existierenden Standards und stärkt ihnen so den Rücken.

X11

Sehr viele, wenn auch lange nicht alle Applikationen brauchen eine grafische Oberfläche. Grafische Anwendungen können durch das statische Linken einiger benötigter Bibliotheken ziemlich große Ausmaße erreichen. LSB standardisiert die niedrigsten Protokollschichten, um

gerade diese Erscheinung ein wenig einzudämmen. Auf höherer Ebene haben die Programmierer alle Freiheiten, zu tun und zu lassen, was sie wollen. So können X11 Anwendungen weiterhin relativ frei auf diversen (Nicht-LSB- oder Nicht-Linux)-Plattformen genutzt werden, die entsprechenden Protokolle vorausgesetzt.

Grundlagen der Softwareinstallation

Ein weiterer Punkt, in dem sich LSB-Systeme untereinander unbedingt einig sein sollten, ist die Verwendung eines einheitlichen Paketformats zur Softwareinstallation. Das letztendliche LSB-Paketformat entspricht (mit einigen Einschränkungen) dem mehr oder weniger allgemein verbreiteten RPM-Format (Definition nach [Maximum RPM](#)). Wie die Pakete erzeugt werden, wird nicht vom LSB spezifiziert, aber es gibt eindeutige Richtlinien zur Namensvergabe und zur Angabe der Abhängigkeiten einzelner Pakete untereinander. Logischerweise dürfen Pakete einer LSB-Distribution nur von anderen LSB-Paketen abhängig sein. Die Verwendung von RPM als Standardpaketformat bedeutet wiederum nicht, daß eine Distribution ihr eigenes Format aufgeben muß, um LSB-konform zu werden. Selbstverständlich kann sie ihren "Hausmechanismus" weiterhin für die Nicht-LSB-Pakete/Nicht-RPM-Pakete verwenden.

ELF

Das "Executable Linkage Format" ist mittlerweile bei Linux Standard und beschreibt, wie Programme und Bibliotheken als Binärdateien aufgebaut sind.

Die bloßen Definitionen verleihen einem Standard allein noch nicht allzu viel Gewicht - wer garantiert denn, daß "überall, wo LSB draufsteht, auch LSB drin ist"? Solange kein Referenzsystem existiert, an dem sich alles messen läßt oder bestimmte Tests, mit denen der Grad der Standardkonformität festgestellt werden kann, ist er das Papier nicht wert, auf das er gedruckt ist.

Aus genau diesem Grund ist ein Referenzsystem nach allen Regeln des LSB bereits in Arbeit. Viel wichtiger ist aber im Moment, daß es bereits einige "Test Suites" gibt, mit deren Hilfe einmal der Anbieter einer Distribution nachprüfen kann, inwieweit sein System LSB-konform ist, und zum anderen ein ISV überprüfen kann, ob seine Applikationen auf einem solchen System laufen. Allerdings ist das ganze Gebiet der Test Suites genauso wie die Referenzimplementierung noch ziemlich "work in progress". Die Anfänge sind gemacht, aber es braucht noch einige Zeit, bis diese Dinge voll umgesetzt, verabschiedet und offiziell freigegeben sind.

Ausblick

Obwohl es zur Freigabe von LSB 1.0 vor zwei Wochen kaum einen großen Medienrummel gegeben hat, kann daraus nicht geschlossen werden, daß dieses Ereignis eine bloße Fußnote im Linuxgeschehen bedeutet. Die große Aufregung um LSB steht bevor, sobald der Rest von LSB auch freigegeben ist und die Zertifizierungen beginnen.

Ein Blick in die Liste der Unterstützer des Projekts (www.linuxbase.org) macht das Gewicht der LSB klar und sollte Zweifel an seiner Zukunft besänftigen können. Das Kunststück, die kommerziellen Interessen der ISVs mit den technischen Interessen der Entwicklergemeinde an einen Tisch zu bringen, wird gelingen, wenn beide Seiten miteinander am Tisch sitzen, wie hier geschieht. Beide Seiten haben ein lebhaftes Interesse daran, die Sache voranzubringen - und im Zweifelsfall sitzt die "Mahnung zur Vernunft" (will sagen: zur Wahrung der gemeinsamen Interessen) auf der gegenüberliegenden Seite des Tisches ;-)

Genausowenig werden sich solche Befürchtungen bewahrheiten, nach denen mit LSB die Vielfalt und Einzigartigkeit der einzelnen Distributionen leiden wird. LSB besagt nicht, daß alle Distributionen gleich zusammengesetzt sein müssen und legt niemanden darauf fest, ausschließlich LSB-konforme Software mit seiner Distribution auszuliefern. Die Unterstützung von LSB wird für viele ein wichtiges zusätzliches Feature darstellen, anstatt die Linuxlandschaft zu einer Monokultur verkommen zu lassen.

Außerdem definiert sich der Erfolg eines Distributors nicht nur über die Softwareauswahl seiner Distribution.

Die verschiedenen LSB-Addons einer Distribution sind:

das LSB-Basissystem

es genügt den Anforderungen des LSB und ermöglicht das Laufenlassen von LSB Applikationen.

die LSB-Pakete

alle Anwendungen die unter einem LSB-System laufen sollen, sind nach den oben beschriebenen Kriterien entwickelt und gepackt worden.

Testsuites

eine Sammlung von Testprogrammen, mit deren Hilfe LSB-Konformität von Basissystem bzw. Paketen/Applikationen festgestellt werden kann.

Entwicklungsumgebung

Zur Entwicklung LSB-konformer Programmpakete stehen den Entwicklern in Zukunft spezielle Werkzeuge zur Verfügung.

Das erste Element muß jede Distribution enthalten, die als LSB-konform eingestuft werden will. Mit den Testsuites und der Entwicklungsumgebung kann der Distributor einen Zusatzservice für Entwickler und ISVs einrichten.

SuSE und LSB

Da zur Zeit noch nicht das ganze Paket des LSB verabschiedet ist, kann keine Distribution LSB 1.0-zertifiziert bzw. -konform sein. SuSE hat bereits mit SuSE Linux 7.1 den FHS weitgehend umgesetzt.

SuSE arbeitet aktiv an der Entwicklung des LSB mit und wird LSB-Unterstützung in die Distribution integrieren und zertifizieren lassen.

Mehr zu LSB

Wer mehr zu LSB erfahren möchte, für den sind in erster Linie die Seiten des LSB Projekts und der Free Standards-Initiative von Interesse.

- Die Homepage des [LSB-Projekts](#) mit vielen weiterführenden Links zu Artikeln und Vorträgen über LSB.
- Die Homepage des [Free Standards-Projekts](#). 