. روش های نصب برنامه ها در لینوکس [۱]

با توجه به گستردگی توزیع های لینوکس مسلما روش های نصب نرم افزار ها نیز متفاوت خواهد بود . در ابتدا به تعریف مفهوم کد منبع یا Source Code می پردازم . یک نرم افزار در ابتدا فقط یک سری حروف و کلمات است که یک برنامه نویس آن ها را در محیطی خاص و یایک ویرایشگر متن معمولی می نویسد این نوشته ها " سورس کد" برنامه یا " کد منبع" برنامه نام دارند. این کد ها را می توانید در یک ویرایشگر متن معمولی وارسی کنید ،اما برای کامپیوتر معنایی ندارند ! چرا که کامپیوتر شما فقط زبان ۰ و ۱ را متوجه می شود.بنابراین باید این برنامه ی نوشته شده را به زبان کامپیوتر ترجمه کنیم ، این کار ترجمه را " کامپایل " می گویند.درواقع شما کدمنبع را کامپایل می کنید تا کامپیوتر شما آن رادرک کند. به فایل حاصل از کامپایل فایل "باینری" می گویند. دلیل این نام گذاری این است که فایل حاصل از کامپایل فقط برمبنای دو (Base2) می باشد ،یعنی فقط شامل صفر و یک است. بنابراین وقتی می گوییم برنامه ای بازمتن (OpenSourec)است این بدان معناست که کد های منبع آن در اختیار دیگران قرار دارد و در برنام های CloseSource یا سورس بسته این کد های منبع در اختیار شما قرار نمی گیرد وفقط فایل های باینری در اختیار شما قرار می گیرد.

یکی از محبوب ترین راه های نصب نرم افزار در لینوکس نصب از کد منبع یا به اصلاح درست تر کامپایل از سورس کد برنامه است.در ویندوز مایکروسافت شما همواره بسته های باینری را در سیستم خودنصب می کردید ،این بسته های باینری که از پیش کامپایل شده اند بر اساس یک استاندارد کلی که سیستم عامل (ویندوز) دارد ، کامپایل شده اند بنابر این شما نمی توانید به راحتی برروی آن ها مانور دهید ،اما در لینوکس قضیه کمی متفاوت است ،شما می توانید کد منبع را بگیرید و از اول روی سیستم خودتان کامپایل کنید .شاید بپرسید این چه مزیتی دارد؟ زمانی که شما برنامه ای را در سیستم خودتان شخصا کامپایل می کنید می توانید آن را بر اساس نیاز های خودتان تنظیم کنید.(در اصطلاح صحیح تر می توانید نرم افزار راباتوجه به شرایط خود بهینه سازی -optimization-کنید.) حسن این کار در آن است که برنامه ی نصب شده در سیستم شما کاملابا سیستم شما خوانایی دارد و اگر شما به امکاناتی که برنامه ارائه می دهد نیازی ندارید می توانید به سادگی آن امکانات را زمان کامپایل ندیده بگیرید.

بنابراین اولین روش نصب برنامه کامپایل از سورس کد بود ،این روش تا حدودی وقت گیر است و نیاز به تجربه دارد تابتوانید به بهترین روش ممکن برنامه ی خود را کامپایل کنید ،البته گاهی راهی جز کامپایل نرم افزار ندارید .برای همین توزیع کنندگان لینوکس به فکر ایجاد روشی کاربر پسندتر افتادند تا کاربران بتوانند در زمانی کوتاه تر و با روشی ساده تر برنامه های خود را نصب کنند ، این دیدگاه باعث بوجود آمدن بسته های نرم افزاری RPM و DEB شد . این بسته ها اغلب همانند بسته های نرم افزاری ویندوز از قبل کامپایل شده اند (البته این فقط یک مقایسه ی ناشیانهات و در عمل با بسته های ویندوزی بسیار متفاوتند) و شما از طریق یک مدیر بسته می توانید این بسته ها را مدیریت کنید. مدیر بسته ها در لینوکس تقریبا همان

برنامه ی Add/Remove در کنترل پانل ویندوز است که با توجه به توزیع و نوع بسته ی نرم افزاری بسیار مختلف است. البته باید توجه داسته باشید که این بسته ها هم می توانند حاوی سورس کد نیز باشند. نکته ی قابل توجه دیگر این است که این بسته ها می توانند شامل فایل های کمکی (Help) و مستندات دیگر مربوط به برنامه باشند.

شما برنامه ی خود رااز هر راهی که نصب کنید در اغلب موارد فایل های آن در مسیر های usr/bin/ و bin/ و usr/sbin/ قرار می گیرند و فایل های مربوط به پیکره بندی آن در دایرکتوری etc/ قرار می گیرند. بنابراین ما در لینوکس چیز به نام رجیستری نداریم و تنظیمات مربوط به برنامه ها در فایل ها ذخیره می شوند که در بیشتر موارد این فایل هافایل های هستند که به راحتی قابل خواندن و ویرایش کردن با ویرایشگر های معمولی متن می باشند.

بسته ها:چرا به برنامه ها بسته (Pakage) میگویند؟ اغلب برنامه ها امروزه شامل فایل های بسیاری می باشند ،از جمله فایل های باینری ،فایل های پیکره بندی و فایل های مربوط به راهنما و روش نصب و مجوز های برنامه که به Documentation یا مستندات معروف هستند. بنابراین می بینیم یک برنامه همراه خود تعداد بسیاری فایل به همراه دارد ،تمام این فایل ها به صورت فشرده در یک بسته قار می گیرند ،خواه این یک بسته ی RPM باشد یا یک بسته ی DEB یا حتی یک بسته ی Tarball باشد. برای آشنایی هرچه بیشتر شما با انواع بسته های نرم افزاری ما در این مقاله به سه نمونه ی عمده ی بسته های نرم افزاری لینوکس اشاره می کنیم ،باید توجه داشته باشید که در این مقاله قصد آموزش کار با بسته های نرم افزاری را نداریم و تنها به معرفی انواع بسته ها بسنده خواهیم کرد.

. بسته های RPM

یکی از معروف ترین بسته های نرم افزاری لینوکس بسته های RPM میباشد.RPM مخفف RedHat Pakage Manager می باشد ،همان طور که از اسم آن بر می آید ابداع شرکت RedHat Linux می باشد اما در توزیع های مختلفی از این بسته ها استفاده می شودو جالب تر این که از این بسته ها می توانید در سیستم های شبه یونیکس دیگر مانند FreeBSD و SunSolaries نیز قابل استفاده هستند.بسته های RPM با پسوند rpm شناخته می شوند .

از جمله توزیع هایی که از بسته های rpm استفاده می کنند می توان لینوکس Mandrake وLinuxPCC و لینوکس YellowDog را که بر پایه ی ردهت است و برای سیستم های با پردازنده ی PowerPC طراحی شده و نیز توزیع SuSE را نام برد.البته توزیع های دیگری نیز از این بسته ها استفاده می کنند .برای مدیریت بسته های rpm شما می توانید از مدیر بسته های rpm استفاده کنید. اگر می خواهید در خط فرمان و براساس محیط متنی کار کنید می توانید از دستور rpm برای مدیریت استفاده کنید. اما اگر می خواهید از محیط گرافیکی برای مدیریت استفاده کنید با توجه به توزیع شما این ابزار متفاوت است. کاربران SuSE با ابزار YaST می توانند این کار را بکنند ،کاربران Mandrake می توانند با

rpmdrake بسته هایشان رامدیریت کنند. البته باید توجه داشته باشید که SuSE و Mandrake گـاهی از بسته های rpm مخصوص خود اسـتفاده مـی کننـد کـه جـز آن توزیـع در توزیـع دیگـری قابـل اسـتفاده نمی باشد.

بسیارخوب تا این جا بسته های RPM را شناختیم، حال قبل از معرفی بسته های DEB به نحـوه ی نام گذاری بسته های RPM خواهیم پرداخت. برای آن که کاربران بـا دیـدن نـام بـسته ی rpm بـه راحتـی بتوانند متوجه محتویات آن بشوند از یک استاندارد برای نام گذاری این بسته هـا اسـتفاده مـی شـود.ایـن نـام گذاری مانند زیر است :

pakagename-a.b.c-x.arch.rpm

در این نام گذاری همان طور که معلوم است از سمت چپ اولین بخش نام نرم افزار اصلی ماست مـثلا xchat یا gaim اما سایر اجزا :

a.b.c نسخه ینرم افزاری است که دردست دارید، این شماره نسخه را اغلب نویسنده ی برنامه تعیین می کند مثلا ۲,۲,۶ یا a۵,۶,۷.

x این شاره به شماره ی number build یا relase number معروف است.این شماره در واقع نشان دهنده ی تغییرات کوچکیست که اغلب هم کار نویسنده ی برنامه نیست مثلا یک وصله یا Patch برای رفـع یـک مـشکل (bug) یـا یـا تغییـر یـک فایـل نـصب یـا اضـافه شـدن یـک فایـل راهنمـا یـا help یـا Documentation به فایل

arch این بخش از نام مربوط می شود به معماری پردازنده ای که از آن استفاده می کنید. بسته های RPM برروی معماری های مختلفی از پردازنده قابل اجراست ،از جمله x86 یا SPARC یا Alpha نیز-IA 64 بنابر این لازم است که نوع پردازنده ی مورد پشتیبانی را در نام فایـل بگنجـانیم تـا در آینـده بتـوانیم بـه راحتی از آن برای پردازنده ی خودمان استفاده کنیم.برای پردازنده های x86 این جا i386 نوشته می شـود. این پردازنده ها باقی مانده ی نسل ۸۰۳۸۶ هستند که شرکت اینتل آن ها را تولید می کند ،البته بایـد خـاطر نشان کنم که در حال حاضر وقتی می گوییم x86 منظور ما هر پردازنده ای سازگار با ایـن معمـاری اسـت از جمله AMD مدل های Duron و Athlon و یا پنتیوم هـای I,II,II,IV, Proو نیـز پنتیـوم MMX ویـا پردازنده های سازگار با x86شرکت Transmeta یا Cyrix بنابراین امـروزه فقـط ایـن اینتـل نیـست کـه پردازنده های خانواده ی x86 را تولید می کند. اگـر بـسته یRPM شـما بـرای پنتیـوم بهینـه سـازی شـده باشداین مقدار ۱۵۸۶ و i686 خواهد بود. البتـه RPMبـرای پردازنـده هـای دیگـر نیـز وجـود دارد از جملـه PowerPC که با ppc مشخص می شود. نکته ی قابل توجه این است که اگر بسته ی rpm شـما محتـوی فایل متنی یا Documentation یا هر چیز دیگری باشد که به پردازنده (CPU) بستگی ندارد ایـن مقـدار noarch خواهد بود.

برای روشن تر شدن این نام گذاری به نام xbill-2.1.3-85.i386.rpm دقت کنید. xbillنام این نرم افزار است که یک بازی لینوکس است.که بسته ی ما buildشماره ی ۸۵ نسخه ی ۲,۱۳ آن است که برای معماری x86 طراحی شده است.

بسته هایDEB

بسته ی دیگری که بسیار استفاده می شود ،بسته ی دبیان یا DEB می باشد ، این بسته ها نیز مانند بسته های RPM هستند و کار کردن با آن ها مانند کار با RPM ها ساده است. این بسته ها را دبیان توسعه داده است و هم اکنون توزیع های فراوانی از لیبرانت تا DammnSmallLinux از این بسته ها استفاده می کنند ،در واقع هر توزیع مبتنی بر دبیان از این بسته ها استفاده می کند.بسته های rpm و deb را نمی توان به جای یکدیگر استفاده کرد اما می توان آ نها را به یکدیگر تبدیل کرد ،البته این تبدیل نیز همیشه موفقیت آمیز نخواهد بود.بسته های دبیان نیز مانند rpm برای معماریهای متفاوتی نوشته شده است .برای مدیریت بسته های دبیان در خط فرمان از ابزار dpkg استفاده می شود dpkg بسیار شبیه RedHat pakage Manager یا rpm است .(منظور ابزار rpm است که برای مدیریت بسته های RPM استفاده می شود.). البته در دبیان و توزیع های مبتنی بر دبیان ابزار دیگری نیز برای مدیریت بسته های نرم افزاری در خط فرمان وجودداردکه کار با بسته های نرم افزاری را بسیار راحت تر و شیرین تر می کند ،این ابزار APT نام دارد.APt مخفف Advanced Pakage Tool می باشد. (تلفظ کنید اپت /æpt/) یکی از خواص بستههای RPM وDEB این است که شما نی توانید به راحتی و از طریق همین ابزار های مدیریت بسته بسته های خودرا در چند ثانیه یاحداکثر چند دقیقه به روز بکنید و لی اگر برنامه ای را از کد منبع آن کامپایل کنید برای هر بار به روز رسانی مجبورید برنامه را از ابتدا کامپایل کنید.البته برای مدیریت بسته های نرم افزاری در دبیان نیز برنامه های مختلفی وجوددارد مانند synaptic ویا برای مدیریت بسته هادر خط فرمان اما کمی راحت تر مانند dselect . نام گذاری بسته های دبیان نیز مانند RPM است با کمی تفاوت در ظاهر که برای طولانی نشدن بحث به آن اشاره نمی کنیم.

.بسته های Tarball

در حقیقت بسته های Tarball چیزی جز بسته های آرشیو شده نیستند ، این بسته ها را می توانیددر هر توزیعی استفاده کنید و مانند RPM و DEB وابسته به توزیع نیستند. بسته های Tarball همان بسته هایی هستند که با ابزار tar در خط فرمان تولید می شوندو توسط یکی از برنامه های compres یا gzip یا bzip2 فشرده شده اند. این بسته های فشرده شده فقط حاوی فایل ها هستند و بعد از باز کردن آن ها می توان در باره ی روش نصب آن ها تصمیم گرفت. بسته های tarball می توانند مانند RPM و Debian حاوی سورس کد ، فایل های باینری و همچنین مستندات برنامه باشدالبته در اغلب موارد این بسته ا با gzip فشرده می شوند که پسوند حاصل tar.gz. می شود و البته برای این که حجم اطلاعات رد و دلی در اینترنت کمتر شود از فرمت tar.bz2. استفاده می شود که بسته های آرشیو tar را با برنامه ی

bzip2 فشرده کرده اند .(فایل هایی که با Bzip2 فشرده شده اند از فایل هایی که با gzip فشرده شده اند حجم کم تری دارند چرا که الگوریتم فشرده سازی در این دو متفاوت است.)

اطلاعات ضروری در زمان نصب برنامه

۱- وابستگی های برنامه (Dependencies)

وابستگی های نرم افزاری یا همان Dependencies در واقع آن برنامه هاو نیاز هایی است که باید داشته باشید تا برنامه ی شما بتواندکار کند!! درواقع شما برای نصب یک برنامه اغلب به یک پیش نیاز احتیاج دارید ، مثلا برای بسیاری از برنامه های لینوکس لازم است تا شما کتابخانه های C را داشته باشید. در واقع وقتی می گوییم یک برنامه فلان وابستگی ها را می خواهد یعنی اولا بدون آن نیازمندی ها نمی تواند کار کند و در ثانی می توان نتیجه گرفت که آن برنامه از برنامههایی که در اصطلاح وابستگی نام دارند برای کار کردن استفاده می کند.

توجه داشته باشید که مفهوم پیشنهادی یا Recommended با وابستگی متفاوت است ،اگر وابستگی ها را نصب نکنید برنامه ی شما کار نخواهد کرد اما اگر پیشنهادی ها را نصب نکنید برنامه ی شما کار خواهد کرد. پس این Recommended ها به چه کاری می آیند؟ اگر شما این Recommended هارا نصب کنید برنامه ی شما بهتر کار می کنه یا بهتر بگویم بازدهی برنامه ی شما بیشتر خواهدشد.

یکی از خواص بسته های Debian و rpmهوشمندی آن ها نصبت به شناخت این پیش نیاز هاست ،در واقع وقتی شمایک بسته ی Debian را نصب می کنید خودش به دنبال پیش نیاز ها می گردد و پیش نیاز ها را نیز برای شما نصب می کند.البته بسته های RPM این هوشمندی راندارند اما برنامه های مدیریت بسته های rpm این کار را می کنند یعنی زمانی که شما از دستور rpm برای نصب استفاده می کنید این برنامه است که تشخیص می دهد ثبل از نصب باید چه برنامه های دیگری نصب شود و به شمااعلام می کند که این بسته ا را نیز نصب کنیدو یا خودش نصب می کند. اما بسته های دبیان در ذات خود این قابلیت را دارا هستند که پیش نیاز هایشان را تشخیص بدهند ولی بسته های Tarball فاقد این هوشمندی هستند و باید خودتان با مطالعه ی فایل های README و INSATLL ویا دیگر راهنماهای موجود در بسته یا سایت آن پیش نیاز ها را یافته و آن ها را نصب کنید.

۲-کتابخانه ها(Libraries)

کتابخانه ها همان DLL هایی هستند که شما در ویندوز می شناختید.(که اغلب گم شده بودند یا به دلیل خرابیشان برنامها کار نمی کرد!!) در لینوکس نیز کتابخانه ها همان مسئولیت را به عهده دارند اما پسوند خاصی ندارند چرا که هر زبان از کتابخانه های خوداستفاده می کند.خوب این مسئولیت کتابخانه ها چیست؟ کتابخانه ها یکسری فایل هستند که شامل یکسری توابع و روتین برای برنامه می باشند ، شما می توانید یک فایل کتابخانه درست کنید و تمام توابعی که در برنامه ی شما استفاده می شود را در آن قرار دهید و از برنامه ی خود بخواهید که این توابع را از فایل مذکور اجرا کند،این کار چه فایده ای دارد؟ فکر کنید همکار شمانیز می خواهد برنامه ای بنویسید او می تواند از ابتدا شروع کند و تمام توابع رادر برنامه ی خود تعریف کند ،این کار باعث صرف وقت بیشتر و خستگی بیشتر و نیز حجم بیشتر فایل ها می شود.راه حل هوشمندانه این است که توابع مشترک برنامه ها را در یک فایل کتابخانه ای قرار دهید و هر کسی که به آن توابع احتیاج داشت زمان اجرای برنامه این توابع را فرابخواند. به این ترتیب لازم نیست که شما هر بار تمام توابع را تعریف کنید کافیست یکبار این توابع در کتابخانه ای نوشته شوند و کاربران این کتابخانه ا را در سیستم خودنصب کنند ،حالا شما و همارانتان و دیگران بدون دغدغه ی حجم زیاد و تعریف مجدد توابع به نوشتن برنامه مشغول می شوید.(حالا متوجه شدید چرا به این برنامه ها کتابخانه می گویند؟

از جمله معروف ترین کتابخانه می توان به کتابخانه های C اشاره کرد که با نام libc و در توزیع های مدرن تر به نام glibc وجود دارند و یا کتابخانه ی معروف GTK+ که شما یک بار نصب می کنید و تمام برنامه های محیط گرافیکی گنوم از آن استفاده می کنند.

۳- ابزار های توسعه(Development Tools)

اگر شما تصمیم به کامپایل یک برنامه گرفته اید باید فایل های کتبخانه ای و ابزار های توسعه ی مربوط به آن برنامه را داشته باشید. مثلا اگر برنامه ای به زبان C دارید باید یک کامپایلر برای C داشته باشید و کتابخانه های مربوط به آن را نیز باید در سیستمتان داشته باشید. اگر شما نیازی به کامپایل برنامه ها ندارید لازم نیست تا ابزار های توسعه رانیز نصب کنید. درواقع وقتی شما یک برنامه را کامپایل می کنید مانند این است که شما برنامه نویسید و برنامه رانوشتیدو کد آن راآماده کردید و می خواهید آن را به زبان ماشین ترجمه کنید،خوب برای این کار به ابزاری نیاز دارید به نام

Development toolsکه بوسیله ی آن برنامه ی خودرا کامپایل کنید. برنامه ی معروف gcc جزو این ابزار هاست ، gccکه یک کامپایلر است و تحت مجوز GNU/GPLنیز منتشر شده است مجموعه ای است از کامپایلر های زبان های مختلف، البته این اسم آن نیز معلوم است!!!!GNU Compiler Collection

۴-.پایگاه داده ی بسته های نصب شده

پایگاه داده ی بسته های نصب شده یا به زبان ساده تر Installed package Database !! محلی است که اطلاعات بسته های نصب شده ی سیستم در آن قرار دارد. شاید شما بخواهیدبدانید که فایل کتابخانه ای X یا Y آیا نصب شده است یا نه؟ یا می خواهید بدانی چه نسخه ای از برنامه در سیستم شما موجود است ،یا حتی می خواهید بدانید برای برنامه ی شما چه نیاز مندی هایی لازم است؟ تمام این اطلاعات و اطلاعات دیگر در یک پایگاه داده ذخیره می شود.برای بسته های Debian این مسیر اغلب در مسیر var/lib/dpkg/ است و برای RPM این مسیر var/lib/rpm/است ،اما برای Tarball هیچ پایگاهی وجود ندارد!! البته توزیع SlackWareخودش توسط برنامه ای به نام slackware Package Tool یا pkgtoolبرای بسته های Tarball ای که از طریق این برنامه نصب می شوند یک پایگاه داده می سازد اما این فقط مختص توزیع است و این امکان در تمام توزیع ها موجود نیست.به همین دلیل است که اغلب بسته های Slackware را جزو بسته های هوشمند به شمار نمی آورند چرا که تنها یک آرشیو فشرده است و حتی نمی تواند وابستگی ها یا Dependency های خود را پیدا کند.

نصب نرم افزار در لینوکس ، تمام آنچه در این مقاله خواندید روش های مختلف نصب در لینوکس است ،می بینید که شما انتخاب های مختلفی دارید و به یک انتخاب وابسته نیستید ، امیدوارم این مقاله برایتان مفید بوده و مطالبی جدید برایتان داشته باشد.

تألیف: مریم مهربخش