

گام دنیا

```
DELIMITER $$  
CREATE FUNCTION hello_world()  
RETURNS TEXT COMMENT 'Hello World'  
BEGIN  
RETURN 'Hello World';  
END;  
$$  
DELIMITER ;  
SELECT hello_world();
```

ماهنامه تخصصی نرم افزارهای آزاد / متن باز | شماره هفتم | شهریور ماه ۱۳۹۴ | ۱۰۲ صفحه



پرونده:

آشنایی با مدل‌های مدیریت داده‌ها

با انواع پایگاه داده‌های پر کاربرد آشنا شوید

روزی که مدیر سیستم‌ها بر زمین حکم راندند
زندگی با شیطانک قرمز
متن باز شدن دیگر اختیاری نیست، حتی برای اپل
۷ کتابی که هر مدیر جامعه‌ای باید مطالعه کند
تدابیری که دولت‌ها برای ترویج نرم افزار آزاد باید به کار گیرند
اهمیت چرخش آزاد اطلاعات دولتی
صرفه جویی میلیون دلاری و دانش آزاد
واکنش به حملات سایبری مهم‌تر از جلوگیری است





دوره های حرفه ای لینوکس

در تلهما مرکز تخصصی آموزش و آزمون بین المللی لینوکس ایران
فناوران آنیسا

- ◀◀ دوره آشنایی با لینوکس (Linux Essential)
- ◀◀ دوره مدیریت پایه لینوکس LPIC-۱
- ◀◀ دوره مدیریت پیشرفته لینوکس LPIC-۲
- ◀◀ دوره پیشرفته محیط های مختلط LPIC-۳۰۰ (LDAP, Samba)
- ◀◀ دوره پیشرفته امنیت در لینوکس LPIC-۳۰۳
- ◀◀ دوره پیشرفته مجازی سازی در لینوکس LPIC-۳۰۴
- ◀◀ دوره مقدماتی و پیشرفته Embedded Linux
- ◀◀ دوره پیشرفته مانیتورینگ لینوکس
- ◀◀ دوره مقدماتی و پیشرفته Astrisk VoIP
- ◀◀ دوره استادی اسکریپت نویسی (Mastering Linux Shell Scripting)
- ◀◀ دوره طراحی و برنامه نویسی وب (LAMP)
- ◀◀ برگزاری آزمون های بین المللی LPI
- ◀◀ و هم اکنون دوره تخصصی رایانش ابری (Cloud Computing)

نشانی ما: میدان آرژانتین، بلوار بیهقی، دوازدهم شرقی، پلاک ۶ - تلفن تماس: ۸۸۵۴۸۶۰۳ و ۸۸۵۴۸۳۶۰

www.anisa.co.ir



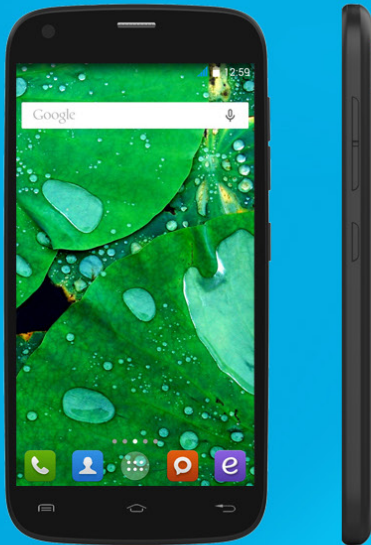
پیشنهاد شگفت انگیز








تا ۱۰ | مگابیت | ۵۰ روز | ۵ گیگ

قیمت : ۵۰۰۰ تومان

شماره تماس: ۲۳۵۴۵









Spring Pro



- | | | | |
|---|---|--|---|
| 
Quad Core 1.3
MTK6582 | 
RAM 1 GB
Built-in Storage 16 GB | 
IPS LCD 4.6"
540x960 | 
8.0 Mega Pixel |
| 
Android 4.4 | 
Dual SimCard | 
1,500 MA | |








Spider 1 Pro



- | | | | |
|---|---|---|--|
| 
Quad Core 1.3
MTK6582 | 
RAM 2 GB
Built-in Storage 16 GB | 
IPS LCD 5.0"
720x1280 | 
13.0 Mega Pixel |
| 
Android 4.4 | 
Dual SimCard | 
USB On-The-Go | 
2,000 MA |



Crystal



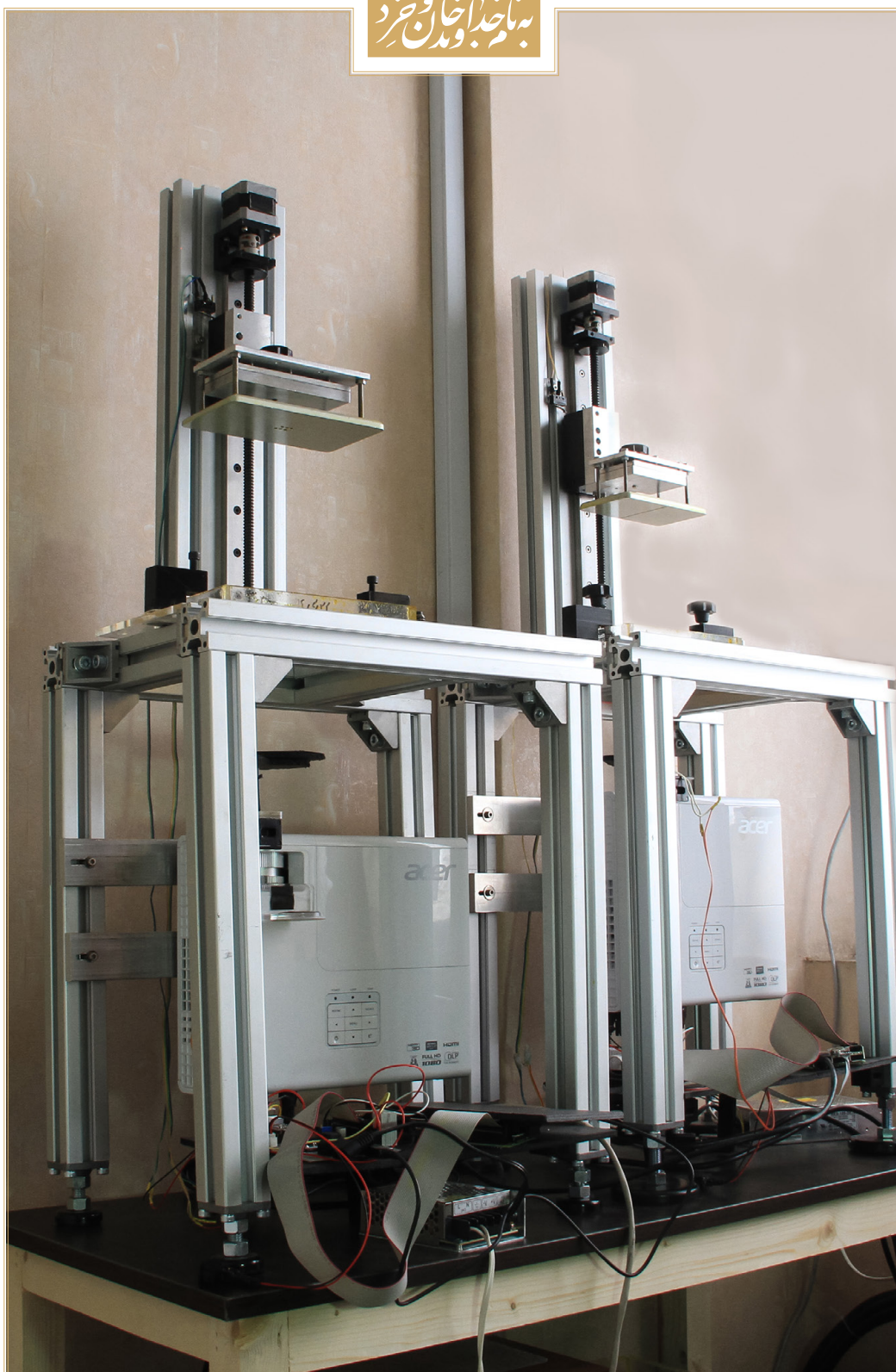
- | | | | |
|---|--|---|---|
| 
Dual Core 1.3
MTK6582 | 
RAM 512 MB
Built-in Storage 4 GB | 
IPS LCD 7.0"
600x1024 | 
2.0 Mega Pixel |
| 
Android 4.4 | 
Dual SimCard | 
USB On-The-Go | 
3,000 MA |

Spark Pro



- | | | | |
|---|--|---|---|
| 
Quad Core 1.3
MTK6582 | 
RAM 1 GB
Built-in Storage 4 GB | 
IPS LCD 5.0"
720x1280 | 
8.0 Mega Pixel |
| 
Android 4.4 | 
Dual SimCard | 
USB On-The-Go | 
2,000 MA |

بنام خداوند عز و جل



چاپگر سه بعدی ایرانی ساخته شده بر اساس دانش آزاد

سلام دنیا

ماهنامه تخصصی نرم افزارهای آزاد / متن باز

شماره هفتم | شهریور ماه ۱۳۹۴

صاحب امتیاز و مدیر مسئول:

یاسر توکلی کرمانی

سردبیر:

محمد دماوندی ■ eic@salam-donya.ir

شورای سردبیری:

محمد افاضاتی

محمد درویش

علی فارمد

احسان کریم خانی

محمد نبی زاده

همکاران این شماره:

دانیال بهزادی ■ مهدی بیگی ■ احسان ترک

امیر حسین حسینی پژوه ■ سپهر حمزه لوی ■ مهدی حمیدی

مهدی خشنودی ■ محمد خلایق ■ حمیدرضا سلیمانی

وحید سهرابلو ■ سهراب سهیلیان ■ نیما صحرانشین سامانی

مسعود صدرنژاد ■ سعید علیجانی ■ سعیده علی محمدی

جادی میرمیرانی

مدیر هنری: علیرضا بخشی

عکاس: نگار مصطفوی ■ محسن کرامت دهر

نشانی: تهران - فلکه دوم صادقیه ■ ابتدای بلوار فردوس

پلاک ۱۴ ■ واحد ۱ ■ تلفن: ۰۲۱-۴۴۰۰۷۵۱

www.salam-donya.ir ■ info@salam-donya.ir



معرفی

معرفی پروژه ژاکارد | ۱۲ |

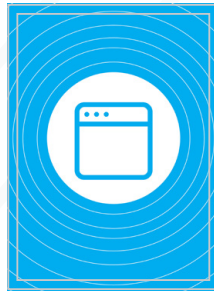
لیبره آفیس برای اندروید | ۱۳ |

مکالمه صوتی از طریق مامبل | ۱۴ |

نرم افزار آزاد، به داد پرونده های پاک شده می رسد | ۱۵ |

۷ برتری جست و جوگر داک داک گو نسبت به گوگل و بینگ | ۱۸ |

معرفی ۸ افزونه کاربردی برای مرورگر گوگل کروم | ۲۲ |



کسب و کار

اهمیت چرخش آزاد اطلاعات دولتی | ۲۶ |

چگونه کسب و کار موفق را راه اندازی کنیم | ۳۰ |

تجربیات مدیرعامل ردهت از چندین سال مدیریت | ۳۲ |

متن باز شدن دیگر اختیاری نیست حتی برای اپل | ۳۴ |



جامعه کاربری

تدابیری که دولت ها می توانند برای ترویج نرم افزار آزاد به کار گیرند | ۳۶ |

نقش جامعه و دولت در حمایت از نرم افزار آزاد | ۳۸ |

زندگی با شیطانک قرمز | ۴۰ |

نامه سرگشاده مدیرعامل موزیلا به مدیرعامل مایکروسافت | ۴۵ |

بهانه های استفاده نکردن از گنو / لینوکس | ۴۶ |

صرفه جویی میلیون دلاری و دانش آزاد | ۴۸ |

آیا نرم افزار امنیتی آزاد امنیت پایینی دارد؟ | ۵۰ |



پرونده

آشنایی با مدل های مدیریت داده ها | ۵۲ |

با انواع پایگاه داده های پر کاربرد آشنا شوید | ۵۶ |

ACID در مقابل BASE | ۶۰ |

فهرست گذاری در پایگاه داده چگونه کار می کند؟ | ۶۲ |

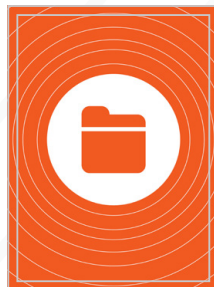
موتورهای پایگاه داده در مایکروسافت | ۶۶ |

ارتباط پایگاه داده ها با دنیای خارج | ۶۹ |

سرگذشت یک پروژه متن باز | ۷۴ |

نکاتی که در فارسی باید رعایت کنیم | ۷۵ |

سه محدودیت NoSQL | ۷۶ |



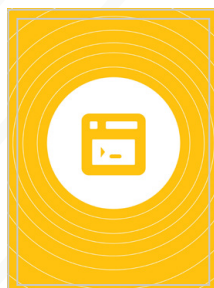
تخصصی

نحوه احیای سرور | ۸۰ |

ایمکس جادوی گنو | ۸۲ |

ماجرای سیس ادمنی | ۸۸ |

واکنش به حملات سایبری مهم تر از جلوگیری است | ۹۰ |



داستان علمی تخیلی | ۹۱ |



محمد دماوندی
سردبیر

نرم افزارهای مختلف از مرورگر و دستیارهای نرم افزاری مانند کورتانا، سیری و مشابه آنها گرفته تا سیستم عامل، از تلفن همراه، تبلت و ساعت هوشمند گرفته، حتی تا تلویزیون، همه و همه نظاره گر رفتار، اعمال و گفتار ما هستند و در آینده ای نه چندان دور، خودرو و حتی پیراهنی هم که به تن داریم به جمع این ابزار خواهند پیوست. بسیاری از این تجهیزات بدون آن که ما بخواهیم به بهانه های مختلف، از جمله کمک در بهبود برنامه و کارایی آن، همگام سازی دستگاه های مختلف و یا به منظور به اصطلاح تسهیل در امور روزمره، در تکاپوی جمع آوری اطلاعات ما هستند. شاید با استفاده از نرم افزارهای آزاد / متن باز بخشی از این تلاش ها را بتوانیم کنترل کنیم. اما وقتی که پای سخت افزار هم به میان می آید، دست هایمان کاملاً بسته است. وقتی پی می بریم که «دوستان بزرگوار» از طریق پردازنده هم به جمع آوری داده های ما مشغول هستند، دیگر دلیل واهی تسهیل در امور روزمره کاملاً رنگ می بازد و آسودگی خاطر برای لذت بردن از امور تسهیل شده، جای خود را به ترس و ناامنی می دهد.

از چند دهه گذشته تا همین چندسال پیش، جمع آوری داده ها یکی از دغدغه های اصلی بود. بیشتر سازمان ها هزینه زیادی برای دسترسی به اطلاعات کاربران صرف می کردند تا بدانند دیدگاه کاربرانشان چیست، بینندگان تلویزیونی از کدام برنامه رضایت بیشتری دارند، کاربران مختلف ابزارشان را با چه کاربردهای خلاقانه ای به کار می گیرند، سیستم در حالت های خاص چه رفتاری از خود بروز می دهد و مواردی از این دست. امروزه با گسترش اینترنت و ابزارهای همراه این نیاز تا

حد زیادی مرتفع شده و اکنون گرایش ها بیشتر به کاربرد داده ها سوق پیدا کرده و دیگر سازمان ها آن قدر اطلاعات دارند که دسترسی آسان، بهینه و ساخت یافته به آنها اهمیت بیشتری پیدا کرده است.

پیدایش مدل های جدید پایگاه های داده، مفاهیم «هوش کسب و کار» را که سال ها پیش توسط ریچارد میلارد ارایه شده بود، وارد دوره جدیدی می کند. آن قدر اطلاعات داریم که به سرعت از تعاریف می گذریم و بتوانیم شاهد ظهور حوزه های جدیدی در پردازش داده ها باشیم. هوش تجاری ابری، هوش تجاری موبایل و سایر موارد همه زاده دیوی انبوه اطلاعات است.

در گذشته گزینه های محدودی برای ذخیره و باز یابی داده ها داشتیم و امروز شاهد پیدایش پایگاه های داده خاص منظوره هستیم. به لطف عرضه محصولات آزاد / متن باز نیز می توانیم برای نیازهای خود به جای اختراع مجدد چرخ بر بهبود چرخ های موجود تمرکز کنیم. حتی برای بخش های مختلف یک سامانه نیز می توانیم فناوری های مختلف را ترکیب و خدمات کارآمدتری ارایه کنیم

بر اساس آمار ارایه شده توسط «برادر بزرگ»، جستجوی عباراتی همچون Big Data در ده سال اخیر بیش از ۱۷ برابر رشد داشته است.

تمامی این نکات ما را به درک این نکته رهنمون می سازد که بر لبه عصر مدیریت اطلاعات و پایگاه داده ها ایستاده ایم و باید منتظر تغییر نگرش، ابزارها و کاربردهای مختلف داده ها و اطلاعات باشیم. تبدیل داده ها به اطلاعات و اطلاعات به دانش در چشم برهم زدن صورت می پذیرد. مدیران به اهمیت این موضوع پی برده اند و ذائقه شان تغییر یافته است.

استفاده از یک نمودار تحلیلی ساده، بدون چالش، اولویت ها را مشخص می کند؛ هرچند تهیه همین نمودارها نیز مستلزم دانش، مهارت و تیزبینی است.

چگونه است که شبکه اجتماعی فیس بوک با بیش از ۱,۵۰۰,۰۰۰,۰۰۰ کاربر فعال در ماه، بدون اختلال محسوس، بر روی پایگاه های داده مای اس کیوال استوار است، ولی در کشور عزیزمان، هنگام اعلام نتایج کنکور، خواب بر مدیران سیستم حرام می شود؟

بنابراین در وهله اول باید با دوراندیشی و ترسیم چشم اندازی مشخص برای نحوه مدیریت و کاربرد داده هایمان، بستر مناسب را انتخاب کنیم و بر اساس نیازمندی ها، قرارگاه اطلاعات را طوری مشخص کنیم که با گذر زمان به آرامگاه اطلاعات تبدیل نشود. انتخابی صحیح، مستلزم شناختی دقیق است. بدون داشتن درک کاملی از نیازمندی های فعلی و آتی و بررسی جزئیات پایگاه داده ها، شناخت نواقص و ویژگی های برتر هر یک، هر انتخابی دچار اختلال می شود.

در این شماره به بررسی مدل های مختلف مدیریت داده ها و ابزارهای آزاد / متن باز پرداخته ایم تا انگیزه لازم را جهت کاربرد پایگاه های داده قدرتمند آزاد برای متخصصان و مدیران خلاق ایجاد کنیم و این پیام مهم را به گوش کاربران متعصب پایگاه های داده انبوهی که اگر حجم انبوهی از داده ها را در اختیار دارید و مدیریت می کنید، اگر تعداد تراکنش های بالایی برای خود متصور هستید، آگاه باشید که استفاده از ابزار انحصاری و گران قیمت، الزاماً مدیریت بهتر را به ارمغان نمی آورد.


<http://sfd.fsug.ir/1394/>


روز آزادی نرم افزار

جشنواره روز آزادی نرم افزار تهران، مورخ ۹ مهر ۱۳۹۴ از ساعت ۹ تا ۱۶ در دانشگاه صنعتی شریف، در دو بخش عمومی و کارگاه‌ها و با محوریت معرفی، ترویج، افزایش کاربرد و استفاده، جذب مشارکت و حمایت جامعه و تولید در زمینه نرم افزار آزاد برگزار خواهد شد. علاقه‌مندان می‌توانند مقالات و موضوعات پیشنهادی خود را جهت ارائه در بخش کنفرانس، یا کارگاه‌ها تا تاریخ ۲۰ شهریور از طریق صفحه ارسال مقاله به دبیرخانه جشنواره ارسال کنند، تا پس از داوری، مقالات برگزیده برای ارائه در روز جشنواره اعلام شوند. کلیه موضوعات تخصصی یا عمومی در حوزه نرم افزار آزاد قابل پذیرش است. برای اطلاعات بیشتر درباره روش داوری، معیارهای انتخاب مقالات و نحوه تنظیم و ارسال مقاله سایت جشنواره را مطالعه کنید. ■


<http://slmd.ir/jx>


جشن تولد ۳۰ سالگی بنیاد نرم افزار آزاد

بنیاد نرم افزار آزاد، دعوت باز خود را برای شرکت در جشن تولدش، اعلام کرد. این جشن در غروب شنبه، یازدهمین روز ماه مهر ۱۳۹۴ در بوستون برگزار می‌شود و از حضور رئیس و موسس بنیاد نرم افزار آزاد، استالمن بهره‌مند خواهد شد. این جشن نشان دهنده اوج سی سالگی است که دربردارنده بزرگترین کنفرانس «لیبره‌پلنت» تا کنون و یک مقاله درباره گنو در نیویورک است. علاوه بر این‌ها، بنیاد یک کنفرانس کوچک رانیز در نظر گرفته است که زمانش ۱۱ مهر ۹۴ و در طول روز است. در این کنفرانس، جامعه نرم افزار آزاد درس‌هایی که در این سی سال به دست آورده است را به اشتراک می‌گذارند و برای آینده برنامه‌ریزی می‌کنند. همچنین احتمال برگزاری یک مراسم شام برای جمع آوری کمک مالی در جمعه ۱۰ مهر ماه نیز وجود دارد. به عنوان رویدادهای موازی، در سراسر جهان نیز حامیان، علاقه‌مندی خود را به برگزاری یک رویداد محلی برای این تولد بیان کرده‌اند. بنیاد از پوشش دادن این رویدادها بر روی بلاگ خود یا اتصال آن‌ها به جشن اصلی خوشحال خواهد شد و استقبال می‌کند. اگر علاقه‌مند به برگزاری رویدادی مشابه در منطقه خود هستید با campaigns@fsf.org تماس بگیرید. ■


<http://slmd.ir/gz>

این دستگاه افکار تان را تایپ می‌کند

در مقاله‌ای از نشریه مادربرد، به تحقیقات انجام شده در آزمایشگاه شالک اشاره شده است که در آن محققین به تازگی نشان داده‌اند می‌توان افکار یک فرد را با چیزی که آن را «مغز به نوشتار» می‌نامند تبدیل به عباراتی خوانا کرد. جیسون کبلر می‌نویسد «این تازه روزهای آغازین این فناوری است، الکترودها دقیقاً باید بر روی مغز جای گذاری شوند و واژه نامه عبارات نیز محدود است. با این حال امواج مغزی ساطع شده از الگوی فکری با احتمالی بیشتر از شانس به متن تبدیل می‌شوند. ■»


<http://slmd.ir/kf>

تهدیدی برای فایل‌ها با فرد میان ابری

با استفاده روشی به نام «فرد میان ابری»، مهاجمان می‌توانند به واسطه سرویس همگام‌سازی ابری، به فایل‌های شما در گوگل درایو، دراپباکس و وان درایو بدون زمر، دسترسی پیدا کنند. براساس پژوهش‌های اخیر که در کنفرانس بلک‌هت در لاس‌وگاس در ۱۴ مرداد ۹۴ مطرح شده است، نحوه دسترسی به فضای سرویس‌های ذخیره‌سازی ابری، به نمایش درآمده است. ■

<http://slmd.ir/kw>

اعطای بیش از ۲۶ میلیارد تومان تسهیلات

در قالب فراخوان اول اعطای تسهیلات از محل وجوه اداره شده، اعطای بیش از ۲۶ میلیارد تومان تسهیلات به ۷۳ شرکت خصوصی فعال در حوزه ارتباطات و فناوری اطلاعات به تأیید کمیته وجوه اداره شده وزارت ICT رسید. این را دکتر صمیمی مدیر کل توسعه فناوری و دبیر وجوه اداره شده وزارت ارتباطات و فناوری اطلاعات، با اعلام پایان فرآیند ارزیابی و اعطای تسهیلات در چارچوب فراخوان سال ۹۳، اعلام کرد.

دکتر صمیمی با اشاره به استفاده حداکثری از بضاعت فنی موجود در بخش خصوصی کشور برای ارزیابی طرح های دریافتی، افزود: با پیشنهاد کمیته وجوه اداره شده و تأیید وزیر ارتباطات، از این پس نمایندگان نظام صنفی رایانه ای و سندیکای صنعت مخابرات به صورت رسمی عضو کمیته های تخصصی وجوه اداره شده خواهند شد و تعامل با این تشکلهای برای معرفی نمایندگان مربوطه در حال انجام است. ■

<http://slmd.ir/ku>

آزاد رسانی سامانه ترگمان صورت گرفت

سامانه ترجمه ماشینی پژوهشگاه ارتباطات و فناوری اطلاعات (مرکز تحقیقات مخابرات ایران) موسوم به «ترگمان»، با امکان ترجمه روزانه بیش از یک میلیون کلمه، آزاد رسانی (متن باز) شد. به گزارش روابط عمومی پژوهشگاه ارتباطات و فناوری اطلاعات، این سامانه که در راستای حمایت از توسعه صنعت ترجمه ماشینی طراحی و فعال شده است، حاصل تلاش چند ساله پژوهشگاه با همکاری دانشگاه صنعتی امیرکبیر است. این گزارش حاکی است، کدها و مستندات این محصول در سامانه اشتراک فایل OpenGitir که شامل ابزارهای لازم جهت راه اندازی یک مترجم ماشینی آماری دوسویه انگلیسی/فارسی است، بارگذاری شده است. خدمت ترجمه ماشینی مبتنی بر محصول ترگمان در نشانی اینترنتی targoman.ir ارائه می شود و دسترسی به کدها و مستندات سامانه ترجمه ماشینی پس از ورود به سامانه آزاد رسانی فوق و مراجعه به قسمت مترجم ماشینی ترگمان و ثبت نام مطابق با ضوابط مربوطه امکان پذیر است. گفتنی است سامانه ترجمه ماشینی ترگمان امکان راهنمایی، پاسخ گویی به پرسش کاربران و آرایه اطلاعات تکمیلی را دارا است. ■

<http://slmd.ir/kv>

آخرین وضعیت شبکه ملی اطلاعات تشریح شد

به گفته علی اصغر انصاری بخش اختصاصی شبکه ملی اطلاعات، ارتباط دستگاه های اجرایی با زیر مجموعه است و این ارتباطات در کشور ۸۰ درصد برقرار شده است. معاون توسعه و مدیریت شبکه ملی اطلاعات در همایش معرفی شبکه ملی اطلاعات و خدمات مرکز داده استان کرمان اظهار کرد: تحقق شبکه ملی اطلاعات به عنوان زیرساخت ارتباط فضای مجازی کشور نیازمند شبکه های متشکل از زیرساخت های ارتباطی با مدیریت مستقل کاملاً داخلی، شبکه ای با قابلیت عرضه انواع خدمات امن به کلیه کاربران و شبکه ای پر ظرفیت، پهن باند و با تعرفه رقابتی است.

همچنین وی افزود که در حال حاضر قریب به ۱۰ میلیون پورت پرسرعت ثابت در کشور راه اندازی و به بهره برداری رسیده است. سیاست های ابلاغی مقام معظم رهبری در برنامه ششم توسعه در حوزه فناوری اطلاعات، نمونه جدیدی است و تاکنون به این صراحت نیامده است. وی اظهار داشت: در بند ۳۲ به کسب جایگاه برتر منطقه در توسعه دولت الکترونیک در بستر شبکه ملی اطلاعات اشاره شده و در این زمینه ماموریت ها کاملاً تفکیک شده است. ■

<http://slmd.ir/ggc5migrate>

به روز رسانی بزرگ

نگارش ۵ جی سی سی در اردیبهشت ماه عرضه شد و اکنون توزیع های دبیان و اوبونتو در فکر به روز کردن تمام بسته های نرم افزاری خود هستند تا در نهایت نرم افزار هایشان با این نگارش جدید، کامپایل شود.

جی سی سی، مخفف GNU Compiler Collection به معنی مجموعه کامپایلرهای گنو، کدهای مبدأ نوشته شده توسط برنامه نویسان را به کتابخانه ها و کدهای اجرایی تبدیل می کند. از این مجموعه کامپایلر برای ساخت هر چیزی، از لینوکس گرفته تا نرم افزارهای کاربردی، استفاده می شود. پنجمین نگارش جی سی سی تغییرات اساسی بیشتری را نسبت به نگارش های پیشین معرفی کرده است و نخستین باری است که به طور کامل از نگارش آخر زبان برنامه نویسی سی پلاس پلاس پشتیبانی می کند. برای به روز کردن بسته ها باید تمام آن ها را از ابتدا کامپایل کرد. ■


<http://slmd.ir/kt>


مرحله نخست فراخوان جویشگر بومی

در مرحله نخست فراخوان جویشگر بومی، ۷۹ پیشنهاد از ۳۸ شرکت دریافت شده است. نشست شورای راهبری جویشگر بومی با حضور رئیس و اعضای این شورا در محل سازمان فناوری اطلاعات با دستور «ارایه نتیجه نهایی فراخوان، بررسی دستورالعمل گرانٹ پژوهشی، بررسی نهایی نظام نامه مدیریت طرح و فراخوان مشاور خارجی» برگزار شد.

دکتر علیرضا یاری مجری طرح جویشگر بومی معیارهای کلان ارزیابی فنی پیشنهادها را مشخصات محصول (آماده، با کمی تغییر آماده، در حال توسعه، دمو و پژوهش)، توسعه داخلی بودن یا استفاده از محصول های متن باز موجود، سابقه ارایه محصول، نوع خدمت دهی محصول در حال حاضر (برخط عمومی و خصوصی، آفلاین)، شمار کاربران برخط محصول، یکپارچه با دیگر خدمات شرکت، امکانات ویژه در مقایسه با رقبا، معماری مبتنی بر ابر و سرویس پایه برای خدمات طرح بیان کرد.

همچنین درباره پیشنهادهای رسیده برای فراخوان، مقرر شد پیشنهادهای پژوهشی از غیر پژوهشی جدا شده و دانشگاه ها با تاسیس شرکت در فراخوان طرح جویشگر بومی شرکت کنند ضمن آن که اولویت نخست با سرمایه گذاری مشترک خواهد بود.


<http://slmd.ir/kz>


فایرفاکس صدا را قطع می کند

موزیلا در حال کار بر روی ویژگی است تا به وسیله آن بتواند زبانه هایی از فایرفاکس که در آن ها موسیقی در حال پخش است را مشخص و با نمایش دادن یک آیکون به کاربر بر روی زبانه، امکان قطع صدا را فراهم کند. البته باید اشاره کرد که کروم نیز بیش از این امکان مشخص کردن زبانه در حال پخش را داشت، اما قطع کردن صدا در آن خیلی ساده نبود. شده اند. اگر به دنبال دیدن این قابلیت در کروم هستید باید با رفتن به آدرس <chrome://flags> و کلیک کردن بر روی `enable-tab-audio` آن را فعال کنید.


<http://slmd.ir/kb>

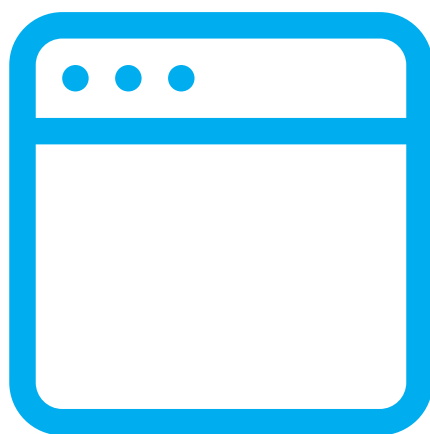

کشف آسیب پذیری ۱۸ ساله در پردازنده ها

آسیب پذیری ۱۸ ساله ای در پردازنده های x86 اینتل کشف شده است که درها را به روی روت کیت ها باز می کند. پژوهشگر امنیتی، کریستوفر دوماس، روشی یافته تا بتواند روت کیت را بر روی میان افزار رایانه نصب کند. این روش از نقضی در طراحی تراشه هایی که از سال ۱۹۹۷ میلادی ساخته شده است، استفاده می کند. این روت کیت حالت مدیریت سیستم پردازنده (System Management Mode) را تحت تاثیر قرار می دهد و امکان پاک کردن UEFI یا حتی تاثیر مجدد پس از نصب سیستم عامل را می دهد. راه کارهای حفاظتی مانند Secure Boot کمکی نمی کند چرا که با اعتماد کامل بر SMM، امن شده اند. این روش تنها بر روی پردازنده های x86 اینتل آزموده شده اما به گفته دوماس، پردازنده های AMD هم باید دارای چنین آسیب پذیری ای باشند.


<http://slmd.ir/kx>


لیبره آفیس ۵ منتشر شد

نسخه جدیدی از نرم افزار لیبره آفیس ویرایش تازه «Fresh» منتشر شد. این ویرایش معمولاً دارای جدیدترین ویژگی های موجود در این مجموعه اداری است و از تمامی ویژگی های تازه برخوردار است. با وجود این، کاربرانی که به پایداری بیشتر اهمیت می دهند نیز قادر خواهند بود از ویرایش دیگری با نام «Still» استفاده کنند که نسخه گذشته از مجموعه اداری لیبره آفیس به همراه رفع باگ تا چندین ماه را شامل می شود که به جای تمرکز بر افزودن موارد و ویژگی های جدید بر رفع نواقص تکیه دارد. لیبره آفیس جدید از ویژگی های متنوعی برخوردار بوده است که این نسخه را می توان بزرگ ترین تغییرات در این مجموعه نرم افزار اداری از بدو تاسیس بنیاد اسناد و انشعاب این مجموعه از این آفیس به حساب آورد.



معرفی

معرفی پروژه ژاکارد | ۱۲

لیبره آفیس برای اندروید | ۱۳

مکالمه صوتی از طریق مامبل | ۱۴

نرم افزار آزاد، به داد پرونده های پاک شده می رسد | ۱۵

۷ برتری جست و جوگر داک داک گو نسبت به گوگل و بینگ | ۱۸

معرفی ۸ افزونه کاربردی برای مرورگر گوگل کروم | ۲۲



معرفی پروژه ژاکارد

انقلاب دنیای منسوجات و فناوری‌های پوشیدنی در راه است

کنترل کند.

ال‌ای‌دی‌ها، هپتیک‌ها (خروجی را به شکل لرزش یا فشار تولید می‌کنند) و خروجی‌های تعبیه شده دیگر، بازخورد را برای کاربر ایجاد می‌کنند و به شکل یک پارچه آن‌ها را به دنیای دیجیتال متصل می‌کنند.

تولید در مقیاس

تولید اجزای ژاکارد مقرون به صرفه است و نخ‌ها و پارچه‌ها را می‌توان با استفاده از تجهیزات استاندارد که در سراسر جهان استفاده می‌شود بافت. یک دستگاه بافندگی می‌تواند طرح‌های مختلفی را به ازای هر فردی در جهان تولید کند.

ساخت پوشاک متصل

پوشاک متصل امکانات جدیدی را برای تعامل با خدمات، وسایل و محیط‌ها می‌دهند. این تعاملات را در هر لحظه‌ای می‌توان دوباره پیکربندی کرد. ژاکارد بوم سفیدی برای صنعت مد است. طراحان بدون آن که نیاز به دانستن الکترونیک داشته باشند می‌توانند از آن به جای هر پارچه‌ای استفاده کنند و لایه جدیدی از قابلیت‌ها را به طراحی خود بیفزایند. توسعه‌دهندگان می‌توانند هر برنامه و خدمت موجود را به پوشاک دارای فن‌آوری ژاکارد خود متصل کنند. و ویژگی جدیدی را برای بن‌سازه خود ایجاد کنند. هم‌چنین متولیان اصلی، در حال توسعه اتصالات سفارشی، ابزارهای الکترونیکی، پروتکل‌های ارتباطی و یک زیست‌بوم از نرم‌افزارهای ساده و خدمات ابری هستند.

بافتن پارچه‌های تعاملی

با استفاده از نخ رسانا می‌توان نقاط لمس یا مناطق حساس به حرکت را در محل‌های دقیق در هر نقطه از منسوج بافت. راه دیگر، بافتن شبکه‌ای از حسگرها در سرتاسر منسوج است تا سطحی بزرگ و تعاملی ایجاد شود.

لوازم الکترونیکی تعبیه شده

اجزای مکمل کاملاً مهندسی شده‌اند تا بتوانند تا حد ممکن گسسته باشند. ما تکنیک‌های نوآورانه‌ای را توسعه داده‌ایم تا نخ‌های رسانا را به اتصالات و مدارهای کوچکی (کوچک‌تر از دکمه‌های یک کت) متصل کنیم. این ابزارهای الکترونیکی مینیاتوری، فعل و انفعالات لمسی و حرکات مختلف را ضبط می‌کنند. با استفاده از این داده‌های ضبط شده و الگوریتم‌های یادگیری ماشین، امکان استنباط بسیاری از حرکات به وجود می‌آید. داده‌های ضبط شده از حرکت و لمس، به شکل بی‌سیم به تلفن همراه یا وسایل دیگر انتقال داده می‌شود تا طیف گسترده‌ای از توابع، اتصال کاربر به خدمات ابری، برنامه‌ها یا قابلیت‌های تلفن همراه را

گروه فن‌آوری و پروژه‌های پیش‌رفته گوگل در حال تحقیق و توسعه محصول جدیدی هستند که می‌تواند انقلابی جدید را در عصر فن‌آوری رقم بزند. این فعالیت در قالب پروژه ژاکارد در حال پیگیری است. هدف پروژه ژاکارد (Project Jacquard)، ممکن کردن بافت لمسی و تعاملی در هر منسوجی با استفاده از دستگاه‌های بافندگی صنعتی است. ژاکارد به طراحان و توسعه‌دهندگان اجازه می‌دهد تا منسوجات حساس به لمس را در محصولات خود به کار ببرند و اشیای روزمره مانند لباس‌ها و میلمان را به سطوح تعاملی تبدیل کنند.

ویسندگی نخ رسانا

این اتفاق به لطف نخ رسانای جدید که در همکاری با شریک صنعتی‌مان ساخته شده است، امکان پذیر شده است. ساختار نخ ژاکارد، آلیاژهای فلزی نازک را با الیاف طبیعی و مصنوعی مانند پنبه، پلی‌استر یا ابریشم ترکیب می‌کند. در نهایت نخ به اندازه کافی قوی می‌شود تا بر روی هر دستگاه صنعتی‌ای بافته شود. نخ ژاکارد از نخ‌های معمولی که امروزه برای تولید پارچه استفاده می‌شود، غیر قابل تشخیص است.



نیم‌صورت آنتون سامانی





Slide 2

What is LibreOffice?



Do more – easily, quickly

LibreOffice is a powerful office suite; its clean interface and powerful tools let you unleash your creativity and grow your productivity. LibreOffice embeds several applications that make it the most powerful Free & Open Source Office suite on the market: Writer, the word processor, Calc, the spreadsheet application, Impress, the presentation engine, Draw, our drawing and flowcharting application, Base, our database and database frontend, and Math for editing mathematics.

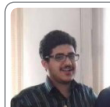
Finally, documents that look good

Your documents will look professional and clean, regardless of their purpose: a letter, a master thesis, a brochure, financial reports, marketing presentations, technical drawings and diagrams.



لیبره آفیس برای اندروید

قدرت و آزادی را در اندروید خود احساس کنید



مترجم
احسان تری

باشد؛ هم‌چنین قادر نیست مانند گوگل درایو پشتیبانی از اسناد را در فضای ابری مختص به خود یا ابزاری ثالث، نگه دارد.

یافتن پرونده‌ها در این نرم‌افزار نیاز به پیمایش در ساختار سامانه پرونده‌ی کم‌تر شهودی اندروید دارد و ممکن است خیلی از افراد پس از برخورد با چنین تجربه‌ای، عطای این برنامه را به لقایش ببخشند. ولی کسانی که می‌مانند، یک مشاهده‌گر اسناد بسیار باکیفیت را می‌یابند، به همراه ویرایشگری که گرچه به اندازه‌ی کافی پایدار است، ولی برای استفاده روی نمایشگر کوچک تلفن خیلی مناسب نیست.

افراد زیادی هستند که از ارائه چنین ابزارهایی خوشحال شوند، از جمله کسانی که علاقه دارند از ابزارهای آزاد بیش‌تری در تلفن همراه خود استفاده کنند. دیگر افراد هم که نگاه جدی‌تری به مقوله‌ی مجموعه‌های اداری دارند، باید به این نرم‌افزار نگاه ویژه‌ای داشته باشند. ■



نکته‌ای که بیش از همه این محصول را از رقبای خود متمایز می‌کند این است که لیبره آفیس روی اندروید از همان موتور پردازشی استفاده می‌کند که روی نرم‌افزار اصلی وجود دارد

نکته‌ای که بیش از همه این محصول را از رقبای خود متمایز می‌کند این است که لیبره آفیس روی اندروید از همان موتور پردازشی استفاده می‌کند که روی نرم‌افزار اصلی وجود دارد و برای مثال می‌توان آن را نخستین مجموعه‌ای دانست که می‌تواند فرمول‌های ریاضی را در اسناد به درستی نشان دهد. با این حال، این نگارش هنوز یک مجموعه‌ی نرم‌افزارهای اداری کامل روی تلفن‌های هوشمند نیست و از طرف این بنیاد به عنوان یک نرم‌افزار مشاهده‌ی اسناد با امکانات ابتدایی ویرایش، مانند تغییر واژگان، تغییر رنگ و نوع قلم و معرفی شده است.

مشاهده‌ی اسناد در این نگارش از لیبره آفیس اندروید احساس آزمایشی بودن را به کاربر منتقل می‌کند و به نظر می‌رسد که هنوز جای کار زیادی دارد. مثلاً به‌طور کامل نمی‌تواند بین اسناد شما در حافظه داخلی و خارجی تفاوتی قائل شود و یا با خدمات ابری هماهنگ شده و به اسناد داخل آن‌ها دسترسی داشته

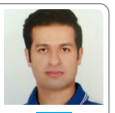
لیبره آفیس یکی از نرم‌افزارهای آزاد معروف نزد کاربران گنو/لینوکس است. این برنامه از نرم‌افزار آزاد اوپن آفیس انشعاب یافته که توسط اوراکل خریداری شد و پس از آن به دلایلی نامعلوم به بنیاد آپاچی منتقل و پروانه انتشار آن نیز به پروانه آپاچی تغییر یافت. لیبره آفیس زمانی از اوپن آفیس منشعب شد که شرکت سان توسط شرکت اوراکل خریداری شده بود. پس از این انشعاب، بنیادی به نام بنیاد مستندات ایجاد شد که از کارمندان و توسعه‌دهندگان سابق شرکت سان در بخش اوپن آفیس بودند.

بنیاد مستندات به تازگی لیبره آفیس جدید را برای اندروید نیز عرضه کرد. این برنامه به کاربران اندروید اجازه می‌دهد اسنادشان را خوانده و ویرایش کنند. پیش از این نیز نگارش‌هایی غیررسمی از اوپن آفیس یا حتی لیبره آفیس برای اندروید منتشر شده بود و در نگارش‌های نخستین برنامه‌ی رسمی نیز، تنها امکان مشاهده‌ی اسناد وجود داشت.

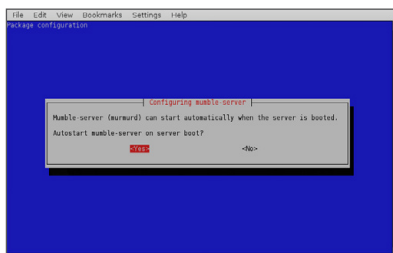




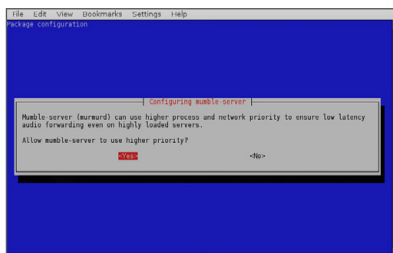
مکالمه صوتی از طریق مامبل



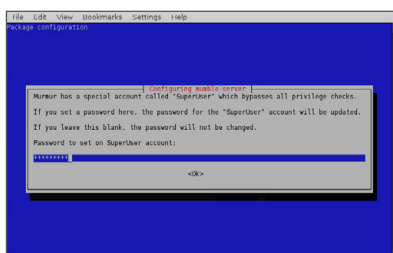
مهدی حمیدانی
نویسنده



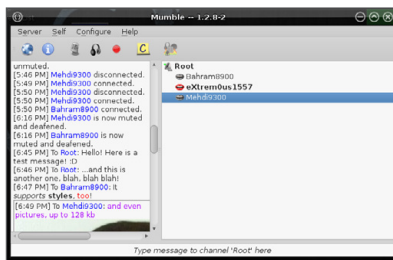
(تصویر ۱)



(تصویر ۲)



(تصویر ۳)



(تصویری از محیط مامبل)

تبریک! شما به سادگی یک نسخه سرور از Mumble را نصب کردید!

پس از نصب، نیاز است پیکربندی‌های نخستین را انجام دهیم. پس دستور زیر را وارد می‌کنیم:

```
$ sudo dpkg-reconfigure mumble-server
```

در بخش نخست از کاربر پرسیده می‌شود که می‌خواهد Daemon (دیمن) مربوط به کارساز مامبل در هر بار راه‌اندازی اجرا شود که می‌پذیریم. (تصویر ۱) در بخش دوم، درباره اولویت اجرای فرایند دیمن مربوط به مامبل از کاربر پرسیده می‌شود. هر چه اولویت این فرایند بالاتر باشد، سیستم‌عامل بازه‌های زمانی بیش‌تری را به آن تخصیص می‌دهد که امکان ایجاد وقفه بین مکالمات پایین‌تر می‌رود. (تصویر ۲) و در نهایت در بخش سوم، گذرواژه کاربر ارشد پرسیده می‌شود. کاربر ارشد قادر است با استفاده از نام کاربری و این گذرواژه وارد تنظیمات کارساز مامبل شده و آن را تنظیم کند. (تصویر ۳)

تنظیمات پیشرفته کارساز مامبل در پرونده `/etc/mumble-server.ini` قرار دارد. از آن تنظیمات می‌توان به پورت در حال شنود (که به طور پیش‌گزیده مقدار ۶۴۷۳۸ را دارد)، پیغام خوش‌آمدگویی، پهنای باند، سیاست جلوگیری از حملات Brute Force، تعداد بیشینه کاربران هم‌زمان، بیشینه طول پیام گپ، اندازه تصاویر ارسالی و مخفی شدن در شبکه اشاره نمود.

نصب کلاینت‌ها:

کلاینت‌های مختلفی برای اتصال به یک کارساز مامبل نوشته شده‌اند. علاوه بر کلاینت‌های مربوط به رایانه‌های رومیزی، کلاینت‌های مامبل برای اکثر بن‌سازهای قابل حمل نیز نوشته شده است. در توزیع‌هایی مانند دیبیا و اوبونتو، نصب این کلاینت‌ها به سادگی اجرای دستور زیر است:

```
$ sudo apt-get install mumble
```

مامبل علاوه بر انتقال صدا، قابلیت گپ زدن از طریق تگ‌های HTML و انتقال عکس را نیز فراهم می‌کند. علاوه بر این، مامبل یک API مخصوص دارد که قابلیتی نظیر Mumble Overlay را ارائه می‌کند. این قابلیت که قابل ترکیب با DirectX و OpenGL است باعث می‌شود که در یک بازی مانند دنیای وارکرفت یا دوتا ۲، صدای مکالمه‌ی دوست خود را از همان جهتی بشنوید که شخصیت او در بازی قرار دارد. قابلیت استفاده موثر از شبکه تور، در کنار رمزنگاری داده‌های مربوط به صدا، امنیتی مثال‌زدنی به وجود می‌آورد. هم‌چنین مامبل سه حالت استفاده را پشتیبانی می‌کند: حالت فعال شدن دستی، حالت فعال شدن با اوج گرفتن صدا و حالت فعالیت دائم. **توضیح:** مامبل از دو بخش تشکیل شده، یک بخش سرور است که در مخازن توزیع‌های بر پایه دیبیا به طور معمول با نام بسته `mumble-server` شناخته می‌شود و بخش کلاینت با همان نام بسته `mumble` شناخته می‌شود. در ادامه به نصب و پیکربندی `mumble-server` می‌پردازیم. `mumble-server` که نام دیگر آن `Murmur` است، قابلیت دسترسی شخصی یا عمومی را به کاربران می‌دهد.

نصب mumble-server:

مقدار حافظه ۵۱۲ مگابایت برای راه‌اندازی نسخه سرور کافی است، آن قدر که ۵۰ مکالمه هم‌زمان بتوانند با کیفیت عالی صورت گیرند. همان طور که اشاره شد، مامبل یک ابزار مستقل از بستر است. به لطف توقع سخت‌افزاری پایین، من توانستم کارساز را روی برد رزبری‌پای به راحتی نصب و استفاده کنم.

نصب و پیکربندی mumble-server در یک توزیع مبتنی بر دیبیا/گنو/لینوکس:

دستور زیر را در پسته گنو/لینوکس خود وارد کنید:

```
$ sudo apt-get install mumble-server
```





نرم افزار آزاد، به داد پرونده‌های پاک شده می‌رسد

تست دیسک، ابزار بازگردانی فایل‌های پاک شده

اکثر افرادی که سر و کارشان با رایانه است، به نحوی خاطره‌ای از پاک شدن ناخواسته پرونده‌های مورد نیازشان را دارند. در این میان بعضی از این افراد سعی در بازیابی داده‌های پاک شده (اصطلاحاً ریکاوری) نموده‌اند که با مشکلاتی مانند عدم بازیابی کامل و... روبرو شده‌اند. ولی کم‌تر جایی سخن از نرم‌افزار آزاد تست دیسک که ابزاری قدرتمند و قابل اجرا بر روی سیستم عامل‌های متعددی است برده شده. در این راهنما قصد داریم شما را به صورت عملی با سناریوی بازگردانی داده‌های پاک شده از روی حافظه‌های جانبی مانند دیسک سخت، فلش، کارت‌های حافظه و آشنا کنیم.

معرفی تست دیسک:



تست دیسک (به انگلیسی TestDisk) یک نرم‌افزار آزاد برای بازگردانی داده‌های پاک شده است. این برنامه عمدتاً برای بازگردانی پارتیشن‌های مفقود شده است و یا برای قابل راه‌اندازی (بوتیل) کردن دیسک‌هایی که این گونه نیستند طراحی شده. این اشکالات می‌توانند به دلایل مختلفی از جمله خطاهای نرم‌افزاری، ویروس‌ها و یا خطاهای انسانی (همانند پاک کردن جدول پارتیشن به صورت تصادفی) به وجود آیند.

با استفاده از تست دیسک، می‌توان اطلاعات دقیق و مشروحی را در مورد دیسک‌های خراب به دست آورد و سپس آن اطلاعات را به منظور تحلیل و بررسی بیشتر برای یک متخصص ارسال کرد. تست دیسک روی سیستم‌عامل‌های مختلفی از جمله داس، ویندوز، مک اواس ده، گنولینوکس، فری‌بی‌اس‌دی، اوپن‌بی‌اس‌دی، نت‌بی‌اس‌دی و اجرا می‌شود. تست دیسک هم‌چنین از بسیاری از طرح‌بندی‌های رایج جدول پارتیشن پشتیبانی می‌کند. از جمله طرح‌بندی‌های ام‌بی‌آر، جی‌پی‌تی و تعدادی دیگر. تست دیسک می‌تواند پارتیشن‌های حذف شده را بازیابی کند، جدول پارتیشن را بازسازی کند و رکورد راه‌اندازی اصلی درایو را بازنویسی کند. وقتی که پرونده‌ای از روی دیسک حذف می‌شود، فهرست خوشه‌هایی که پرونده اشغال کرده بود هم پاک شده و پرونده‌های دیگر می‌توانند از این سکتورها برای ذخیره اطلاعات خود استفاده کنند. اگر پرونده تکه‌تکه نشده باشد و خوشه‌ها توسط پرونده‌های دیگر استفاده نشده باشند، تست دیسک قادر است پرونده مورد نظر را بازیابی کند. این برنامه از فایل سیستم‌های متعددی پشتیبانی می‌کند و تقریباً می‌توان گفت تست دیسک از همه فایل سیستم‌های رایج در سیستم‌عامل‌های مختلف پشتیبانی می‌کند. در جولای ۲۰۰۸، تست دیسک و فوتورک (در قالب یک برنامه) بیش از ۱۵۰۰۰۰ بار فقط از پایگاه وب اصلی دریافت شدند.

این برنامه‌ها به قدری محبوب هستند که برخی از دیسک‌های زنده گنولینوکس از جمله ناپیکس، جی‌پارتد، پارتد میک و به صورت پیش‌گزیده به همراه تست دیسک عرضه می‌شوند. [نقل شده از ویکی‌پدیا]

۳ گام به گام بازگردانی پرونده(های) پاک شده از حافظه:

پیش از شروع به انجام هر کاری، باید یادآور شویم که تست دیسک دربردارنده قابلیت‌های بسیاری است که بازگردانی پرونده‌های پاک شده یکی از آن‌ها است. تست دیسک با توجه به نوع قابلیت، از فایل سیستم‌های مختلفی پشتیبانی می‌کند که فایل سیستم‌های پشتیبانی شده جهت قابلیت بازگردانی پرونده‌های پاک شده شامل FAT، NTFS، exFAT و خانواده Ext است. بنابراین خیال ویندوزی‌ها از بابت پشتیبانی از کلیه فایل سیستم‌های حافظه‌های جانبی‌شان راحت باشد، ولی کاربران دیگر سیستم‌عامل‌ها حتماً پیش از استفاده از این قابلیت به نوع فایل سیستم حافظه جانبی مورد نظر دقت داشته باشند.

۱.۳ بازگیری / نصب:

ابتدا با مراجعه به صفحه دریافت تست دیسک پرونده اجرایی برنامه را دریافت کنید. خوشبختانه تست دیسک دارای نگارش‌های ویندوز، گنولینوکس، مک، داس و است.

اگر کاربر اوبونتو یا دبیان گنولینوکس هستید می‌توانید با استفاده از رابط گرافیکی مدیریت بسته‌های نرم‌افزاری یا اجرای دستور زیر در خط فرمان اقدام به نصب این نرم‌افزار کنید:

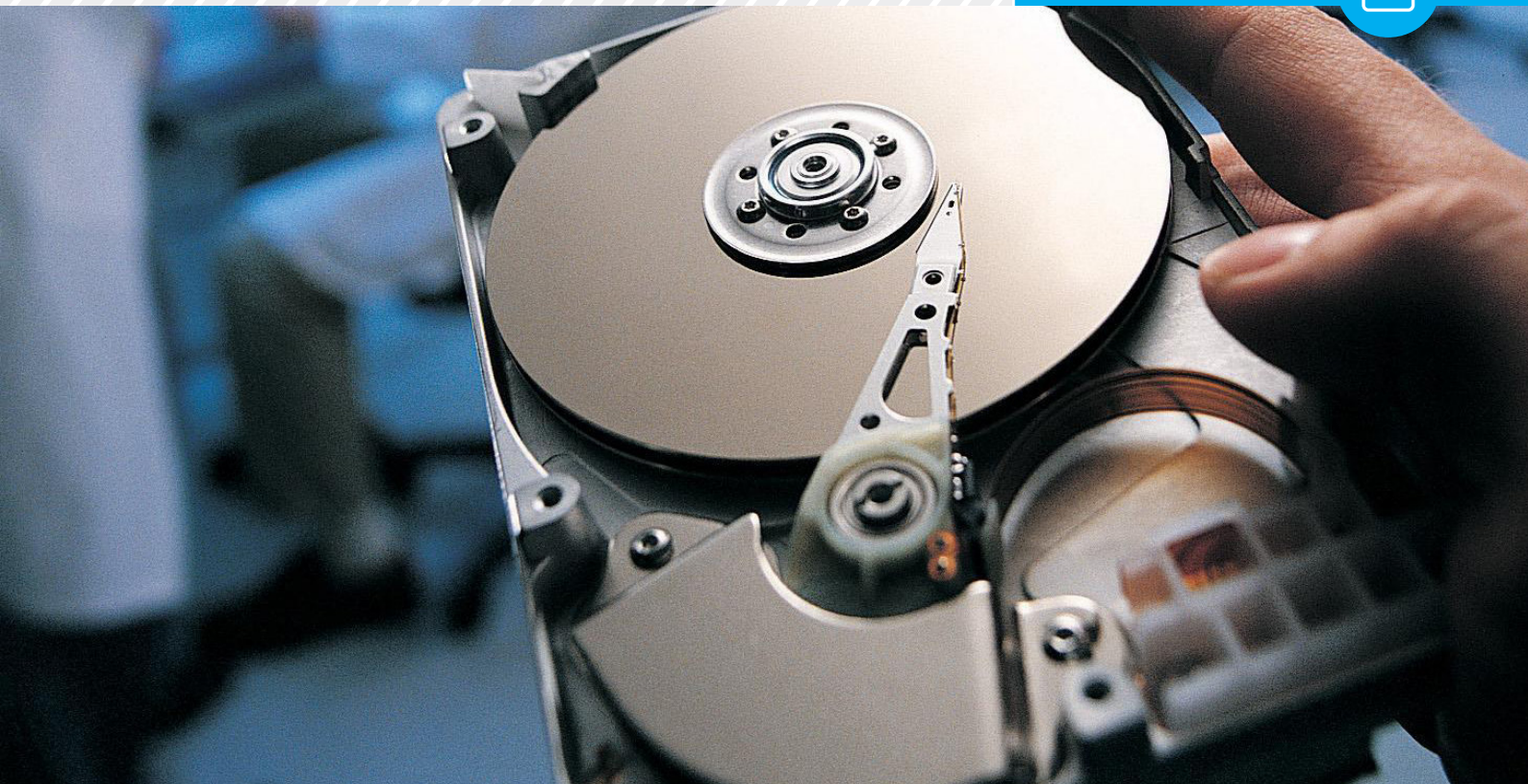
```
sudo apt-get update; sudo apt-get install testdisk
```

هم‌چنین اگر کاربر سنت‌اواس یا دیگر توزیع‌های گنولینوکس مبتنی بر فدورا هستید، می‌توانید با استفاده از رابط گرافیکی آن توزیع گنولینوکس یا اجرای دستور زیر در خط فرمان اقدام به نصب این نرم‌افزار کنید:

```
su; yum update; yum install testdisk
```



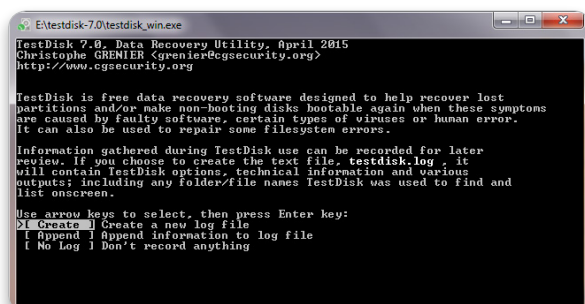
بهنام توکلی
نویسنده



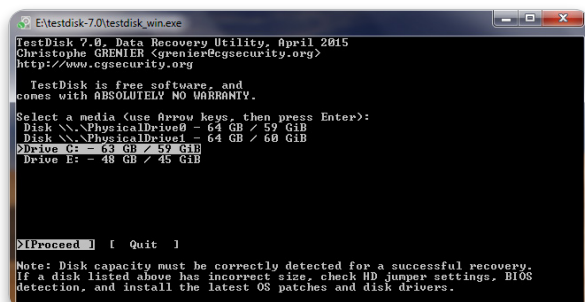
اجرا و بازگردانی پرونده(ها):

ریشه می‌توانید وارد محیط برنامه شوید (جهت اجرا به صورت کاربر ریشه در توزیع اوبونتو کافی است از دستور `sudo testdisk` استفاده کنید). توجه داشته باشید که `testdisk` دارای رابط گرافیکی نیست، ولی کار با رابط متنی آن بسیار ساده و در همه سیستم‌عامل‌ها به یک نحو است. پنجره ای مطابق با تصویر ۱.

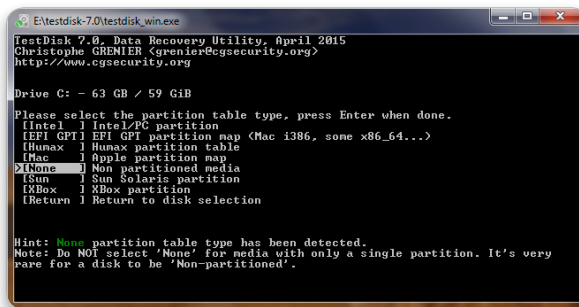
اگر از نگارش ویندوزی استفاده می‌کنید، باید پرونده `testdisk_win.exe` را اجرا کنید. اگر کاربر گنو/لینوکس هستید با زدن دستور `testdisk` در محیط خط فرمان به عنوان کاربر



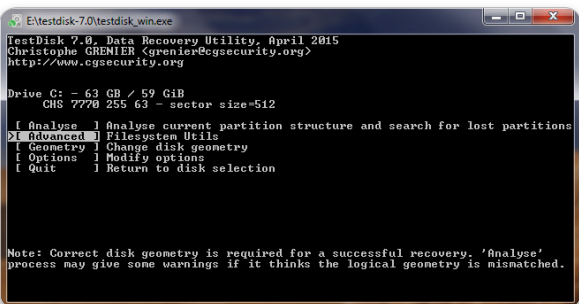
پس از اجرای برنامه با پنجره روبه‌رو مواجه می‌شوید. اگر تمایل دارید که کلیه فعالیت‌هایی که انجام می‌شود در یک پرونده متنی ثبت شود باید گزینه نخست را جهت ایجاد یک پرونده جدید یا گزینه دوم را جهت اضافه شدن گزارش‌های جدید به پرونده‌ای که از پیش ایجاد شده است انتخاب کنید. اما اگر نیازی به این گزارش ندارید یا فایل سیستم حافظه جانبی شما به صورت فقط خواندنی درآمده است، باید گزینه آخر یعنی `No Log` را برگزینید.



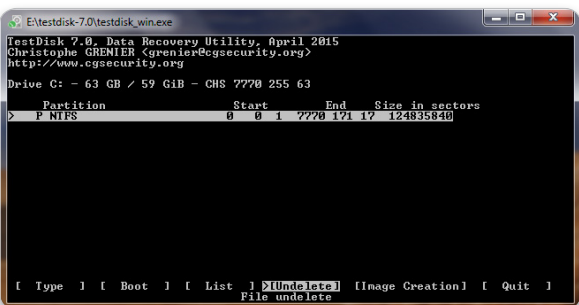
در مرحله بعدی از شما حافظه جانبی‌ای که قصد انجام فعالیت روی آن را دارید، پرسیده می‌شود و باید با کلیدهای بالا/پایین آن را برگزینید.



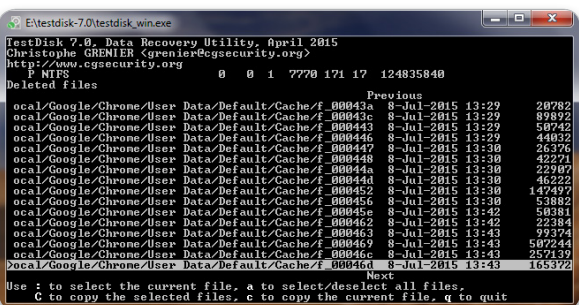
در مرحله بعد، تست دیسک نوع جدول پارتیشن حافظه جانبی را می‌پرسد که باید گزینه مناسب را انتخاب کنید. در این جا گزینه None که پیش گزیده است را برمی‌گزینیم.



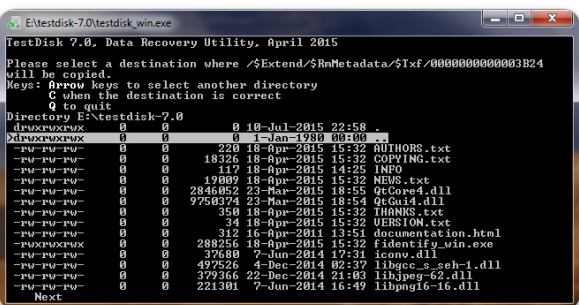
در مرحله بعدی باید قابلیت مورد نظر جهت انجام روی حافظه انتخاب شده را انتخاب کرد که جهت بازگردانی پرونده‌های پاک شده باید گزینه Advanced را برگزید.



سپس فهرستی از پارتیشن‌های حافظه جانبی انتخاب شده نمایش داده می‌شود که باید پارتیشن مورد نظر را با استفاده از کلیدهای بالا/پایین انتخاب کرده و سپس با کلیدهای چپ/راست گزینه Undelete را از گزینه‌های عملیاتی پایین برگزیده و اجرا کنید.



سپس فهرستی از پرونده‌های پاک شده روی پارتیشن مورد نظر نمایش داده می‌شود که با استفاده از دکمه‌های بالا/پایین می‌توانید ادامه فهرست را مشاهده کنید. جهت انتخاب یک پرونده خاص، روی آن پرونده رفته و علامت «» را بنویسید. هم‌چنین برای انتخاب کلیه پرونده‌های نمایش داده شده می‌توانید حرف «a» را بنویسید. موارد انتخاب شده به صورت رنگی نمایش داده خواهند شد.

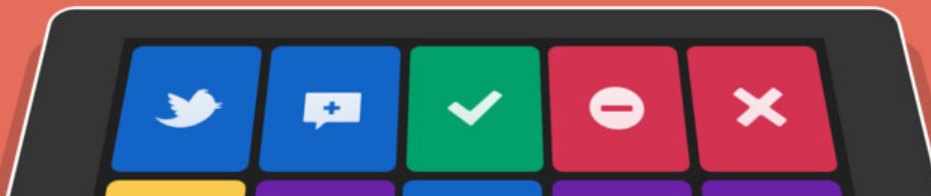


سپس برای رونوشت گرفتن از پرونده‌ای که روی آن قرار دارید، می‌توانید حرف C (سی بزرگ) را نوشته یا برای رونوشت گرفتن از پرونده‌های انتخاب شده (توسط علامت: یا نوشتن حرف a) می‌توانید حرف C (سی کوچک) را بنویسید. در مرحله بعدی باید مشخص کنید که پرونده‌های انتخاب و رونوشت شده در کجا قرار داده شوند. در این مرحله باید مسیر مورد نظر را مشخص کنید. لازم به ذکر است که عبارت «» (یک نقطه) نمایانگر مسیر فعلی و عبارت «» (دو نقطه کنار هم) نمایانگر شاخه والد فولدر (پیشین) است. سرانجام هنگامی که به مسیر مورد نظر رفتید با نوشتن حرف C (سی بزرگ) پرونده‌های انتخاب شده شروع به رونوشت شدن در آن مسیر می‌کنند و بدین صورت کار بازیابی پرونده‌های پاک شده به پایان می‌رسد.





Search on DuckDuckGo...



با خیال راحت در وب جستجو کنید

۷ برتری جستوجوگر داکداکگو نسبت به گوگل و بینگ

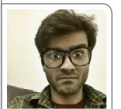
بنابراین، داکداکگو بسیار جذاب است. در بسیاری از موارد، همانند موتورهای جستوجوی دیگری مثل گوگل و بینگ است، ولی ترفندهایی در آستین دارد که آن را از رقبایش جدا می‌کند.

❶ محرمانگی و امنیت

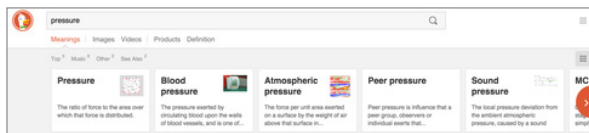
این بزرگ‌ترین نقطه تمایز داکداکگو است. برخلاف بقیه موتورهای جستوجو معروف، داکداکگو هیچ گونه اطلاعات شخصی را جمع نمی‌کند. همین مسئله باعث می‌شود که به شکل پیش‌گزیده، هیچ تاریخچه‌ای از جستوجو، نشانی‌های IP و کوکی‌ها را ذخیره نکند. علاوه بر این، برای امنیت بیش‌تر، داکداکگو به شکل خودکار نگارش‌های رمز شده (HTTPS) را نمایش می‌دهد. البته به همین‌جا محدود نمی‌شود و می‌توانید با تور از داکداکگو استفاده کنید. برای اطلاعات بیشتر سری به این‌جا بزنید:

<https://duckduckgo.com/privacy>

❷ داکداکگو (duckduckgo) یک موتور جستوجوی وب است که در عملکرد خود تا حد زیادی بر داده‌های جمع‌سپاری‌شده هم‌چون ویکی‌پدیا تکیه دارد. یکی از نکات مهم این موتور جستوجو (به ادعای خودش)، عدم ردگیری کاربران است. این موتور جستوجوی وب را می‌توان حاصل نگرانی‌ها پیرامون حریم خصوصی افراد دانست. در صفحه سیاست محرمانگی این پایگاه وب آمده است که هیچ گونه اطلاعات شخصی‌ای را جمع‌آوری نکرده و به اشتراک نمی‌گذارد. بر خلاف دیگر موتورهای جستوجو در سیاست داکداکگو آمده است که تاریخچه جستوجوهای کاربران را ذخیره نمی‌کند. از ویژگی‌های جستوجو در این موتور، می‌توان به شیوه جستوجوی واژه‌های دارای ابهام در آن اشاره کرد. بر خلاف برخی از شیوه‌های دیگر جستوجو که بر پایه بهترین نتایج در صفحه نخست کار می‌کنند، داکداکگو تلاش می‌کند تمام معانی مختلف یک واژه تفسیرپذیر را در صفحه نخست بگنجانند تا نتایج این صفحه به یک معنی خاص سوگیری نداشته باشد. افزون بر این، کاربر را راهنمایی می‌کند تا مقصود خود را بهتر بیان کند.

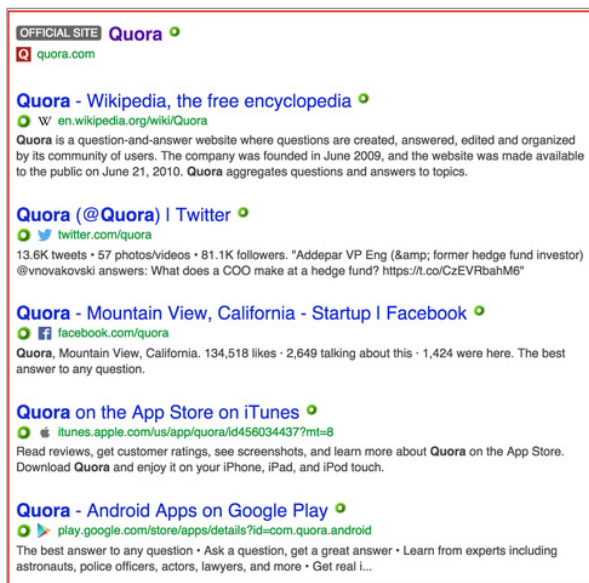


نویسنده:
پیمان شایر

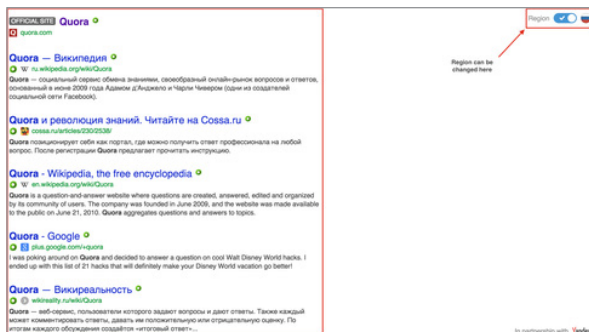


جستوجوی منطقه‌ای

کاربران داک‌داک‌گو می‌توانند نتایج مرتبط با یک منطقه مکانی را با تغییر منطقه به‌دست آورند. به عنوان مثال من «Quora» را در حالی که منطقه ایالات متحده آمریکا بود، جستوجو کردم و به نتایج زیر رسید.

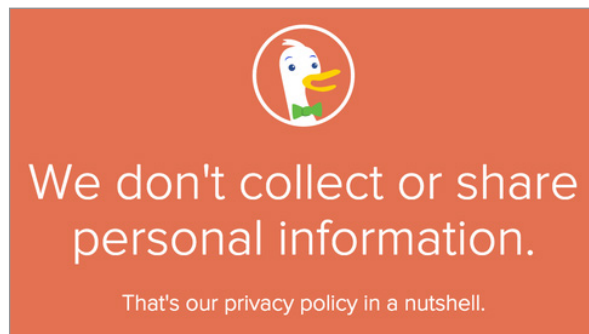


حالا اگر منطقه را به روسیه تغییر بدهم، نتایج به شکل زیر خواهند شد. پس با یک کلیک، می‌توان نتایج را برای کاربری با منطقه دیگر ایجاد کرد.



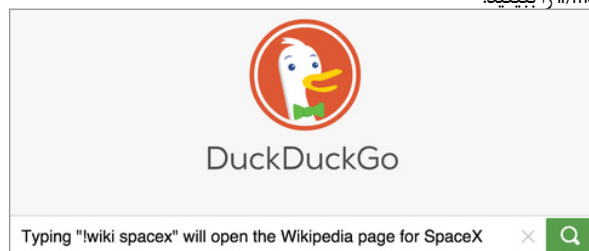
جستوجوی بدون تبلیغات

با این‌که به شکل پیش فرض در داک‌داک‌گو تبلیغات نشان داده می‌شود، اما می‌توان آن را با یک کلیک متوقف کرد. برای این کار به مسیر تنظیمات پیشرفته عمومی تبلیغات بروید و خاموش کنید. این‌که بتوانیم به عنوان یک کاربر امکان کنترل داشته باشیم، اتفاق مبارک و خوبی است، در زمانی که هیچ‌کدام از جستوجوگرهای گردن کلفت هم این ویژگی را ندارند. البته من باید توضیح بدهم که این تبلیغات بدون جاسوسی، مرتبط و تمیز هستند. به همین دلیل من غیرفعالشان نمی‌کنم، شاید با این کار بتوانم از شرکت مربوط به آن پشتیبانی کنم.



بنگ

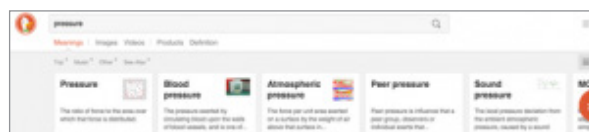
این یکی از ویژگی‌های برجسته شده داک‌داک‌گو است که به شما امکان می‌دهد مستقیماً جستوجو را روی پایگاه‌های وب دیگر انجام دهید. مثلاً من می‌توانم بنویسم «Gmail quora» و داک‌داک‌گو مستقیماً برای من «quora» را در حساب gmail من جستوجو کند. یا می‌توانم بنویسم «amazon ps4 controller» و داک‌داک‌گو مستقیماً به این صفحه می‌رود: <http://slmd.ir/lz> هزاران پایگاه وب وجود دارد که برایشان میان‌برهای بنگ وجود دارد. اگر نتوانستید میان‌بر بنگ را برای پایگاه وبی پیدا کنید، خوب بسازیدش. آن‌طور که داک‌داک‌گو گفته: «هشدار: استفاده از بنگ روش جستوجوی شما را برای همیشه تغییر می‌دهد. وقت آزاد شما را بیشتر می‌کند و امید به زندگی بیش‌تری پیدا می‌کنید.» واقعاً این جمله درست است. یک بار که به سراغ بنگ بروید متعادلش می‌شوید و دیگر تمام. برای اطلاعات بیشتر <http://slmd.ir/mo> را ببینید.



صفحات دسته‌بندی

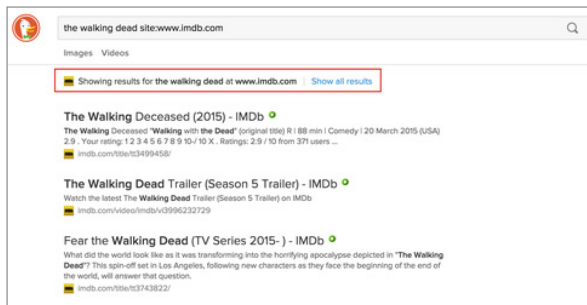
این یکی از ویژگی‌های مورد علاقه من در داک‌داک‌گو است. اساساً می‌توانید صفحات دسته‌بندی را صدا بزیند تا جواب سریعی را در دسته بندی خاصی پیدا کنید. این ویژگی با آماده کردن جواب‌های سریع، متفاوت کوتاه و گزینه‌های معنی‌دار مربوط به عبارتی که جستوجو کرده‌اید، در زمانتان صرفه جویی می‌کند. مثلاً زمانی که برای «pressure» جستوجو کردم، دسته‌بندی‌های زیر را به من نشان داد.

به طور مشابه زمانی که در مورد «south park characters» جستوجو کردم، به



این دسته‌ها رسیدم.

علاوه بر این که پاسخ‌های فوری را ایجاد می‌کند، پرونده ارائه‌ای که پیدا کرده بود نیز خیلی خوب بود.



ویژگی‌های دیگر

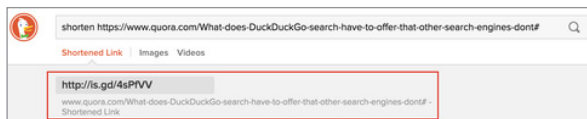
در ادامه برخی ویژگی‌های جذاب و کوچک داک‌داک‌گو را آورده‌ام.

صفحه نتایج قابل لغزیدن:

پیش‌گزیده داک‌داک‌گو این است که نتایج را صفحه بندی نکند، پس خیلی ساده می‌توان به پایین لغزید و نتایج بیشتری را پیدا کرد. البته اگر نوع صفحه بندی شده را دوست داشته باشید، امکان فعال کردنش وجود دارد.

کوتاه کردن و بسط دادن نسانی‌ها:

می‌توانید با داک‌داک‌گو در یک ثانیه نشانی هر آدرسی را کوتاه کرده یا بسط یافته‌اش را پیدا کنید. برای کوتاه کردن تنها کافی است عبارت کلیدی «short» را پیش از آدرس بنویسید و داک‌داک‌گو کوتاه‌شده آن را ایجاد می‌کند.



به طور مشابه برای بسط دادن یک پیوند از کلیدواژه «expand» استفاده کنید. لطف این کار در این است که بدون باز کردن پیوند می‌توانید متوجه شوید که نشانی به کجا می‌رود.

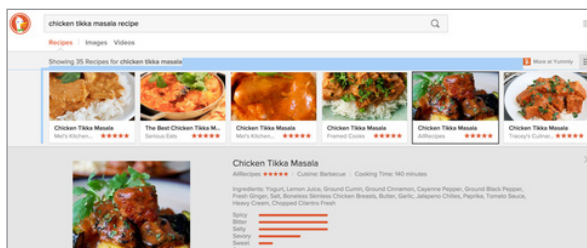
حسابگر پرداخت:

داک‌داک‌گو می‌تواند به شما کمک کند تا پرداخت ماهانه وام خود را به آسانی حساب کنید. برای این منظور از کلیدواژه «loan» استفاده کنید. مثل تصویر زیر:



یابنده دستور آشپزی:

می‌توانید دستور پخت غذاهای مختلف را با به کار بردن «recipe» در جست‌وجو، پیدا کنید. داک‌داک‌گو دستورات مختلف را در قالبی زیبا به شما نشان می‌دهد. درست مثل تصویر زیر:



Advertisements

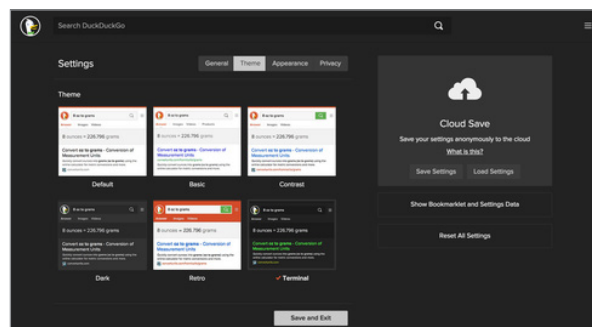
If you still want to support us, help spread DuckDuckGo

Off

شخصی سازی فراوان

داک‌داک‌گو به کاربرانش امکان شخصی کردن ظاهر را می‌دهد. مثلاً کاربران می‌توانند قالب، قلم‌ها و رنگ‌ها را عوض کنند. تصویر زیر نمایش نتایج با قالب «خط فرمان» است، چیزی شبیه به حالت شب.

علاوه بر این، کاربران می‌توانند زبان را تغییر دهند، میان‌برهای صفحه‌کلید را عوض کنند و نشان پایگاه‌های وب در نتایج را ببینند. آخرین و بهترین امکان شخصی‌سازی، امکان ذخیره تنظیمات به صورت ناشناس است.



جست‌وجوی داخل سایت

اگرچه گوگل و بینگ هر دو امکان جست‌وجو در داخل وب سایت را دارا هستند، ولی پیاده سازی داک‌داک‌گو ساده‌تر و کاربر پسندتر است. علاوه بر این، به ازای هر نتیجه جست‌وجو، گزینه «نتایج بیش‌تر» را نمایش می‌دهد که با کلیک کردن رویش، داک‌داک‌گو تنها نتایج آن پایگاه وب خاص را نمایش می‌دهد. می‌توانم روی هر نتیجه‌ای بروم، آن‌گاه گزینه «نتخاب‌های بیش‌تر» نمایش

The Walking Dead - AMC

The official site for AMC's epic series The Walking Dead. Games, making-of videos, plus episode & character guides

amctv.com/shows/the-walking-dead

The Walking Dead (TV Series 2010-) - IMDb

Created by Frank Darabont. With Andrew Lincoln, Jon Bernthal, Sarah Wayne Callies, Steven Yeun. Sheriff's Deputy Rick Grimes leads a group of survivors in a world overrun by zombies.

imdb.com/title/tt152021/

The Walking Dead (TV series) - Wikipedia, the free encyclopedia

The Walking Dead is an American horror drama television series developed by Frank Darabont, based on the comic book series of the same name by Robert Kirkman, Tony Moore, and Charlie Adlard.

en.wikipedia.org/wiki/The_Walking_Dead_TV_series

The Walking Dead - TV.com

The Walking Dead: Watch full length episodes & video clips. Read the latest The Walking Dead episode guides & recaps, fan reviews, news, and much more.

tv.com/shows/the-walking-dead/

The Walking Dead - The Official Site from Robert Kirkman

The Walking Dead official website from Robert Kirkman's Skybound Entertainment. Comic, TV Show, Games, and more.

thewalkingdead.com

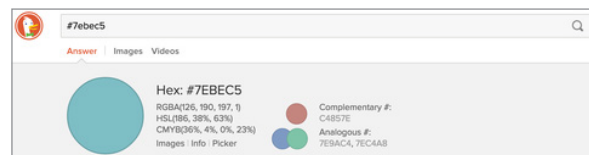
داده می‌شود. در این مثال من روی «IMDb» رفته‌ام.

همان طور که در شکل مشخص است، با کلیک کردن روی «نتایج بیش‌تر» فقط نتایج «IMDb» نمایش داده می‌شود.

خیلی ساده شد، این طور نیست؟

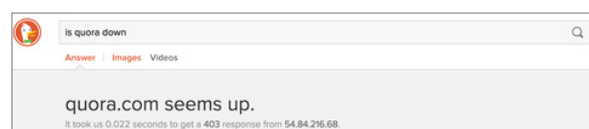
3 یابنده رنگ:

با داشتن کد مبنای شانزده هر رنگی، می‌توانید آن رنگ را پیدا کنید. برای مثال:



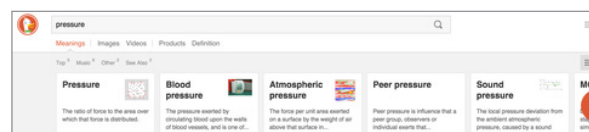
3 بررسی بالا بودن پایگاه وب:

داک‌داک‌گو در جعبه جست‌وجوی خود به آسانی امکان بررسی کار کردن یک پایگاه وب را می‌دهد.



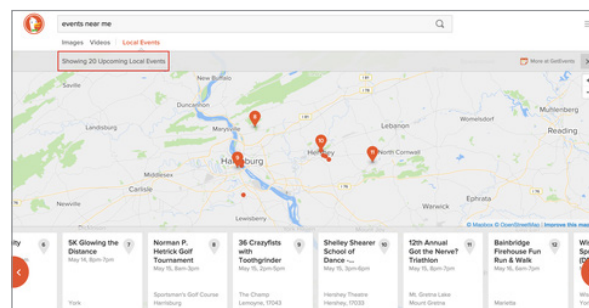
3 یابنده جایگزین:

می‌توانید جایگزین‌هایی را برای برنامه‌ها، پایگاه‌های وب، نرم‌افزارها یا چیزهای دیگر، با استفاده از داک‌داک‌گو پیدا کنید.



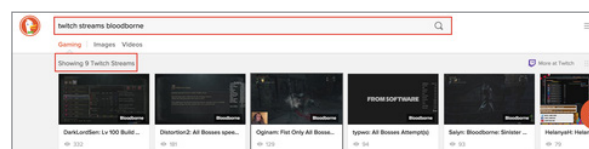
3 یابنده رخدادهای:

با پرسش «event near me» در داک‌داک‌گو می‌توانید رخدادهای آینده موجود در اطراف خود را ببینید.



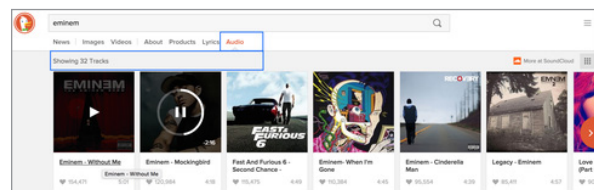
3 جریان‌های زنده Twitch:

اگر دوست‌دار تماشای جریان‌های زنده بازی ویدیویی بر روی Twitch باشید، داک‌داک‌گو این کار را در صفحه جست‌وجوی خود ساده کرده است.



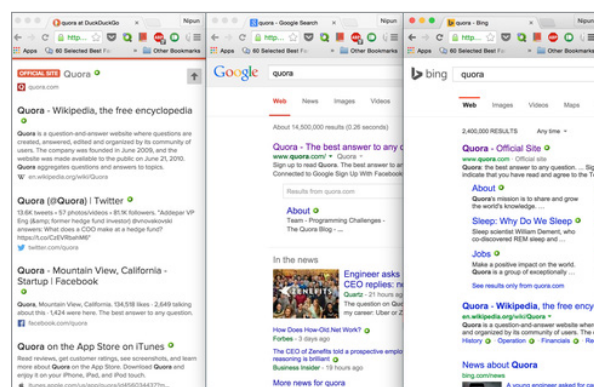
3 موسیقی:

در داک‌داک‌گو می‌توانید به آهنگ‌های هنرمندان در صفحه نتایج جست‌وجو گوش دهید. مثلاً زمانی که «Eminem» را جست‌وجو کنیم، می‌توانم بدون ترک کردن صفحه جست‌وجو، معروف‌ترین آهنگ‌هایش را گوش بدهم.



3 طراحی واکنش‌گرا:

این مورد بیشتر به تجربه کاربری مربوط است تا به ویژگی‌ها، ولی صفحه نتایج داک‌داک‌گو واکنش‌گرا است (خود را با اندازه صفحه تطبیق می‌دهد). در حالی که گوگل و بینگ این‌طور نیستند. به مقایسه زیر دقت کنید.



برای آشنایی با ویژگی‌های جذاب دیگر به صفحه ویژگی‌ها بروید. این صفحه شامل تعداد زیادی از ویژگی‌های کاربردی مثل حل‌کننده مغلوب، مردم در فضا و کمیک‌های xkcd در داک‌داک‌گو است.



خلاصه این‌که بهترین چیز در مورد داک‌داک‌گو، قدرت دادن به کاربر برای کنترل است. علاوه بر این که خیلی جذاب هم هست. فقط داکش کنید

■!(duck it)





معرفی ۸ افزونه کاربردی برای مرورگر گوگل کروم

مرورگر موزیلا فایرفاکس بسیار کندتر از قرار دادن افزونه در فهرست افزونه‌های گوگل کروم در پایگاه اینترنتی فروشگاه افزونه‌های گوگل کروم است، این مسأله به دلیل بررسی‌های بیش‌تری است که توسط تیم توسعه مرورگر فایرفاکس برای افزونه‌ها رخ می‌دهد. با وجود این، برخی توسعه‌دهندگان ممکن است از این بررسی‌های طولانی مدت راضی نباشند. در هر حال، مرورگر کروم با فهرستی از افزونه‌های بزرگ و کوچک همراه است که دسترسی هر یک از این افزونه‌ها به اطلاعات شما در آن مشخص شده است. وقتی قصد نصب یک افزونه را دارید، بهتر است فهرست دسترسی‌های آن افزونه را مطالعه کنید تا اگر مورد اضافی در اختیارات آن وجود داشت، در نصب آن افزونه تجدید نظر کنید. از ویژگی‌ها و برتری‌های مرورگر گوگل کروم نسبت به مرورگر موزیلا فایرفاکس می‌توان به عدم نیاز این مرورگر به راه‌اندازی مجدد مرورگر پس از هر بار نصب افزونه‌ها اشاره داشت. با وجود این که برخی از افزونه‌های مرورگر فایرفاکس نیازی به راه‌اندازی مجدد ندارند؛ ولی متأسفانه خیلی از آن‌ها هنوز هم برای فعال شدن پس از نصب، نیاز به راه‌اندازی مجدد مرورگر دارند.

اکثر کاربران مرورگرهای پیش‌رفته امروزی نیاز دارند تا از افزونه‌های آماده و نوشته شده برای مرورگرها استفاده کنند تا برخی تنظیمات و قابلیت‌هایی که به صورت عادی در مرورگرها وجود ندارد را به مرورگر خود بیفزایند. مرورگرهای امروزی مانند مرورگر گوگل کروم، فایرفاکس و اپرا یا حتی سافاری، این روزها از قابلیت افزودن افزونه‌ها پشتیبانی می‌کنند و به راحتی قادر هستید که در این مرورگرها، افزونه‌های خود را نصب کنید. در این بین اگر کاربر گوگل کروم باشید با فهرست بلند بالایی از افزونه‌های قابل نصب در این مرورگر مواجه خواهید شد که به راحتی قادر هستند اکثر امور ساده و حتی پیش‌رفته مورد نیاز شما در وب‌گردی را تأمین کنند. این ابزارها که به افزونه یا ابزار توسعه مرورگر معروف هستند، می‌توانند قابلیت‌هایی را برای مرورگر کروم و یا کرومیوم فراهم کنند که به صورت پیش‌گزینه در این مرورگر قابل انجام نیستند. در این مقاله به بررسی هشت افزونه کاربردی و مفید در مرورگر کروم خواهیم پرداخت که با نصب آن‌ها، وب‌گردی راحتی را تجربه خواهید کرد با وجود آن که افزودن افزونه جدید به فهرست افزونه‌های موجود در پایگاه اینترنتی

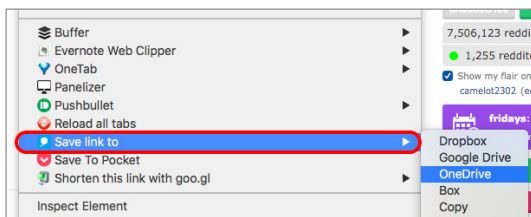
نمایش ویدیو از طریق کروم‌کست Chrome Cast

نوعی دانگل اچ دی ام آی (به انگلیسی: HDMI) ساخت شرکت گوگل است که در ۲۴ ژوئیه ۲۰۱۳ عرضه شد. قیمت کروم‌کست گوگل برابر ۳۵ دلار است که در مقایسه با رقبای خود نظیر فایرتی وی آمازون، دارای قیمتی بسیار ارزان است از کروم‌کست برای پخش خدمات برخط شبکه‌هایی چون نت‌فلیکس و سرویس‌های گوگل مانند گوگل پلی و یوتیوب و پرونده‌های رسانه‌ای که روی دستگاه‌های اندرویدی قرار دارند استفاده می‌شود. کروم‌کست را می‌توان یک جایگزین ارزان قیمت برای ابزارهای نمایش ویدیو و چندرسانه‌ای از طریق تلویزیون دانست که رقیب اصلی آن که توسط اپل ایجاد شده است، اپل تی وی نام دارد. در مرورگر کروم، افزونه‌ای قرار دارد که با نصب آن در این مرورگر، هنگام مشاهده یک ویدیو یا شنیدن موسیقی از طریق اینترنت، می‌توانید با استفاده از یک دکمه، آن را برای ادامه پخش به کروم‌کست ارسال کنید. در این هنگام اگر کروم‌کست شما از طریق اچ دی ام آی به تلویزیون و

اینترنت وصل باشد، قادر خواهد بود تا ادامه آهنگ و ویدیو را در تلویزیون به نمایش بگذارد. این افزونه از طریق صفحه افزونه‌های گوگل کروم قابل دریافت و نصب است. گفتنی است این افزونه علاوه بر این که قادر است ویدیو و آهنگ را به کروم‌کست ارسال کند، قادر است که صفحات بازدید شده شما را نیز با استفاده از کروم‌کست روی تلویزیون نمایش دهد. ■



افزونه بالن «Ballon»



اگر از نحوه نوشتن لاتین کلمه بالن (بادکنک) در نام این افزونه تعجب کرده‌اید و گمان می‌کنید این یک اشتباه تایپی است، در اشتباه هستید. نام این افزونه از سه حرف ال انگلیسی تشکیل شده است. شاید این نام به این دلیل به این شکل نوشته شده‌است که توسعه‌دهندگان از استفاده از نام بالن به شکل صحیح معذوریت قانونی داشته‌اند. در هر حال نام برنامه به این شکل است و اشتباهی در نوشتن آن رخ نداده است! این افزونه برای افرادی مناسب بود که معمولاً اطلاعات زیادی را در

فضاهای مختلف ابری بارگذاری می‌کنند. برای نمونه اگر شما اطلاعات خود را در چند خدمت ابری رایگان مانند دراپ‌باکس، وان‌درايو، گوگل درایو و... دارید، با استفاده از این افزونه خواهید توانست با تنها یک کلیک، اطلاعات خود را با یک بالن به ابرها سپرده‌اید. این ابزار برای افرادی که از سیستم‌عامل گوگل کروم استفاده می‌کنند نیز بسیار کاربردی است. به شکلی که هر گاه بخواهند اطلاعاتی را ذخیره کنند، به جای ذخیره، مستقیم در دیسک سخت، آن را به شکل مستقیم به فضای ابری خود ارسال می‌کنند که از تمامی خدمات ابری مشهور نیز پشتیبانی به عمل می‌آورد. برای مثال اگر در حال مشاهده یک آلبوم تصویری هستید، می‌توانید روی آن کلیک راست موس را فشرده و در فهرست نمایش داده شده، از طریق گزینه «ذخیره پیوند در...» آن را در خدمت ابری رایگان و یا پولی مورد نظر خود ذخیره کنید. ■

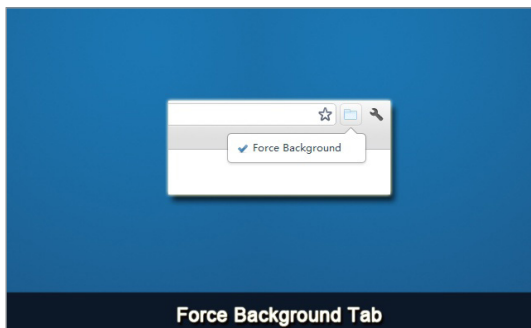
یوتیوب را به صورت پنجره‌های معلق تماشا کنید Floating YouTube



یوتیوب از آن دسته از خدمات برخط هم‌رسانی ویدیو است که هر چه خدمات مشابه ایرانی برای آن ایجاد شود، هرگز نمی‌توانند در مقابل امکانات و سطح خدمات و کیفیت آن دوام آورند، مگر با جلوگیری از دسترسی به این خدمت در داخل کشور که باعث اجبار کاربران برای مراجعه به خدمات دیگر خواهد شد. یوتیوب خدمتی است که توسط شرکت گوگل خریداری شده و هم‌اکنون نیز توسط گوگل پشتیبانی می‌شود و میلیون‌ها ویدیوی آموزشی، علمی، سرگرم‌کننده و جالب به زبان‌های مختلف دنیا در آن قرار دارد. تمامی ویدیوهای یوتیوب دارای کیفیت‌های مختلفی هستند که فراخور سرعت اینترنت، می‌توان یکی از این کیفیت‌ها را برگزید. در سال‌های اخیر قابلیت مشاهده ویدیو با کیفیت چهار هزار «4K» و نرخ فریم‌های بالاتر از سی فریم بر ثانیه نیز به یوتیوب افزوده شده‌است که

برتری خاصی به این خدمت هم‌رسانی ویدیوی اینترنتی گوگل بخشیده است. با وجود این، مشاهده ویدیو در این خدمت از روش‌های مختلفی میسر است. چه با استفاده از ابزار ارزان قیمت کروم‌کست این دسترسی انجام شود، چه از طریق مرورگر اینترنتی، کیفیت این خدمت هم‌رسانی ویدیو تقریباً یکسان و راضی‌کننده است. نرم‌افزار کاربردی آن در اندروید و آی‌اواس قابلیت خوبی را برای کاربران فراهم می‌آورد که در آن می‌توان یک ویدیو را به صورتی کوچک در پایین و سمت چپ قرار داد. هر چند این قابلیت، امکان بسیار محدودی است اما با نصب افزونه‌ای در مرورگر اینترنتی گوگل کروم، قادر خواهید بود که ویدیوهای نمایش داده شده در یوتیوب را به شکل پنجره‌های معلق مشاهده کنید. کاربرد این افزونه برای افرادی است که به مشاهده ویدیو پرداخته و در عین حال کارهای دیگر را نیز در کنار تماشای آن ویدیو انجام می‌دهند. ■

اجبار برای گشودن سربرگ‌ها در حالت پس‌زمینه Force Background Tab



برخی مواقع ممکن است برای شما نیز پیش آمده باشد که در یک پیام رایانامه که برایتان ارسال شده، چندین پیوند به دیگر پایگاه‌های اینترنتی قرار دارد و شما مجبورید با کلیک بر هر یک از پیوندها، به اطلاعات این پیوندها دسترسی داشته باشید. در این مواقع شما هنوز خواندن آن پیام را به پایان نرسانده‌اید، ولی می‌خواهید پیوندهای فوق در سربرگ‌های جداگانه‌ای نمایش داده شوند. هم‌چنین در مواقعی که یک مطلب را مشاهده می‌کنید که در آن چند پیوند جالب قرار دارد، با کلیک روی آن، به آن پایگاه اینترنتی و یا صفحه موجود در پیوند منتقل خواهید شد و سربرگ جدید مقابل شما قرار گرفته و تمرکز شما را برای مطالعه آن مطلب به هم می‌ریزد. برای رفع چنین مشکلی، افزونه‌ای ساده در گوگل کروم قرار دارد که می‌تواند تمامی پیوندهای باز شده در سربرگ جدید را به صورت ساکت و در پس‌زمینه اجرا کند. به این شکل، آن سربرگ‌ها در سربرگ جدید باز می‌شوند؛ اما تا زمانی که نخواسته‌اید، به آن سربرگ منتقل نخواهید شد. ■



گشودن چندین پیوند در یک حرکت Linkdump

در برخی مواقع شاید نیاز داشته باشید، چندین پیوند را به آسانی و با یک کلیک راست در سربرگ‌های مختلف باز کنید. برای مثال چند عدد پیوند برای بارگیری چندین فایل موسیقی در یک صفحه قرار دارد که شما می‌خواهید به راحتی و با یک کلیک تمامی آنان را بارگیری کنید. برای این کار تمامی پیوندها را با کلیک راست موس انتخاب کرده و پس از این که در اطراف پیوندها کادر قرمز رنگی کشیده شد، موس را رها کنید تا تمامی پیوندها در سربرگ‌های مختلف باز شوند. برای دریافت این افزونه نیز به صفحه افزونه در این پیوند به پایگاه اینترنتی افزونه‌های گوگل کروم مراجعه کرده و با کلیک روی دکمه آبی رنگ نصب افزونه‌ها، آن را نصب کنید. ■



نمایش تمامی سربرگ‌ها در یک سربرگ OneTab

هنگامی که چندین سربرگ در مرورگر اینترنتی به صورت هم‌زمان در حال اجرا باشند، معمولاً سامانه دچار کندگی و اختلال می‌شود. به طوری که کارایی و بازدهی آن تا مقدار زیادی کاهش خواهد یافت. در این مواقع به دلیل بارگذاری موارد متعدد مانند تصاویر و... در حافظه اصلی رایانه، حافظه پر می‌شود. با استفاده از افزونه یک سربرگ «OneTab»، هرگاه سربرگ‌های زیادی در مرورگر کروم در حال اجرا بود و کارایی سامانه تحت تأثیر قرار گرفت، می‌توانید روی نقشک این ابزار در نوار ابزار کلیک کنید تا تمامی سربرگ‌های در حال اجرا به یک‌باره بسته شده و تنها یک سربرگ که فهرستی از پیوندها به آن سربرگ‌ها است، ایجاد شود. این کار باعث می‌شود سربرگ‌های در حال اجرا در مرورگر بسته شده و حافظه اصلی خلاص شود و همچنین نشانی سربرگ‌ها به همراه عنوان آن‌ها برای استفاده‌های بعدی حفظ شود. ■



تماشای ویدیو به صورت پنجره‌های معلق Panelizer

در یکی از افزونه‌های گفته شده در این مطلب، افزونه‌ای را معرفی کردیم که برای نمایش ویدیوهای یوتیوب به صورت معلق و در پنجره‌های کوچک کاربرد داشت. حال اگر بخواهید ویدیوها و موسیقی‌های برخی خدمات برخط مانند نت‌فلیکس را در کادری معلق به صورت یک اعلان مشاهده کنید، می‌توانید افزونه‌ای را نصب کنید که ویدیو را در پایین سمت راست نمایش می‌دهد. برخلاف افزونه مورد استفاده برای معلق کردن ویدیوهای یوتیوب، افزونه پنلایزر «Panelizer» قادر است ویدیو را خارج از چارچوب مرورگر نیز به نمایش بگذارد.

ابزار فوق، با خدمات مختلفی قابل استفاده است. با استفاده از این افزونه به راحتی می‌توانید ویدیوها را در یک کادر معلق نمایش داده و به دیگر کارهای خود مشغول شوید. برای دریافت این افزونه باید از طریق این پیوند به پایگاه اینترنتی افزونه‌های گوگل کروم شده و با کلیک روی دکمه آبی رنگ بالای کادر به نمایش درآمده،

افزونه را نصب و استفاده کنید. برای استفاده از افزونه، ابتدا وارد پایگاه اینترنتی و یا صفحه دارای ویدیوی مورد نظر شده و با کلیک راست کردن روی ویدیو و مشاهده فهرست، روی گزینه «Panelizer» کلیک کنید. این کار باعث نمایش ویدیوی فوق به صورت کادری معلق خواهد شد. ■

تغییر تصویر پس‌زمینه سربرگ

اگر یک کاربر اینترنت فعال و همواره برخط هستید و در اکثر خدمات اینترنتی فعالیت می‌کنید، ممکن است برخی مواقع در استفاده از مرورگر و گشت و گذار در دنیای اینترنت با کمی کسالت و خستگی مواجه شوید که حوصله شما را سر خواهد برد. در این مواقع، انجام برخی تنظیمات ظاهری در مرورگر و سفارشی کردن آن بسیار مناسب خواهد بود. با وجود این اگر تصاویر قالب‌های کروم و... برایتان خسته‌کننده می‌شود، بهتر است از افزونه‌ای درج تصویر برای سربرگ‌ها «pics» استفاده کنید که با اتصال به اینترنت، تصاویر جدید را بارگیری و به عنوان تصویر زمینه در سربرگ‌ها از آن‌ها استفاده می‌کند. در هربار استفاده، تصویری جدید از اینترنت بارگیری می‌شود که باعث تنوع در ظاهر مرورگر خواهد بود. ■





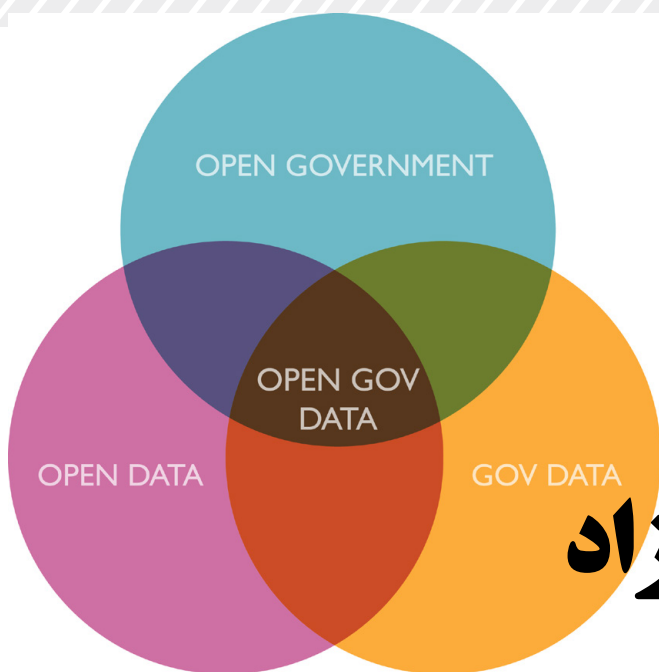
| کسب و کار |

| ۲۶ | اهمیت چرخش آزاد اطلاعات دولتی

| ۳۰ | چگونه کسب و کار موفق را راه اندازی کنیم

| ۳۲ | تجربیات مدیرعامل ردهت از چندین سال مدیریت

| ۳۴ | متن باز شدن دیگر اختیاری نیست حتی برای اپل



اهمیت چرخش آزاد اطلاعات دولتی

نکات کلیدی به دست آمده از سطح بلوغ شهرهای مورد پژوهش در زمینه چرخش آزاد اطلاعات را می‌توان به موارد زیر خلاصه کرد؛ ۱۷ درصد از آنان هیچ گونه آشنایی و ارتباطی با اطلاعات آزاد نداشته و با این موضوع کاملاً غریبه بودند

۲۴ درصد آنان تجارب کمی در مورد اطلاعات آزاد داشتند؛ مانند برخی اطلاعات خاص و نه اطلاعات شهری و یا اطلاعات سیستماتیک و مهم

۴۷ درصد از این شهرها از تجارب نسبتاً متوسط و خوبی در زمینه اطلاعات آزاد و چرخش اطلاعات برخوردار بودند. با این حال سیاست مشخصی در قبال انتشار اطلاعات و یا قوانین مرتبط با اطلاعات آزاد در آنان وجود نداشت و همچنین به صورت شفاف مشخص نبود که چه مواردی باید محرمانه باشد.

۱۲ درصد از شهرها نیز تجربیات پیشرفته و بالایی را در این زمینه دارا بودند؛ از سیاست‌های شفاف و درگاه‌های اطلاع‌رسانی اینترنتی شهری، بسیار خوبی برخوردار بودند و به طور کلی در شرایط مطلوبی قرار داشتند. همچنین بر اساس پژوهش صورت گرفته در این پژوهش و مطالعه مشخص شد که ۱۰ درصد از شهرها اصولاً دارای سامانه اطلاع‌رسانی یا پایگاه اینترنتی و حتی درگاه منسجم ارتباطی برای ارائه اطلاعات دولتی و شهری نبوده و در زمینه ارائه اطلاعات آزاد و

«حکمرانی خوب» است. چنان که حکمرانی خوب را بنیان توسعه خوانده‌اند. حکمرانی خوب از جمله مباحث تازه‌ای است که در دو دهه اخیر توجه پژوهشگران و محافل علمی و بین‌المللی جهان را به سوی خود جلب کرده است. به این دلیل که حکمرانی خوب به چرخش آزاد اطلاعات بستگی دارد و حکمرانی خوب نیز از ملزومات توسعه است، امروزه همه صاحب‌نظران به این موضوع معترف هستند که باید افراد به اطلاعات دولتی دسترسی بیشتری داشته باشند و مسائل مالی و سیاسی در دولت شکلی شفافتر به خود گیرد.

در تحقیقاتی جدید در کمیسیون اروپا بر روی پژوهش و مطالعه‌ای کار می‌شود که با استفاده از این پژوهش، بتوان چرخش آزاد شکاف‌ها را بین دولت و اصحاب عرصه اطلاع‌رسانی کاهش دهند و درک متقابل هر یک را نسبت به موضع بالاتر ببرند. بر این اساس آن‌ها افراد را تشویق می‌کنند تا اطلاعات خود را به وسیله نرم‌افزارهای آزاد / متن‌باز منتشر نمایند. همچنین دولت‌ها نیز می‌توانند با استفاده از نرم‌افزارهای آزاد و یا حتی تولید نرم‌افزارهای دولتی به شکل متن‌باز به آزادی چرخه گردش اطلاعات، کمک کنند. این پژوهش و مطالعه تا به حال در ۱۴۰ شهر از ۶ کشور جهان برای گسترش و توسعه و انتقال دید مناسب درباره آزادی اطلاعات، مشغول اطلاع‌رسانی و پژوهش‌های محلی هستند.

سازمان ملل متحد در تحقیقات و پژوهش‌های اخیر که از حدود ۵ سال پیش تا حال، نیز به این نکته اشاره داشته است که اطلاعات آزاد می‌تواند به گشودن عرضه، تولید و انتشار دانش محلی جهانی شود. لازم به ذکر است که گردش آزاد اطلاعات دولتی و اطلاعات باز/آزاد برای تمامی دولت‌ها حیاتی است. موسسه اطلاعات باز (Open Information Foundation) براساس گزارش شرکت مک‌کینزی «۷ بخش به تنهایی می‌توانند ۳ میلیارد دلار در سال ارزش افزوده را به واسطه اطلاعات آزاد ایجاد کنند». «آزادی اطلاعات» و «حق دسترسی به اطلاعات»، به این معنی است که اکثر شهروندان توانایی دسترسی به بخش اعظمی از اطلاعات دولتی غیرمحرمانه و حساس را داشته باشند، که برای شفافیت دولت و از بین بردن فساد اقتصادی امری حیاتی است. چندان که آزادی اطلاع‌رسانی و دسترسی به اطلاعات در قرون جدید و پسارنسانس به «رکن اصلی دموکراسی» تعبیر می‌شود.

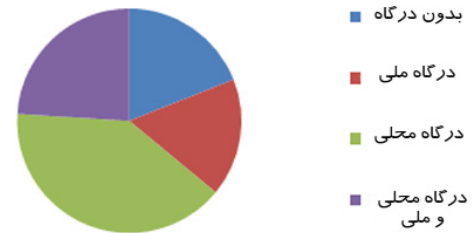
بی‌هیچ شک اطلاعات باز و آزاد، سیاست عمومی بسیار مهمی است و دغدغه اصلی کشورهایی در حال توسعه قرار گرفتن در مسیر صحیح توسعه و عدالت و شفاف‌سازی است؛ از سوی دیگر آمار و پژوهش‌های رخ داده در سازمان‌های بین‌المللی، بیانگر آن است که از مهم‌ترین عوامل در رشد و توسعه کشورها،



دکتر جولیتا کالیناوسکایته

صاحب‌نظران به این موضوع معترف هستند که باید افراد به اطلاعات دولتی دسترسی بیشتری داشته باشند و مسائل مالی و سیاسی در دولت شکلی شفافتر به خود گیرد.

نمودار شهرهای برخوردار از درگاه اطلاع‌رسانی



چرخش اطلاعات، پیشرفت خاصی نداشته‌اند. همچنین ۱۷ درصد از آنان به یک درگاه ملی اطلاع‌رسانی آزاد برخوردار بودند و ۴۴ درصد از آنان از درگاه محلی برای ارائه اطلاعات آزاد برخوردار بوده و ۲۹ درصد از آنان نیز هم دارای درگاه محلی بودند و هم درگاه ملی.

در پایان نیز بر اساس قالب انتخابی برای انتشار آمار و اطلاعات دولتی و قرار دادن آن در اختیار مردم، حدود ۷۷ قالب مجزا در حال استفاده است که برخی از این قالب‌های انحصاری و برخی دیگر نیز آزاد / متن‌باز بودند. برخی از این قالب‌ها نیز قالب‌های تخصصی برای برخی امور خاص علمی و آماری بودند. پرکاربردترین قالب را می‌توان قالب CSV که نوعی قالب صفحه گسترده است، دانست که قالب ۶۲ درصد از پرونده‌های ارائه شده را شامل می‌شد. قالب بعدی نیز قالب XML بود که یک زبان نشانه‌گذاری دیگر مانند HTML است. یکی دیگر از قالب‌های پرکاربرد دیگر که برای نگهداری اسناد و جداول صفحه‌گسترده به کار می‌رود همان پرونده‌های برنامه اکسل مایکروسافت یعنی XLS بود که با وجود انحصاری بودن در اکثر نرم‌افزارهای تحت‌وب، گوشی‌هوشمند و یا آزادی چون لیبره‌آفیس قابل ایجاد و مشاهده است. اکثر شهرها اطلاعات خود را با استفاده از یک یا دو سه قالب خاص منتشر کرده‌اند. باین‌حال در رسم زیر می‌توانید نموداری از ۱۰ قالب محبوب در ارائه اطلاعات آزاد در این شهرها را مشاهده نمایید:

۱۰ قالب محبوب در ارائه اطلاعات آزاد در این شهرها؛ (اطلاعات به درصد)



براساس پژوهش این گروه مشخص گشته است که حداقل نیمی از شهرها درک صحیحی از مزایای مرتبط با استفاده از برخی قالب‌های مورد استفاده خود ندارند؛ براین‌اساس اکثر شهرهای مورد پژوهش توسط این پژوهش و مطالعه نشان می‌دهد که به جز معدود شهرهای پیشرفته اکثر این شهرها از قالب‌های مناسب برای انتشار اطلاعات استفاده نمی‌کنند. بر اساس استانداردهای جاری بین‌المللی برای انتشار اطلاعات به صورت آزاد، اطلاعات باید از قالب‌های خاصی استفاده نمایند که برای محتوایی که قرار است توسط آن ارائه شود کاملاً مناسب باشند و علاوه‌براین موضوع بهتر است متن‌باز و آزاد باشند. زیرا اگر وابسته به فناوری و نرم‌افزار خاصی باشند، قابلیت انتقال کمتری خواهند داشت؛ به شکلی که اگر نرم‌افزار ایجادکننده آن پرونده تغییر یافت به راحتی بتوان به ویرایش و مشاهده آن داده‌ها پرداخت.

همان‌طور که اشاره شده استفاده از قالب‌های آزاد یکی از بهترین روش‌های انتشار محتوای که تضمین می‌کند اطلاعات عرضه شده بدون وابستگی به نرم‌افزار خاصی قادر به مشاهده و ویرایش باشند. بدون این ویژگی عملی بخشی از آزادی چرخش اطلاعات محدود به افراد خاصی خواهد شد که دسترسی به نرم‌افزار مورد استفاده برای ایجاد پرونده، داشته و می‌توانند از آن استفاده نمایند. بنابراین بهتر است برای

ارائه اطلاعات از نرم‌افزارهای آزاد / متن‌باز استفاده شود.

در ابتدا اکثر افراد حتی در گروه پژوهشی مورد اشاره از مدل داده‌های پیوندی متن‌باز (آزاد) LOD حمایت می‌کردند که داده‌های مورد نیاز را به صورت مجموعه‌ای به هم پیوسته ذخیره می‌نمود. همچنین در این مدل از روش آر‌دی‌اف «RDF» برای ذخیره و بازیابی معنای قابل پردازش توسط ماشین بکار می‌رفت؛ به این دلیل که این مدل با زبان اکس‌ام‌ال نوشته می‌شود؛ در تمامی ویرایشگرهای متنی قابل ویرایش است. باین‌حال به دلیل پیچیدگی این مدل و سختی یادگیری ایجاد چنین پرونده‌هایی، این مدل دیگر توسط افراد غیرحرفه‌ای و عادی استفاده نمی‌شود، به این‌خاطر این مدل رواج زیادی در شهرهای مورد پژوهش ندارد.

در طی سالیان مختلف استفاده از قالب زبان نشانه‌گذاری اکس‌ام‌ال در حال افزایش است. این زبان قدرت بالایی در نمایش اطلاعات مختلف به صورت‌های مختلف دارد که برتری خاصی به آن بخشیده است. از دیگر مزایای آن می‌توان به امکان ارائه آسان و راحت آن در درگاه‌های اینترنتی اشاره کرد که حتی می‌توان آن را با پایگاه اینترنتی و دیگر مطالب به صورت درونی ارائه داد. این قالب کاربردهای فراوانی دارد و از ساختار بسیار ساده‌ای نیز برخوردار است به عنوان مثال اکثر برنامه‌های گرافیکی از ساختار اکس‌ام‌ال برای چیدمان عناصر بصری نرم‌افزار خود استفاده می‌کنند؛ مانند گوگل اندروید که از اکس‌ام‌ال برای طراحی عناصر گرافیکی نرم‌افزارهای کاربردی خود بهره می‌برد.

اخیراً قالب جدید و متن‌بازی با نام سی‌اس‌وی «CSV» یا (مقادیر جداسده با ویرگول) مورد استفاده قرار می‌گیرد که داده‌های آن با ویرگول از هم جدا می‌شوند. پرونده‌هایی که با استفاده از چنین روشی ایجاد شده‌اند را می‌توان با استفاده از نرم‌افزارهای اکسل و کالک (calc) لیبره‌آفیس باز نمود. سی‌اس‌وی را می‌توان یکی از محبوب‌ترین قالب‌ها برای ارائه محتوای صفحه‌گسترده دانست که به صورت گسترده توسط شهرهای مورد پژوهش نیز استفاده شده است. از مزایای آن نسبت به اکس‌ام‌ال می‌توان به سادگی کار با آن اشاره کرد؛ علاوه‌براین بدون نیاز به نرم‌افزار خاص و تنها با یک ویرایشگر ساده نیز قابل ویرایش است. همچنین زبان‌هایی مثل پایتون و پرل نیز از توابع درونی خاصی برای کار با این پرونده‌ها برخوردارند که توسط نرم‌افزارهای نوشته شده در این برنامه‌ها نیز می‌توان داده‌های ذخیره شده در این قالب را پردازش کرد. به‌طور کلی از تمام موارد گفته شده و با وجود مدل ارائه داده پیشرفته LOD و آر‌دی‌اف که توسط اکثر شهرهای مورد پژوهش استفاده نشده‌اند می‌توان گزاره‌های زیر را نتیجه گرفت:

دارندگان اطلاعات شهری سادگی و راحتی استفاده از قالب سی‌اس‌وی را ترجیح می‌دهند

طرفداران اطلاعات باز همواره از سی‌اس‌وی در مقابل مدل‌هایی مثل آر‌دی‌اف، پشتیبانی کرده و معتقدند اطلاعات آن وضوح بیشتری دارند.

گروه درحال‌رشد و بزرگی از برنامه‌نویسان و توسعه‌دهندگان ترجیح می‌دهند به جای استفاده از بانک‌های اطلاعاتی و غیره، از قالب سی‌اس‌وی استفاده نمایند که انعطاف بیشتری را برای کار با داده‌ها، فراهم می‌کند.

شهروندان عادی آرزو دارند از اطلاعات در قالب سی‌اس‌وی استفاده نمایند؛ زیرا چنین قالب‌هایی برای استفاده توسط آنان بسیار راحت‌تر و مناسب‌تر است تا استفاده از ساختار پیشرفته اکس‌ام‌ال و آر‌دی‌اف.

استفاده گسترده از اطلاعات ساده و آسانی چون سی‌اس‌وی، نشان می‌دهد که قالبی که اطلاعات را با پیچیدگی کمتری در خود نگاه داشته و نمایش دهد، بسیار بیشتر از قالب‌های پیشرفته مورد استفاده قرار می‌گیرد. همچنین کاربران حرفه‌ای و توسعه‌دهندگان نیز برای انعطاف و راحتی بیشتر در هنگام برنامه‌نویسی چنین قالب‌هایی را به دیگر قالب‌ها ترجیح می‌دهند. ■





داده‌های باز بیشتر برای کاربران بیشتر...

کشور دنیا دارای دولت بابستره
داده‌های باز هستند

۴۰+

مجموعه اطلاعاتی باز در سایت
data.gov برای امریکا وجود دارد

۹۰,۰۰۰+

بازدید از سایت «اطلاعات باز» انگلستان
در تابستان ۲۰۱۳ صورت پذیرفته

۱,۴۰۰,۰۰۰

در روز هکاتون بین‌المللی «اطلاعات
باز» ۲۰۱۳ مشارکت کردند

۱۰۲ شهر

مجموعه اطلاعاتی در سراسر دنیا
توسط دولت‌ها باز شده‌اند

۱,۰۰۰,۰۰۰+

تا سال ۲۰۱۳

... ارزش بیشتری را

می‌تواند به همراه آورد

+ ۵۰٪

سهام مصرف‌کننده از ارزش
بالقوه اطلاعات باز

+ ۱۰۰,۰۰۰

نرم افزار پزشکی، سلامتی و
تناسب اندام برای تلفن‌های
هوشمند

۳ ساعت در سال

صرفه جویی برای مسافران با
آگاهی از تغییر زمان بندی‌ها
بر اساس اطلاعات باز

۳,۰۰۰,۰۰۰,۰۰۰ واحد

گاز دی‌اکسید کربن برای ساختمان‌هایی
که می‌توانند توسط اطلاعات باز
شناسایی شوند، کمتر منتشر می‌شود

۳,۰۰۰,۰۰۰,۰۰۰ دلار

تخمین ارزش بالقوه اطلاعات
باز در هفت زمینه است



۱۱ نکته برای هنگامی که استارت آپ شما با بازار جدید مواجه می شود

چگونه کسب و کار موفق را راه اندازی کنیم

در کجای کار ایستاده‌اید؛ این اعمال باعث می‌شود که به سرعت اشکالات کار را رفع کنید و یا چاه‌هایی را پر کنید، که در صورتی که شنونده خوبی نباشید؛ ممکن است درون آنان بیفتید.

دیدنی کلی از مشتری نهایی داشته باشید، B2B یا B2C کارهایی اشتباه هستند مسائلی که باعث می‌شود پول خود را دور بریزید یا منجر به تلاشی بی‌فایده در پایین جاده شود.

۵ پتانسیل شراکت:

قبل از این که وارد بازار جدید از مشتریان شوم، تحقیقات خود را انجام می‌دهم، سپس به یکی از شرکا و یا وابستگان تماس می‌گیرم تا از آنان مشورت بگیرم. این کار را با دعوت کردن از آنان برای یک ناهار انجام می‌دهم. من از آنان تمامی خوب و بد و جزئیات کار را جویا خواهم شد. این به یکی از بزرگ‌ترین روش‌های من تبدیل شده است تا به استراتژی خود شکل دهم. به طور کلی اگر شریکی وجود داشته باشد که پلی بین خلاءهای بازار جدید و قدیم من ایجاد کند، من تلاش خواهم کرد تا آن را کاملاً آشکار کنم.



در روزها و
یا ماه‌های
اول سؤالات
و فرم‌هایی را
به مشتریان
بدهید تا از آنان
اطلاعاتی را جمع
کنید

بر مشتریان جدید تغییر دهند. بهترین جواب‌های آنان در زیر آمده است:
باز خورد مشتری:
وقتی در حال توسعه مدل خود هستید، ضروری است که شما باعلاقه و به سرعت به نیازهای مشتریان خود توجه کنید. چیزهایی که شما میدان‌دار ثابت آنان هستید؛ زمانی که وارد میدان جدید می‌شود ممکن است دیگر این‌طور نباشد. ضروری است که در روزها و یا ماه‌های اول سؤالات و فرم‌هایی را به مشتریان بدهید تا از آنان اطلاعاتی را جمع کنید که به صورت بالقوه برای گرفتن بازخورد مفید هستند. این بازخوردها را تحلیل کنید و بسنجید که

جلب نظر و مجذوب کردن مشتریان از آن دسته چالش‌هایی هستند که اکثر فعالین اقتصادی با آن دست و پنجه نرم می‌کنند. اما حتی اگر شما یک Busi-ness Man کاملاً موفق باشید، نیاز به کشف بازارهای جدید و مشتریان جدید خواهید داشت. حسی که در این مواقع به شما دست خواهد داد حس شروع دوباره همه این مراحل است که طی کرده‌اید. برای صرفه‌جویی در هزینه و زمان، من از چند تن از کارآفرین‌های YEC (Young Entrepreneur Council) (سؤالاتی پرسیدم که چه چیزی را فراتر از هر چیزی در اولویت قرار داده اند، زمانی که می‌خواستند، مدل خود را مبتنی



آلکساندر گریب
تأسیس‌دهنده

من معتقد هستم که مشتریان هم از این همگام بودن لذت خواهند برد و این همیشه مانند آن است که گویی برند مورد علاقه آنان به آن‌ها انسانی برخورد کرده و دوست آنان هستند. اگر آنان با شما می‌شوند، آن وقت است که به تک تک کلمات شما باور خواهند داشت.

۳ بررسی تحقیقات اینترنتی:

مطمئن شوید که شما در مورد بازار مناسبی تحقیق می‌کنید، از سایت‌هایی مثل آمازون استفاده کنید وارد دسته‌بندی‌های مشابه شوید و ببینید که دیگران در مورد محصولاتی که فعلاً در بازار هستند چه نظراتی دارند. مثلاً مشکلات و گرفتاری که آنان با محصولات دیگر دارند و یا چیزهایی که در مورد آن محصولات دوست دارند. همچنین چک کنید که معمولاً مردم برای چه چیزهایی پول پرداخت کرده‌اند و چه محصولاتی فروش بیشتری داشته‌اند، آمازون (در ایران دیجی کالا یا ... و Yelp سایت‌های خوبی برای انجام این دست تحقیقات هستند.

۳ رقابت:

شاید هرگز نیاز نباشد تا به ارزیابی کلی از رقبا یا ظرفیت اسمی بازار بپردازید. مهم است که بدانید رقبای شما تا چه حد قدرتمند هستند، چه کارهایی را به خوبی انجام می‌دهند، چه چیزهایی را به خوبی انجام نمی‌دهند، خودتان را از بازار چگونه می‌توانید اقتراق دهید و این که بازار هدف چه قدر بزرگ است. همچنین شما نباید چنین تصور کنید که قرار است به شکلی خیره‌کننده از رهبر فعلی بازار بزرگ‌تر شوید.

۳ مشتریان فعلی شما:

کاربران و مشتریان فعلی در حال حاضر اعتماد لازم را به برند شما دارند، مطمئن شوید که هرگز آنان را در هنگام انجام این تغییرات ناامید نکنید (بیگانه نشمارید) زیرا این مشتریان سابق شما هستند که باعث شده‌اند که چیزی که امروز هستید؛ باشید، با این حال سعی کنید خود را در چیزی که هستید ایزوله نکنید!

۳ پتانسیل تغییرات:

هنگامی که زمان آن فرا می‌رسد تا مدل خود را مبتنی بر مشتریان جدید بنا نمایید، باید روی این موضوع تمرکز کنید که شما قرار است برند جدید را معرفی کنید که بتواند مخاطبین جدید را تحت تأثیر خود قرار دهد. این کار ممکن است برای سازمان‌ها که قصد تغییر هویت دارند چالش‌برانگیز باشد که هویت زیبای خود را حفظ کنند و همین‌طور بتوانند موارد جدید و قابل قبولی برای عرضه داشته باشند. هنگامی که شما مراحل تکامل به سمت بدل شدن به یک برند و هویت جدید را طی می‌کنید، نیاز دارید تا نام تجاری خود را در موضوعات جدید جای بیاورید، از بازار جدید که قصد ورود به آن را دارید، یک مصاحبه از افراد داشته باشید تا درک درست و جامعی از نیازها و ترجیحات آنان داشته باشید. این به شما کمک خواهد کرد که اگر شرایط در بازار جدید بحرانی شد بتوانید از پس بحران پیش آمده برآیید.

۳ نیازها را از نو تعریف کنید:

شما باید وقت بیشتری را صرف کنید و تلاش بیشتری نمایید تا به مشتری جدید دست یابید. با تلاشی بسیار بیشتر از زمانی که باید برای جلب مشتریان فعلی خود را انجام دهید. مردم معمولاً خیال می‌کنند، که تغییر بازار و کسب به هدفی جدید بسیار ساده است؛ اما اغلب فراموش می‌کنند که تلاش برای دستیابی به بازارهای جدید، همیشه برای آنان و اطرافیان‌شان تلاش بزرگی بوده است. درباره اولین بازاری که تجارت شما در آن کلید خورد فکر کنید. این کار به همان اندازه نیاز به تلاش دارد که هنوز برند شما شناخته شده نبود. در هر حال شما نیاز دارید تا نیازهای تجاری خود را از نوع تعریف کنید.

۳ دیدگاه از زمینه‌های دیگر :

روش معمول برای تحقیق در مورد یک مارکت به راحتی با نگاه کردن به روند مشتریان در زمینه کاری شما و پرسش سؤال از مشتریان فعلی قابل انجام است، که چه چیزی را می‌خواهند در آینده شاهد باشند. به جای آن که خود را طبقه‌بندی

کنید! تلاش کنید تا دید مناسبی، از موضوع کاری خود به وسیله مطالعه دیگر شرکت‌ها در زمینه مشابه خود به دست آورید. از تجربیات آنان بیاموزید و از نحوه ورود آنان به بازار جدید و راه‌حل‌های آنان کسب تجربه کنید.

۳ تجزیه و تحلیل کامل بازار:

شما نیاز دارید که تشخیص دهید که چه چیزی واقعاً نیاز است، این کار را با مطالعه و تحقیق دقیق و جمع‌آوری اطلاعات به دست خواهید آورد. شما نباید این تصمیمات را سرسری بگیرید. یک مطالعه درست می‌تواند از چالش‌هایی که ممکن است برای شما اتفاق بیفتد پرده‌برداری کند. پس شما می‌توانید آنان را پیش‌بینی کنید و محصول و کمپین خود را به صورت مناسبی برنامه‌ریزی کنید.

۳ پیام‌های استراتژیک:

مطمئن شوید که پیام خود را در مورد محصولات به درستی به مخاطبین انتقال می‌دهید. اگر شما می‌خواهید که بین محصول جدید که در بازار قدیم عرضه می‌کردید و محصولات جدید در بازار جدید، تمایز قائل شوید؛ باید در پیام‌های شما این هدف و منظور کاملاً روشن باشد. توسعه یک محصول شمار را آماده می‌کند با این فرصت که بتوانید مشتریان فعلی را درباره رشد، مسیر شما و قدرت خلاقیتان، هیجان زده کنید. باین حال اگر موقعیت مناسبی را نتوانید فراهم کنید، موفق نخواهید بود.

۳ تغییرات در فروش و خدمات:

به چالش‌های کلی در رابطه با فشارها بین مشتری و تیم خدمات چشم نپوشید. شما به افراد مشخصی آموزش داده‌اید تا محصول خاصی را به فروش برسانند یا با مشتریان خاصی مواجه شوند. حال زمانی که مدل خود را تغییر می‌دهید، باید خود را آماده کنید تا چرخه‌ی فروش، تغییرات محصولات و خدمات را به‌روز نمایید. حتی مجبور خواهید بود تا چرخه‌ی توزیع، فروش و یا خدمات پس از فروش را از نوع تنظیم نمایید. ■



مهم است که بدانید رقبای شما تا چه حد قدرتمند هستند، چه کارهایی را به خوبی انجام می‌دهند، چه چیزهایی را به خوبی انجام نمی‌دهند





تجربیات مدیر عامل ردهت از چندین سال مدیریت

و آموخته‌های وایت‌هارست در عرض هفت سال مدیریت وی بر ردهت دانست. این کتاب چراغ راهی است برای مدیران دیگر موسسات مرتبط با مدل آزاد / متن‌باز که چگونه بتوانند با حفظ شایسته‌سالاری، اصول همکاری و شفافیت خود را برای رویارویی با قرن ۲۱ ام آماده کنند؛ به به‌طوری که وایت‌هارست معتقد است این کتاب در مورد مواردی است که او تا به حال آموخته است. همانند بخش‌های دیگر دنیای آزاد / متن‌باز این کتاب نیز برای نقد و بررسی در اختیار عموم بوده است؛ تا این که بالاخره به تدریج منتشر شود. خوانندگان می‌توانند نظرات و پیشنهادات

متن‌باز بهره‌جسته‌اید؟» وایت‌هارست سریعاً پاسخ داد که «بله، ما کار را با استفاده از نرم‌افزار آزاد انجام دادیم، از لیبره‌آفیس که یک واژه‌پرداز محبوب متن‌باز است استفاده کرده‌ایم. و درحالی که میکروفون را با هیجان زیاد در دست خود می‌فشرد گفت: «و البته ماجراهای جالبی در این باره دارم».

بله، ما کار را با استفاده از نرم‌افزار آزاد انجام دادیم، از لیبره‌آفیس که یک واژه‌پرداز محبوب متن‌باز است استفاده کرده‌ایم

۳ سازمان‌های آزاد (باز) در عمل

مثل تمامی محصولات شرکت ردهت کتاب سازمان‌های آزاد هم ناشی از تلاش مشترک همه همکاران در ردهت است که به واسطه تجارب و بینش آنان شکل گرفته است. کتاب فوق را می‌توان چکیده‌ای از تجارب

در خلال جشنی که برای انتشار کتاب «The Open Organization» توسط جیم وایت‌هارست (مدیرعامل شرکت ردهت) در حال انجام بود، بیش از ۴۵۰ نفر از رهبران کسب‌وکارهای کوچک و بزرگ و فعالین بازار در یک کنفرانس مجازی اینترنتی دور هم جمع شده بودند تا به سخنان مدیرعامل شرکت ردهت درباره مدیریت تغییرات سریع رخ داده در اقتصاد در آینده و حال صحبت نمایند. شروع پرسیدن سوالات در ابتدا از جانب لیندا مریونس صورت گرفت که سردبیر نشریه تجاری هاروارد است. سوال او این بود که «آیا برای نوشتن کتاب خود از نرم‌افزاری



نویسنده
جیم وایت‌هارست

خود را درمورد هر بخشی از مطلب در کتاب به واسطه قابلیت کامنت «Comment» لیبره آفیس بیافزایند؛ که به خوانندگان اجازه حاشیه‌نویسی بر مطالب را خواهد بخشید. با این حال زمانی که جیم وایت‌هارست می‌خواست آنان را چاپ کند، در مکان‌هایی که انتظار می‌رفت باشند، حضور نداشتند. آن‌ها به بخش زیرین صفحات می‌افتادند ولی جیم وایت‌هارست می‌خواست آنان را به قسمت‌های کناری و حواشی کتاب منتقل کند، درست همان طور که دلش می‌خواست و در هنگام کار با صفحه‌نمایش مشاهده می‌کرد.

بنابراین او ایده جدید خود را برای بهبود این قابلیت در لیبره آفیس با همکاران و دیگر توسعه‌دهندگان در ردهت، در میان گذاشت که عرض چند هفته یکی از توسعه‌دهندگان نرم‌افزارهای آزاد در شرکت ردهت کائولان مک‌نامارا این ایده را برای نسخه جدید لیبره آفیس پیاده کرد. (به نظر می‌رسد در دنیای نرم‌افزارهای آزاد / متن‌باز نیز حرف مدیران زودتر مورد توجه قرار می‌گیرد!)

جیم می‌گوید «در واقع دیگر چنین قابلیت را در خود نرم‌افزار لیبره آفیس داشتیم، روزی فقط یک به‌روزرسانی در (توزیع) فدورای خود مشاهده کردم، که در واقع همان ویژگی مورد انتظار من بود!» با این حال جیم وایت‌هارست تنها کسی نبود که از این قابلیت منفعت می‌برد، بلکه پایگاه اینترنتی ZDNet نیز افزوده شدن این قابلیت را به عنوان «قدم حقیقی و روبه‌جلو» جشن گرفته بود.

قطعا این کتاب از این جهت برای جیم وایت‌هارست مهم است؛ زیرا او داستان‌های زیادی را در مورد آموخته‌هایش در آن آورده است که وی امیدوار بوده برای سازمان‌های آزاد (باز) مهم هستند. (براساس این ایده‌ای که می‌گوید؛ پروژه‌های آزاد / متن‌باز الگویی را ارائه می‌دهند که موفقیت سازمان‌ها را تضمین می‌کند.) به صورت کلی کتاب با هدف آموزش مدیران و دیگر رهبران نوشته شده است تا به آنان یادآوری کند، تا چه حد مدیریت و ایده‌های سنتی ناقص بوده و به طور فزاینده دیگر حرفی برای گفتن ندارند. جیم وایت‌هارست می‌گوید «دنیایی که ما در حال اعمال مدیریت در آن هستیم، با دنیای

قدیمی و سنتی دیروز که در آن آموزش دیده‌ایم؛ کاملا متفاوت است. اما پروژه‌های آزاد / متن‌باز موارد جدیدی را در چنته دارند؛ راه‌های جدید برای سازمان‌دهی، همکاری و رهبری یک تیم.» به عنوان مثال و در موردی که برای نرم‌افزار لیبره آفیس صورت گرفته بود؛ جیم وایت‌هارست موفق شد با فشار بر کارکنان خود بتواند چنین قابلیت را به نرم‌افزار مجموعه دیگری اضافه کند. چنین امکانی برای نرم‌افزارهای انحصاری و غیرآزاد وجود ندارد و فرد نمی‌تواند به کدهای آن دسترسی داشته باشد. بنابراین دانش افراد در یک شرکت می‌تواند برای شرکت‌های دیگر نیز مورد استفاده قرار گیرد که منجر به همکاری بهتر شرکت‌ها نیز می‌شود.

با تمام این تفصیل، جیم وایت‌هارست معتقد است در سازمان‌های آزاد تصمیم‌گیری‌ها به صورت دموکراتیک نیست و برخی موارد زور و قدرت کاملا بین اعضا تقسیم نخواهد شد و در ساخت و همکاری و تولید و راه‌اندازی یک پروژه، اولویت با کسانی خواهد بود که در این مسائل زودتر حاضر بوده‌اند. جیم وایت‌هارست می‌گوید: «صادقانه بگویم، صدای برخی افراد بیشتر از، دیگر افراد شنیده می‌شود.» که این موضوع بر اساس شهرت است. اکثر افراد شهرت خود را با استفاده از گفتگو و مکالمه به دست آورده‌اند؛ جیم وایت‌هارست هم معتقد است که مکالمه و گفتگو در دنیای آزاد / متن‌باز یکی از دلایل شهرت افراد بوده و یکی از اساسی‌ترین روش‌های جدا کردن سره را از ناسره به شمار می‌رود. به طور کلی یکی از روش‌هایی که افراد در جامعه آزاد / متن‌باز محبوب و مشهور می‌شوند نوع و سطح ارتباط آنان با افراد جامعه است که منجر به این محبوبیت خواهد شد.

جیم وایت‌هارست با خنده می‌گوید «ما یک سازمان کوم‌بایا «kumbaya» نیستیم، روش سنتی طوفان مغزها، جایی که افراد همه ایده‌های خود را مطرح می‌کنند تا یکی از آنان برگزیده شود دیگر روش خوبی برای تصمیم‌گیری نیست؛ بلکه مباحث ساینده و مکالمات هستند که باعث اتخاذ تصمیم درست می‌گردند.» وی می‌افزاید که شما به پوستی کلفت هم احتیاج دارید؛ تا به نظرات و ایده‌های آنان توجه کنید و از بین

این نظرات و ایده‌ها، نظرات خوب و برتر را انتخاب نمایید. این موضوع نیازمند آن است که توجه و وقت بیشتری را صرف دقت به نظرات دیگران نمایید. وی معتقد است، این روش، یکی از روش‌های جدیدی است که در پروژه‌های منبع‌باز از آن استفاده می‌شود. همچنین جیم وایت‌هارست اشاره می‌کند که شما کدهایتان نیستید، از منظر او افراد برای پیشرفت پروژه خود باید احساسات شخصی را کنار گذاشته و نظرات دیگران را به خوبی بشنوند و انتقادات را جدی بگیرند و اجازه ندهید انتقاد شما را به هم بریزد بلکه با روحیه‌ای شاد با آن مواجه شوید.

همچنین به عنوان یک نصیحت عرض می‌کنم که کینه به دل نگیرید. عصبانی و نگران ماندن از یک اشتباه می‌تواند کار آتی شما را هم تحت تاثیر قرار دهد. اشتباه را از ذهن خود بیرون کنید. بر این مساله که در آینده به بهترین شکل کارها را انجام دهید متمرکز شوید. به طور کلی انتقاد امر مهمی است و بهتر است آن‌چه را که شنیده‌اید مورد تجزیه و تحلیل و ارزیابی قرار دهید، مشخص کنید که آیا این یک انتقاد معتبر و صحیح است یا خیر و تصمیم بگیرید که چکار می‌توانید بکنید که مشکل را حل کنید یا اشتباه را اصلاح کنید. اگر این موضوع شکایتی است که شما مرتباً آن را می‌شنوید باید فکر کنید که شما از شرایط و وضعیت به وجود آمده چه چیزی می‌توانید یاد بگیرید که این کار دوباره رخ ندهد.

جیم وایت‌هارست رو به مخاطبین در کنفرانس مجازی متذکر شد که تبدیل شدن به یک شرکت متن‌باز دیگر یک مزیت به شمار نمی‌رود بلکه به بقای شرکت‌ها مرتبط گشته است. مادامی که شرکت‌های بیشتری در حال خدمت‌محور شدن هستند، بهتر است، شرکت‌های دیگر نیز به چنین مواردی اهمیت دهند تا بتوانند به موفقیت‌های بیشتری دست‌یابند.

در آخر این جاست که وصله نرم‌افزاری برای یک پردازشگر کلمه یا مدیریت یک کتاب برای یک رهبر یکسان خواهد بود. تمامی این‌ها به این به‌روزرسانی نیاز دارند؛ یکی به‌روزرسانی نرم‌افزار و دیگری به‌روز کردن برای بهبود قابلیت رهبری و کار جمعی در یک جمع و به‌صورت گروهی. ■



مباحث ساینده و مکالمات هستند که باعث اتخاذ تصمیم درست می‌گردند. وی می‌افزاید که شما به پوستی کلفت هم احتیاج دارید؛ تا به نظرات و ایده‌های آنان توجه کنید





اپل را نیز، از متن باز شدن گریزی نیست

متن باز شدن دیگر اختیاری نیست حتی برای اپل



کلیت‌فانی
نویسنده

زمینه نوشتن برنامه‌های بومی سیستمش ببرد و با به جان خریدن ریسک استفاده از زبان بومی خود، در رقابتی چون اندروید به تقویت زبان خود پردازد حتی اگر در سیستم‌عامل دیگری نیز استفاده شود.

۳ تا چه حد متن باز است؟

اپل قرار است هسته اصلی این زبان و تمامی موارد مرتبط هم چون کامپایلر را تحت مجوز خاص خودش متن باز نماید که نشان‌دهنده این است که تا چه مقدار متن باز خواهد بود. به نظر می‌رسد هم چون فیس تایم که تا مدتی متن باز شد، سوئیفت هم از حالت متن باز خارج شود یا همانند جاوا به صورتی محدود باشد. با این حال استفاده از این زبان در سیستم‌عامل دیگر، برای سوئیفت یک تغییر بزرگ خواهد بود.

اپل بدون نیت، خود به اجرای برنامه‌های سوئیفت در گنولینوکس کمک کرده است، با متن باز شدن زبان سوئیفت کاربران و توسعه‌دهندگان لینوکس به راحتی می‌توانند از نحوه کار آن آشنا شوند و یا حتی با استفاده از آن زبان‌های دیگر مانند Rust موزیلا را ارتقاء دهند. با این وجود برای اجرای برنامه‌های آی-اواس و اواس-ایکس در گنولینوکس نیاز به متن باز شدن کوکو (رابط برنامه‌نویسی) نیز هست. کوکو (Cocoa) رابط برنامه‌نویسی شیء‌گرای محلی اپل برای سیستم‌عامل اواس ده است. این کار لازم است؛ تا کاربران بتوانند به اجرای برنامه‌های آی-اواس پردازند. اپل در این مورد حرفی نگفته و یا نگارنده چیزی نشنیده است.

در هر صورت در استفاده از سوئیفت نیز نباید بسیار خوشبین بود! با این حال متن باز شدن زبان یا نرم‌افزارهایی این چنین که شرکت‌های بزرگ پشت آنان هستند برای مطالعه کد منبع آنان و استفاده از روش‌ها و الگوریتم‌های مورد استفاده آنان و بهره‌گیری درست از این روش‌ها باعث بهبود دیگر گزینه‌ها نیز خواهد شد. هرچند که برنامه‌نویسی این روزها بدون دیدن خطراتی از جانب یک وکیل کاملاً کم پیش می‌آید. ■

در این بین بنیاد موزیلا نیز بیکار نبوده و زبان Rust را توسعه می‌دهد. هر یک از این زبان‌های برنامه‌نویسی دارای قوت و ضعف مختص به خود هستند. هرچه باشد آینده در اختیار چنین زبان‌هایی است که به طور ایده‌آل اکثر آنان متن باز هستند. از گزینه اپل گرفته تا گزینه موزیلا.

مایکروسافت نیز سال گذشته اعلام کرد که چارچوب دانتنت را متن باز خواهد کرد و با نام «ویژوال استودیو کد»، قصد گسترش نفوذ ابزارهای توسعه نرم‌افزار خود به دیگر سیستم‌عامل‌ها دارد. مایکروسافت همچنین تلاش زیادی برای استفاده کاربران و توسعه‌دهندگان دیگر سیستم‌عامل‌ها از محصولات توسعه‌ای و زبان‌های خود دارد و در حال حاضر در حال تلاش برای فراهم کردن امکانی برای نصب برنامه‌های نوشته شده توسط دانتنت در سیستم‌عامل‌های آی-اواس و او-اس-ایکس است. به طوری که حتی بتوان برنامه‌های آی-اواس را در ویندوز و یا اواس-ایکس و با استفاده از ویژوال استودیو نوشته و به راحتی در آی-اواس اجرا کنند. بنابراین با چنین شرایطی اپل چاره‌ای نداشت جز این که زبان برنامه‌نویسی جدید خود را به یک زبان توسعه‌پسند، تبدیل کند.

با این حال اپل در استفاده از متن باز قریب نیست. آن‌ها از داروین که انشعابی از بی‌اس‌دی است بهره گرفته و سیستم‌عامل او-اس-ایکس را به یک سیستم‌عامل شبه-یونیکس تبدیل کردند در ضمن وب‌کیت را نیز با استفاده از توسعه پروژه متن باز کی‌اچ‌تی‌ام‌ال ایجاد کردند که موتور اصلی برنامه کانکور بود که با تبدیل به وب‌کیت به موتور اصلی سفاری و بعدها کروم مبدل گشت. همچنین «سامانه عمومی چاپ در یونیکس» CUPS هم ابزاری است متن باز که توسط اپل توسعه داده‌می‌شود و در سطح گسترده‌ای در توزیع‌های گنولینوکس مورد استفاده است.

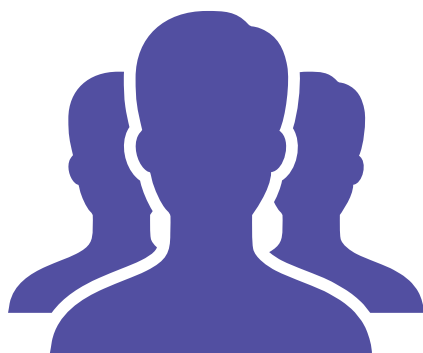
در هر حال اکثر توسعه‌دهندگان اپل از سوئیفت و Objective C استفاده خواهند کرد ولی اپل می‌خواهد تا با متن باز کردن سوئیفت سر مایکروسافت را در

مهمترین نکته کنفرانس توسعه‌دهندگان اپل در سراسر جهان تنها مختص به معرفی نسخه‌های جدیدی از اواس-ایکس و آی-اواس یا حتی سرویس جدید اپل با نام اپل موزیک نبود بلکه زمانی بود که معاون اپل مهندسی کریگ فدریگی به روی صحنه آمد و اعلام کرد که سوئیفت «Swift» زبان برنامه‌نویسی اختصاصی اپل متن باز گشته است. نکته هیجان‌انگیز ماجرا این است که اکثر توسعه‌دهندگان در حال حاضر به سمت توسعه محصولات با بهره‌گیری از ابزارهای آزاد / متن باز سوق یافته و استفاده از این ابزارها طی پنج سال گذشته افزایش قابل توجهی داشته است. همچنین اپل کاربران را به این سو سوق داده است که بهتر است از برنامه‌های خانگی اپل استفاده کنند و ابزار شخص ثالثی چون ادوبی فلش دوری گزینند چون معتقد است ابزاری بی‌منفعت هستند. اما به نظر نمی‌رسد حتی اپل هم بتواند جلوی گزینه‌های شخص ثالث بایستد؛ به این دلیل که ابزارهای شخص ثالث بسیاری در دسترس هستند که برای انجام امور توسعه‌ای در اپل به کار بیایند پس اپل برای جلوگیری از استفاده از دیگر ابزارها باید ابزار خود را متن باز می‌کرد.

با متن باز شدن سوئیفت توسط اپل، این شرکت به توسعه‌دهندگان در دیگر سیستم‌عامل‌ها این امکان را نیز داده است تا به ایجاد برنامه‌های خود برای سیستم‌عامل دیگر نیز با استفاده از کدهای سوئیفت پردازند؛ که منجر به ایجاد ویرایش مشابه از برنامه‌های کاربردی اپل در سیستم‌عامل‌های نوپایی مثل اوپنتو تاج نیز خواهد بود به این شکل که می‌توان کاری کرد که برنامه‌های کاربردی آی-اواس و نوشته شده در سوئیفت را در اوپنتو تاج اجرا کرد و یا هر سیستم‌عامل دیگر.

به طور قطع سوئیفت در حال رشد است اما زبان‌های برنامه‌نویسی مشابه دیگری نیز وجود دارند که در چند سال اخیر توسعه داده شده‌اند. به عنوان مثال فیس‌بوک با زبان‌های D و Hack در حال خطا و آزمون است و گوگل نیز زبان جدید Go را در آستین دارد.





جامعه کاربری

- تدابیری که دولت‌ها می‌توانند برای ترویج نرم‌افزار آزاد به کار گیرند | ۳۶
- نقش جامعه و دولت در حمایت از نرم‌افزار آزاد | ۳۸
- زندگی با شیطانک قرمز | ۴۰
- نامه سرگشاده مدیرعامل موزیلا به مدیرعامل مایکروسافت | ۴۵
- بهانه‌های استفاده نکردن از گنو/لینوکس | ۴۶
- صرفه‌جویی میلیون دلاری و دانش آزاد | ۴۸
- آیا نرم‌افزار امنیتی آزاد امنیت پایینی دارد؟ | ۵۰



تدابیری که دولت‌ها می‌توانند برای ترویج نرم‌افزار آزاد به کار گیرند



محمّد بنی‌شیر

آموزش است چرا که آینده کشور را به راحتی شکل می‌دهد:

❶ فقط نرم‌افزار آزاد آموزش داده شود:

تمام فعالیت‌های آموزشی یا حداقل آن‌هایی که توسط نهادهای دولتی ارائه می‌شوند باید تنها نرم‌افزار آزاد را تدریس کنند (بنابراین آن‌ها هرگز نباید دانش آموزان را به سمت استفاده از یک نرم‌افزار غیرآزاد ترغیب کنند) همچنین باید دلایل مدنی‌ای که باعث اصرار بر استفاده از نرم‌افزار آزاد می‌شود را به دانش آموزان آموزش داد چرا که آموزش نرم‌افزار غیرآزاد آموزش وابستگی است که این در تضاد با اهداف یک مدرسه است.

❷ دولت و مردم

همچنین سیاست‌های دولت در مورد اعمال نفوذ بر نرم‌افزارهایی که شرکت‌ها و سایر افراد استفاده می‌کنند نیز بسیار حیاتی است:

هرگز اجباری بر استفاده از نرم‌افزار انحصاری وجود نداشته باشد:

قوانین و شیوه‌های اجرای بخش دولتی باید تغییر کند به نحوی که آن‌ها هرگز سازمان‌ها و افراد مستقل را مجبور به استفاده از نرم‌افزار غیرآزاد نکنند یا بر آن‌ها فشار وارد نکنند. همچنین آنان باید جوامع و شیوه‌های عمومی‌ای که چنین پیامدی را به دنبال دارند دلسرد کند (مانند DRM).

❸ تنها نرم‌افزارهای آزاد توزیع شوند:

هنگامی که یک نهاد دولتی نرم‌افزاری را برای عموم منتشر می‌کند مانند نرم‌افزارهایی که در صفحات وب‌شان ارائه یا مشخص می‌شود این نرم‌افزارها باید به عنوان نرم‌افزار آزاد توزیع شوند و باید توانایی اجرا در بسترهای تماماً مبتنی بر نرم‌افزار آزاد را داشته باشند

❹ وب‌سایت‌های دولتی

وب‌سایت‌های نهادهای دولتی و خدمات مبتنی بر شبکه باید به گونه‌ای طراحی شوند که کاربران بدون لطمه خوردن بتوانند از آن‌ها استفاده کنند.

این مقاله سیاست‌هایی را برای تاثیرگذاری عمیق و پایدار تلاش‌هایی که برای ترویج نرم‌افزار آزاد در سطح دولت و به طبع آن هدایت کشور به سمت نرم‌افزار آزاد انجام می‌شود پیشنهاد می‌دهد

هدف یک دولت ساماندهی جامعه برای فراهم کردن آزادی و رفاه مردم است. یکی از جنبه‌های این مأموریت در زمینه محاسبات کامپیوتری، تشویق کاربران برای پذیرفتن نرم‌افزار آزاد است: نرم‌افزاری که آزادی کاربران احترام می‌گذارد.

یک نرم‌افزار انحصاری (غیرآزاد) آزادی کاربرانی را که از آن استفاده می‌کنند پایمال می‌کند. این یک مشکل اجتماعی است که دولت باید برای ریشه‌کن کردنش تلاش کند. دولت باید برای حفظ حاکمیت خود بر محاسباتش بر استفاده از نرم‌افزار آزاد اصرار کند. تمام کاربران مستحق این هستند که بر محاسباتشان کنترل داشته باشند اما دولت در مقابل مردم مسئول است که کنترل خودش را بر روی محاسباتی که از طرف مردمش (و با داده‌های آن‌ها) انجام می‌دهد حفظ کند. بخش زیادی از فعالیت‌های دولتی اکنون بر محاسبات کامپیوتری استوار است و کنترل این فعالیت‌ها در گرو کنترل کردن محاسبات صورت گرفته است. از دست دادن این کنترل در ادارهای که مأموریت‌های حیاتی دارد می‌تواند امنیت ملی را به مخاطره بیندازد.

مهاجرت ادارات دولتی به نرم‌افزار آزاد فواید ثانویه‌ای را نیز به دنبال دارد، مانند صرفه جویی در هزینه‌ها و تشویق و توسعه کسب و کارهای محلی مبتنی بر پشتیبانی نرم‌افزار.

در این متن «نهادهای دولتی» به تمامی بخش‌های دولتی مانند بخش‌های عمومی شامل مدرسه‌ها، نهادهای مشارکتی عمومی و خصوصی، فعالیت‌هایی که دولت پشتیبانی می‌کند مانند مدرسه‌های خصوصی و شرکت‌های خصوصی‌ای که توسط دولت کنترل می‌شوند یا با اجازه ویژه یا دستورات دولتی کنترل می‌شوند، اشاره می‌کند.

❺ آموزش

مهم‌ترین سیاست در این میان مربوط به بخش

تماما با استفاده از نرم‌افزار آزاد.

❻ قالب‌ها و پروتکل‌های آزاد

نهادهای دولتی تنها باید از قالب‌ها و پروتکل‌های ارتباطی‌ای استفاده کنند که به خوبی توسط نرم‌افزارهای آزاد پشتیبانی شوند ترجیحاً با مستندات و قواعد منتشر شده. (منظور ما از این قسمت «استانداردها» نیستند زیرا این قواعد باید در مورد اجزای منطبق با استاندارد و نیز اجزای غیر منطبق اعمال شوند)

❼ باز کردن بند مجوزها از پای کامپیوترها

فروش کامپیوترها نباید نیازمند خرید یک نرم‌افزار انحصاری باشد. فروشنده باید از طرف قانون ملزم به پیشنهاد گزینه خرید کامپیوتر بدون نرم‌افزار انحصاری و بدون پرداخت هزینه مجوز به خریدار باشد.

تحلیل هزینه‌ها اشتباه دیگری است که در این میان رخ می‌دهد و نباید توجه ما را از بی‌عدالتی ذاتی موجود در نرم‌افزارهای انحصاری منحرف سازد. از دست دادن آزادی‌ای که نتیجه استفاده از آن نرم‌افزارهاست. با این حال سواستفاده از اجبار کاربران برای پرداخت، برتری ناعادلانه دیگری را شامل حال توسعه‌دهندگان نرم‌افزارهای انحصاری می‌کند که برای آزادی کاربران مضر است. برای یک دولت شایسته است که جلوی چنین سواستفاده‌ای را بگیرد.

3 حاکمیت محاسباتی

برخی سیاست‌ها، حاکمیت محاسباتی دولت را به مخاطره می‌اندازند. نهادهای دولتی باید کنترلشان را بر محاسباتی که انجام می‌دهند حفظ کنند نه این که این کنترل را به بخش خصوصی واگذار نمایند. این اصول در مورد تمام کامپیوترها از جمله تلفن‌های هوشمند نیز صدق می‌کند.

3 مهاجرت به نرم‌افزار آزاد

نهادهای دولتی باید به نرم‌افزار آزاد مهاجرت کنند و نباید نرم‌افزارهای انحصاری را نصب و یا به استفاده از آن‌ها ادامه دهند مگر به شکل موقتی و زودگر. تنها یک نهاد باید قدرت اعطای مجوز استفاده از نرم‌افزار انحصاری به شکل موقتی را داشته باشد آن هم تنها در شرایطی که دلایل قانع‌کننده برای این کار وجود داشته باشد. هدف این نهاد باید به حداقل رساندن این استثنائات تا رسیدن به مرز صفر باشد.

3 توسعه راه حل‌های آزاد مربوط به فناوری اطلاعات

وقتی که یک نهاد دولتی برای توسعه یک راه حل محاسباتی هزینه می‌کند قرار داد باید پیمانکار را مجبور به ارائه نتیجه به عنوان نرم‌افزار آزاد نماید همچنین طراحی باید به شکلی باشد که یک شخص ثالث بتواند نرم‌افزار را بر روی یک سکوی صد در صد آزاد اجرا کند و توسعه دهد. این مورد باید در تمام قراردادهای الزامی باشد. در نتیجه اگر توسعه‌دهنده موافق این الزامات نبود نتوان برای آن کار هزینه کرد.

3 انتخاب کامپیوترها برای نرم‌افزار آزاد

وقتی که یک نهاد دولتی اقدام به خرید یا اجاره کامپیوتر می‌کند باید از بین مدل‌هایی که (در کلاس خودشان) نزدیک‌ترین سازگاری را برای کار بدون هیچ گونه نرم‌افزار انحصاری دارند دست به انتخاب بزنند. دولت باید برای هر رده از کامپیوترها یک فهرست مجاز با توجه به این معیار تهیه کند. مدل‌هایی که هم برای دولت و هم عموم مردم در دسترس باشند باید به مدل‌هایی که تنها برای دولت در دسترس هستند ترجیح داده شوند.

3 مذاکره با تولید کنندگان

دولت باید به شکل فعال در تمامی حوزه‌های مربوطه با تولید کنندگان در مورد ارائه محصولات سخت‌افزاری قابل قبولی که نیازمند اجرای نرم‌افزار انحصاری نباشند در بازار (هم برای بخش دولتی و هم برای عموم مردم) مذاکره کند.

3 اتحاد با سایر دولت‌ها

دولت باید از سایر دولت‌ها نیز برای مذاکره به شکل جمعی با تولید کنندگان در مورد محصولات سخت‌افزاری مورد قبول دعوت به عمل آورد چرا که احتمال موفقیت به شکل جمعی بیشتر است.

3 حاکمیت محاسباتی II

حاکمیت محاسباتی (و امنیتی) دولت شامل کنترل بر کامپیوترهایی که کارهای دولتی را انجام می‌دهند نیز می‌شود. این حاکمیت شامل پرهیز از استفاده از خدمات نرم‌افزار به عنوان سرویس دهنده (مگر این که سرویس توسط یک نهاد دولتی مربوط به همان شاخه از دولت ارائه شود) و همچنین سایر شیوه‌هایی که از میزان کنترل دولت بر محاسباتش می‌کاهد خواهد شد.

3 دولت باید کامپیوترهای خودش را کنترل کند

هر کامپیوتری که دولت استفاده می‌کند باید متعلق به همان شاخه از دولت باشد یا توسط همان شاخه اجاره شده باشد و آن شاخه نیز نباید به مجموعه‌های دیگر حق تصمیم‌گیری در مورد این که چه کسی دسترسی فیزیکی به کامپیوتر داشته باشد، چه کسی مسئولیت نگهداری (سخت‌افزاری یا نرم‌افزاری) را داشته باشد یا چه نرم‌افزاری باید روی آن نصب شده باشد را اعطا نماید. اگر کامپیوتر قابل حمل نباشد در آن صورت در هنگام استفاده باید در محلی قرار گرفته باشد که در تملک دولت باشد (به شکل ملکی یا اجاره‌ای)

3 ترغیب به توسعه

سیاست‌های دولت بر توسعه نرم‌افزارهای آزاد و غیرآزاد تاثیر گذار است :

3 پشتیبانی از نرم‌افزار آزاد

دولت باید با روش‌هایی توسعه‌دهندگان را برای ایجاد یا بهبود نرم‌افزارهای آزاد و فراهم کردن آن برای عموم تشویق کند. مانند معافیت‌های مالیاتی و سایر انگیزه‌های مالی. به صورت مخالف، این انگیزه‌ها نباید برای توسعه، توزیع یا استفاده از نرم‌افزارهای انحصاری وجود داشته باشند.

3 عدم حمایت از نرم‌افزار غیر آزاد

به شکل خاص، توسعه‌دهندگان نرم‌افزارهای انحصاری نباید توانایی اهدای نسخه‌هایی از نرم‌افزارشان را به مدارس و در قبال آن تقاضای

تخفیف مالیاتی در قبال ارزش اسمی نرم‌افزارشان را داشته باشند. نرم‌افزارهای انحصاری نباید در مدارس قانونی باشند.

3 ضایعات الکترونیکی (E-waste)

3 نرم‌افزارهای قابل جایگزینی

بسیاری از کامپیوترهای مدرن به گونه‌ای طراحی شده‌اند که جایگزین کردن نرم‌افزارهای پیش‌فرض در آنان با نرم‌افزارهای آزاد غیرممکن باشد بنابراین تنها راه آزاد سازی آن‌ها دور انداختن آن‌هاست. این عمل برای جامعه مضر است. بنابراین این کار باید غیرقانونی باشد یا حداقل به شکل قابل ملاحظه‌ای با استفاده از مالیات‌های سنگین محدود شود. این محدودیت باید بر روی فروش، واردات یا توزیع عمده کامپیوترهای نو (نه دست دوم) یا محصولات کامپیوتری‌ای که در مورد اتصالات سخت‌افزاری شان پنهان کاری می‌کنند یا برای توسعه، نصب و یا جایگزینی کل سیستم یا هریک از نرم‌افزارهای نصب شده که تولید کننده توانایی به‌روزرسانی آن را داشته باشد محدودیت‌های عمده ایجاد می‌کنند اعمال شود. به طور خلاصه تمام دستگاه‌هایی که برای نصب یک سیستم عامل متفاوت نیاز به «jailbreaking» دارند یا آن‌هایی که اتصالات مورد استفاده برای برخی لوازم جانبی شان مجهول است.

3 بی‌طرفی در فناوری

با تدابیر ذکر شده در این مقاله، دولت می‌تواند کنترل خویش بر محاسباتش را بازگرداند و شهروندان، کسب و کارها و سازمان‌های کشور را به سوی کنترل بر روی محاسباتشان رهنمون سازد. هرچند مخالفت‌هایی وجود دارد که این امر اصل بی‌طرفی در مورد فناوری را نقض می‌کند.

ایده بی‌طرفی در فناوری این است که دولت نباید به شکل دلخواه تمایلات خودش را در موارد تکنیکی اعمال کند. هر چند درستی این اصل اعتراض‌پذیر است اما این اصل به مواردی محدود است که تنها مشکل موجود در آن‌ها مشکل فنی است. تدابیری که در این مقاله از آن‌ها دفاع شده در مورد مسائل اخلاقی، اجتماعی و خطرات سیاسی هستند. در نتیجه این رهنمودها خارج از چهارچوب بی‌طرفی در فناوری هستند.

تنها کسانی که خواستار درهم شکستن یک کشور هستند می‌توانند پیشنهاد دهند که دولت‌شان در مورد حاکمیتش یا آزادی شهروندانش بی‌طرف باشد. ■





نقش جامعه و دولت در حمایت از نرم افزار آزاد

بین کشورها پررنگ تر می شود اما نرم افزار آزاد بدون آن که شعار جهانی سازی را داده باشد، به کمک اینترنت مردم تمام دنیا را به هم متصل کرده است. بالاترین شکل تعامل، تعامل برای رسیدن به یک هدف مشترک است و این در نرم افزار آزاد نمود پیدا کرده است. زیرا جامعه نرم افزار آزاد خلق یک نرم افزار را برای رفع یک نیاز مشترک، هدف قرار داده است و امروزه نمی توان روی نرم افزارهای آزاد برچسب کشور یا شرکت به خصوصی را زد.



جامعه نرم افزار آزاد خلق یک نرم افزار را برای رفع یک نیاز مشترک، هدف قرار داده است و امروزه نمی توان روی نرم افزارهای آزاد برچسب کشور یا شرکت به خصوصی را زد.

شناخت جایگاه جامعه و دولت در حمایت از نرم افزار آزاد

دولت ها، جامعه و شرکت ها انگیزه های متفاوتی برای حمایت از نرم افزار آزاد دارند چرا که نیاز آن ها برای استفاده از نرم افزار آزاد متفاوت است. اگر نیاز هر کدام به درستی تشخیص داده شود می توان درباره نقاط اشتراک آن ها بحث کرد و این اهداف را در راستای هم قرار داد. هدف دولت ها از استفاده از نرم افزار آزاد معمولاً به امنیت بیشتر اطلاعاتشان مربوط می شود. شرکت ها به

استفاده از سیستم عامل اوبونتو روی کامپیوترهای دستکاپ کارکنان سازمان فناوری اطلاعات یک گام رو به جلو در راستای استفاده از نرم افزار آزاد بوده است. افزایش آگاهی جامعه از طریق برگزاری همایش ها یا حمایت از گروه هایی نظیر لاگ تهران می تواند در راستای ترویج استفاده از نرم افزار آزاد باشد. حمایت از سیستم عامل بومی زمین هم می تواند در راستای توسعه نرم افزار آزاد شناخته شود.

جهانی سازی

نرم افزار آزاد بخشی از فرایند جهانی سازیست. جهانی سازی به معنای متصل کردن تمام مردم دنیا به یکدیگر و حذف مرزهای جغرافیایی. یعنی فراهم کردن شرایطی که مردم دنیا فارغ از زبان، ملیت، مذهب و... بتوانند با یکدیگر تعامل داشته باشند و در کنار هم و با هم زندگی کنند. کشورهای توسعه یافته که اغلب شعار جهانی سازی می دهند، برخلاف ادعای خود عمل کرده اند چرا که روز به روز سفر به کشورهای توسعه یافته سخت تر می شود و مرزهای

در این نوشتار سعی شده ابتدا به صورت اجمالی وضع کنونی، اهداف و دستاوردهای دولت در استفاده، ترویج و توسعه نرم افزار آزاد در ایران تحلیل شود. در «شناخت جایگاه دولت و جامعه در حمایت از نرم افزار آزاد» این اهداف با واقعیت مقایسه شده و در نهایت «راهکار پیشنهادی» نوشته شده است.

اهداف و دستاوردهای دولت

اگر دستاوردهای سازمان فناوری اطلاعات را در راستای اهداف دولت در نظر بگیریم می توان این اهداف را در سه دسته کلی تقسیم بندی کرد. دسته اول استفاده از نرم افزار آزاد است. حتی از واژه به کارگیری در نام این تشکیلات استفاده شده است، که نشان دهنده اهمیت این هدف برای سازمان فناوری اطلاعات است. دسته دوم ترویج استفاده از نرم افزار آزاد است به این معنی که دیگران هم از نرم افزار آزاد استفاده کنند و دسته سوم توسعه نرم افزار آزاد است. پیش از این که به بررسی خود این اهداف بپردازیم برای هر کدام یک مثال بارز خواهیم آورد.



سیدعلی حاجی نوریستاده



مسعود صدرزاد نویسنده

دنبال راه‌حل‌های کم‌هزینه‌تر و قابل اعتمادتر برای حل مسائل خود هستند و جامعه به دنبال راهکاریست که نیازهای خود را برطرف کند

بخش قابل توجهی از قدرت و ثروت به ترتیب در اختیار دولت‌ها و شرکت‌هاست، در حالی که توسعه نرم‌افزار آزاد نیازی به قدرت و ثروت ندارد. به این معنی نیست که این دو نمی‌توانند به توسعه نرم‌افزار آزاد کمک کنند بلکه به این معنیست که نرم‌افزار آزاد بدون قدرت دولت‌ها و ثروت شرکت‌ها هم می‌تواند توسعه یابد همان‌طور که بدون این دو به جایگاه کنونی خود رسیده است.

اگر دولت متوجه سهم کم قدرت در توسعه نرم‌افزار آزاد باشد، جایگاه خود را تشخیص می‌دهد و به اندازه سهم خود و در جای مناسب، مسئولیت اجتماعی خود را ایفا می‌کند. اگر چنین شود، دولت متوجه می‌شود که مجبور کردن افراد یک سازمان دولتی به نصب اوبونتو روی کامپیوترهای شخصی خود، کمکی به توسعه نرم‌افزار آزاد نمی‌کند، به دلیل این که استفاده افراد به عنوان کاربر نهایی از یک سیستم‌عامل آزاد نقش چندان موثری در توسعه نرم‌افزار آزاد نخواهد داشت. همچنین ایجاد یا حمایت از ایجاد سیستم‌عامل زمین به دلیل عدم استفاده از این سیستم‌عامل توسط شرکت‌های مبتنی بر ابر، چه در داخل و چه در خارج کشور کمکی به توسعه نرم‌افزار آزاد نمی‌کند نه به این خاطر که لزوماً این سیستم‌عامل ضعیف عمل کرده یا کم هزینه شده، بلکه به این دلیل که این سیستم‌عامل از ابتدا هم قرار نبوده توسط مردم جهان مورد استفاده قرار بگیرد و یک سیستم‌عامل ملی/بومی بوده. این در حالیست که یکی از افتخارات دولت استفاده از سیستم‌عامل غیربومی اوبونتو است که بخش قابل توجهی از توسعه آن توسط جامعه کاربری آن از سراسر جهان صورت می‌گیرد. ابهامی که در واژه ملی قرار دارد آیا ناشی از عدم شناخت ماهیت نرم‌افزار آزاد و ارتباط آن با فرایند جهانی سازی است؟ آیا اجباری در کار است که دستگاه‌های دولتی از این اسامی برای دریافت تایید نام سازمان و پروژه‌های آزاد استفاده می‌کنند؟

3 راهکار پیشنهادی

در بخش قبل گفتیم نرم‌افزار آزاد به قدرت و ثروت نیاز ندارد. نرم‌افزار آزاد به جای این دو به جامعه مشارکت‌کننده نیاز دارد. نرم‌افزارهای آزاد در سراسر دنیا توسط جوامع کاربری تولید، آزمایش و منتشر می‌شود و شرکت‌ها و دولت‌ها تنها حامی این جوامع و گروه‌های کاربری هستند. پرسشی که در این‌جا مطرح می‌شود اینست که پس دولت چه نقشی می‌تواند در توسعه نرم‌افزار آزاد داشته باشد؟ به این پرسش در دو بخش جداگانه پاسخ می‌دهیم. بخش اول شناسایی نیازهای جوامع و رفع بخشی از آن که در توان دولت می‌باشد و بخش دوم تالیف سیاست‌های درست است.

3 شناسایی نیازهای جوامع و رفع بخشی از آن‌ها

برای مورد دوم به یک دایرکتوری از پروژه‌های نرم‌افزار آزاد و دانش آزاد از سراسر دنیا نیاز هست برای همین ابتدا باید معیارهایی مشخص شود که بر اساس آن هم آزاد بودن پروژه‌ها و هم صلاحیت آن‌ها چه از نظر فنی و چه از نظر توان اجرایی مورد بررسی و تایید قرار بگیرد. بعد از تهیه دایرکتوری گام بعد مشخص کردن نیازهای هر پروژه و حیطة تخصصی هر کدام هست. برای مثال سایت گنو، لانچپد، داک‌داک‌گو، ویکی‌پدیا و... امکان ترجمه به صورت گروهی را فراهم کرده‌اند و هر فردی که به زبان انگلیسی مسلط باشد، می‌تواند به این پروژه‌ها کمک کند. ممکن است بعضی از پروژه‌ها نیازمند گرافیکست، بعضی ایجاد مستندات فنی، بعضی برنامه‌نویس یا حتی حمایت مالی (سیستم دونیشن را هم شامل می‌شود) و... داشته باشند. بعد از طبقه‌بندی نیازها باید برای هر پروژه و تخصص راهنمایی تهیه کرد که شامل روال مشارکت در آن پروژه باشد. البته تمام نیازها در این طبقه‌بندی قرار نمی‌گیرد. برخی نیازها ممکن است حالت خاص‌تری داشته باشد ولی نکته مهمی که وجود دارد اینست که نیاز را افرادی تشخیص می‌دهند که در این پروژه‌ها کار می‌کنند نه نگارندگان این مقاله یا کارکنان سازمان‌های دولتی.

جمع‌بندی مراحل پیشنهادی برای ایجاد بستر شناسایی نیازهای جوامع و حمایت از آن‌ها بدین است:

- ❖ صفر- مشخص کردن معیارها و حداقل‌های مورد نیاز برای آزاد بودن یک پروژه و صلاحیت فنی و اجرایی آن
- ❖ یک- تهیه دایرکتوری جوامع و پروژه‌ها و معرفی آن‌ها
- ❖ دو- دسته‌بندی جوامع فعال در حوزه نرم‌افزار یا دانش آزاد
- ❖ سه- بررسی نیازها از طریق پرس‌وجو از فعالین آن حوزه
- ❖ چهار- طبقه‌بندی نیازها بر حسب تخصص‌ها یا منابع مالی مورد نیاز

3 سیاست‌ها

هرچند مهاجرت دولت و توابع آن از جمله مدارس و سازمان‌ها و شرکت‌های دولتی به نرم‌افزار آزاد می‌تواند فوایدی از جمله ذخیره شدن منابع مالی و تشویق و گسترش شرکت‌های پشتیبان نرم‌افزار در داخل کشور، اثر نکردن فشارهای دولت‌ها برای تحریم ایران توسط شرکت‌های انحصاری و... داشته باشد اما مأموریت اصلی دولت سازماندهی جامعه برای آزادی و سعادت مردم است. یکی از جنبه‌های این آرمان در حیطة کامپیوتر تشویق کاربران به فراگیری نرم‌افزار آزاد یعنی نرم‌افزارهایی که به آزادی کاربران احترام می‌گذارند، است در حالی که نرم‌افزار انحصاری آزادی کاربران را پایمال می‌کند و این یک مشکل اجتماعیست که دولت باید آن را ریشه کن کند.

مهمترین سیاست در حوزه آموزش مطرح است زیرا آموزش آینده یک کشور را می‌سازد. فعالیت‌های آموزشی و افراد مرتبط با امور آموزشی باید فقط نرم‌افزارهای آزاد را تدریس کنند و نباید هیچوقت دانش‌آموزان را به استفاده از نرم‌افزارهای انحصاری عادت دهند. این افراد باید دلایل اجتماعی پافشاری روی نرم‌افزار آزاد را آموزش دهند چرا که آموزش نرم‌افزار انحصاری آموزش وابستگیست که این با مأموریت مدارس در تضاد است.

3 جستارهای وابسته

آخرین نسخه مقاله به همراه اسلایدهای این مقاله، ارائه شده در نشست هم‌اندیشی توسعه بکارگیری نرم‌افزار آزاد-متن‌باز در کشور در ویکی بنیاد دانش آزاد در دسترس علاقه‌مندان قرار گرفته است. ■



مجبور کردن

افراد یک

سازمان دولتی به

نصب اوبونتو روی

کامپیوترهای

شخصی خود،

کمکی به توسعه

نرم‌افزار آزاد

نمی‌کند، به دلیل

این که استفاده

افراد به عنوان

کاربر نهایی از

یک سیستم‌عامل

آزاد نقش چندان

موثری در توسعه

نرم‌افزار آزاد

نخواهد داشت





یک اتاق کوچک خنک و پر نور و درخشان در شرکت ایمن پردیس خانه اول بابک فرخی است. ترجیح می‌دهد به جای سر و کله زدن با آدم‌ها یا وقت گذراندن در شبکه‌های اجتماعی در این اتاق کوچک پشت آی‌مک ۲۷ اینچی‌اش بنشیند و آخرین پیش‌نویس‌های IETF را مطالعه کند و نظرش را درباره استاندارد «مانیتورینگ کیفیت شبکه‌های موبایل در ابعاد بزرگ» بنویسد یا در کنار کارمند روسی nginx و کارمند آمریکایی netflix در بهبود عمل‌کرد الگوریتم‌های IP forwarding به «پیرمردهای بی‌اس‌دی» کمک کند. تا این حد که همسرش می‌گوید: «گاهی اوقات بچه‌ها هم می‌ترسند به اتاقش بروند که مبدا آن خلوت را به هم بزنند». کنار دستش یک لپ‌تاپ سونی نیز قرار دارد که هنوز منتظر آپدیت ویندوز ده‌اش است. همان لحظه با خوشحالی در حال توسعه سیستم‌عامل شرکت نیز هست و وقتی تمام این کارهای بزرگ را نه مانند یک کار دشوار بلکه مانند یک سرگرمی نشاط‌آور شرح می‌دهد. کتاب‌هایی زیادی سمت راست میزش روی هم چیده شده از فرهنگ سامورایی تا جامعه‌شناسی و تاریخ باستا و فلسفه و رمان را می‌توان در میان آن‌ها مشاهده کرد. یک کیندل در شرکت دارد و یکی در خانه و می‌گوید: «برای من وظیفه است که هر شب قبل از خواب کتاب بخوانم»

به نظر بابک فرخی لینوکس برای یک محیط «کاتینگ‌اج» بوده و برای توسعه و پیشروی مناسب‌تر است. او هرگز در جمع‌های لینوکسی پا نگذاشته و خودش می‌گوید اجتماعی نیست. روی

گفتگو با بابک فرخی توسعه‌دهنده سولاریس و BSD و مرور فناوری در دهه هفتاد

زندگی با شیطانک قرمز

به هر شرکتی که پا می‌گذاشت دیگر حتی لازم نبود توضیحی بدهد. کافی بود بگوید تمام سیستم‌های سازمان را به BSD تغییر دهید تا بدون هیچ شرط و شروطی آن را بپذیرند. پشتوانه سال‌ها کار تخصصی در پروژه‌های داخلی و ارتباط با چهره‌های بین‌المللی او را در حوزه کاری خودش آوازه خاص و عام کرده بود. به قول خودش: «همیشه برای همه مرموز و عجیب بودم».



نویسنده:
امیر حسین حسینی‌بازوه

هیچ نرم‌افزاری تعصب ندارد. جمله‌ای که بارها برای ابزارهای مختلف تکرار می‌کند این است که «من هم خیلی باهوش خوشحالم» و این تنها دلیلش برای کار کردن با نرم‌افزار یا سخت‌افزاری است. به قول خودش ترجیح می‌دهد به جای آدم‌ها با ماشین‌ها سر و کله بزند «۳۶ ساعت می‌شینم پای کامپیوترم که یک مشکلی را حل کنم و واقعا هم خسته نمی‌شم و هر چه جلوتر می‌روم موتورم گرم‌تر می‌شود ولی حرف زدن با آدم‌ها و مذاکره و فروش اذیت می‌کنم ترجیح می‌دهم در را ببندم و یک مشکل جهانی را حل کنم».

آپادانا

اوایل دهه ۷۰ است و بابک سال‌های دبیرستان را سپری می‌کند. روزهایی که رایانه محصولی نو در خانه‌های ایرانی به حساب می‌آید و به راحتی قابل خرید نیست. اولین رایانه عمرش که یک آمیگا ۵۰۰ است را در خانه یکی از دوستانش می‌یابد و هرگز فکرش را نمی‌کرد ۲۴ سال بعد فرزندان همین دستگاه بهترین همدمش شوند. آن زمان وقتی کسی رایانه‌ای را می‌خرد نه خبری از فیلم‌های با کیفیت بود نه اینترنتی بنابراین بیشتر از آن که یک رایانه جنبه مصرفی داشته باشد دستگاهی بود که افراد کمی از آن برای ساخت برنامه استفاده می‌کردند. می‌گوید: «همیشه دوست داشتم یک چیزی بسازم» برای همین سعی می‌کند یکی از زبان‌های رایانه‌ای را یاد بگیرد و اتفاقی با زبان اسمبلی موتورولا آشنا می‌شود. می‌گوید: «ز شانس بد من اولین زبانی که یاد گرفتم اسمبلی بود، زبان سختی است اما برای من هیجان انگیز بود». برنامه‌نویسی با این زبان را شروع می‌کند و اتفاقاً نرم‌افزارهای جالب و خوبی هم می‌نویسد.

مدتی بعد اولین رایانه شخصی را می‌بیند. یک PC XT ۸۰۸۸ است. می‌گوید: «مال یکی از اقوام بود رفتیم در خیابان جمهوری یک شرکتی که دو سه نفر از آمریکا آمده بودند و رایانه‌های ۲۸۶ دست دوم می‌فروختند با مانیتور EGA که رزولوشنش ۳۲۰ در ۲۰۰ بود. الان فکر کنم یک اپل واچ از این بیشتر باشد». در اولین کار می‌خواهد برای آن هم با اسمبلی برنامه بنویسد اما متوجه می‌شود معماری آن با موتورولا فرق می‌کند به هر حال کم‌کم شروع یادگیری اسمبلی اینتل می‌کند. می‌گوید: «کتابی مال پیتر نورتون بود که من سمپل‌ها رو می‌نوشتیم بعد توسعه می‌دادم، فایل منیجر، ادیتور و امثال این‌ها را روی سیستم عامل داس ۵ و داس ۶ می‌نوشتیم». مدتی بعد یکی از دوستانش مسیر یادگیری او را تغییر داده و زبان سی را به او معرفی می‌کند. یک

کتاب فارسی همدم او در یادگیری این زبان جدید می‌شود. می‌گوید: «دیدم چقدر من با اسمبلی مکافات می‌کشیدم» با turbo C و از آن طریق با نرم‌افزار دیگر همین مجموعه یعنی turbo pascal آشنا می‌شود و شروع به یادگیری پاسکال می‌کند. تمام این اتفاقات در طی دو الی سه سال روی می‌دهد و تنها منبع‌اش هم کتاب‌های این و آن است. می‌گوید: «یک کتاب بنفش رنگ انگلیسی به من داد که پاسکال ۵/۵ را یاد می‌داد دیدم چقدر از سی راحت‌تر است» و مدتی بعد از طریق دوستان دبیرستانش با بیسیک آشنا می‌شود. در حقیقت به قول خودش مسیر یادگیری برنامه‌نویسی را برعکس می‌رود.

در دوران دبیرستان یکی از دوستان بابک که در الکترونیک سر رشته دارد بردی طراحی می‌کند. این برد به پورت پرینتر وصل می‌شود و با آن می‌توان دو رایانه را از طریق باند FM به هم وصل کرد و از این طریق فایل جابجا کرد. دستگاهی که کارکردش برای الان هم عجیب است. «ما اون موقع FM مودولاسیون می‌دانستیم و روی پورت پرینتر هم بلد بودیم برنامه‌نویسی کنیم برنامه را با کوییک بیسیک نوشتیم که اینترفیسی که با پورت پرینتر صحبت می‌کرد با اسمبلی بود» به همین نحو سال‌های انتهایی دبیرستان با خرده‌فعالیت‌هایی در زمینه برنامه‌نویسی طی می‌شود.

آن روزها اینترنتی در کار نبود اما شبکه‌هایی به نام BBS یا Bulletin board system وجود داشتند که رایانه‌ها را از طریق ترمینال به هم متصل می‌کرد. او با مودمش که ۲۴۰۰ بیت بر ثانیه سرعت داشت به شبکه بی‌بی‌اس مروا وصل می‌شد. می‌گوید: «وقتی آدم یاد این خاطرات می‌افتد خنده‌دار به نظر می‌آید اما ما با این مودم چه کارا که نکردیم و چه فایل‌ها که از این BBS دانلود نکردیم»

به سرعت پای اینترنت به ایران باز می‌شود اما اینترنت آفلاین. بابک فرخی که دبیرستان را تمام کرده یک روز در مجله رایانه با سرویس ایمیل آپادانا آشنا می‌شود و بعد از تماس با سپهر محمدی مدیر این شرکت تک نفره در خیابان پامنار در حوالی بازار تهران ملاقات می‌کند. اولین تصورش از شرکت را این‌طور می‌گوید: «دیدم یک مرد است با دو کامپیوتر و دو میز» این شرکت به مردم آدرس ایمیل می‌دهد و روزی یک بار به شبکه دایال‌آپ آمریکا وصل شده و ایمیل‌ها را می‌فرستاد و می‌گرفت و کیلو بایتی ده تا ۱۵ تومن پول دریافت می‌کرد. این آشنایی ناگهانی باعث می‌شود بابک فرخی ادامه تحصیل را رها کرده و به دنبال علاقه‌مندی و رؤیای خود در آپادانا شروع به کار کند. خانواده هم که عشق و علاقه بابک را که

می‌بینند تسلیم تصمیم او می‌شود و به او برای ادامه تحصیل اصرار نمی‌کنند.

سال ۷۴ است. شرکت آپادانا سعی می‌کند فعالیت‌های قبلی خود را توسعه دهد. بابک روی ویندوز NT 3.5 یک برنامه می‌نویسد که ایمیل‌های دریافت شده شرکت آپادانا در روز را اصطلاحاً دیس‌پچ کرده و همه را در قالب یک فایل در می‌آورد آن‌ها این فایل را با FTP می‌گرفتند. آن زمان ویندوز 3.1 شبکه DialUp و استانداردهای TCP/IP را نداشت بلکه با یک برنامه ثالث به نام trumpet این امکان اضافه می‌شد و کاربران با برنامه pegasus mail ایمیل‌هایشان را می‌نوشتند و به شبکه آپادانا می‌فرستادند. بابک هم به همراه سپهر هر شب آن‌ها را با telnet روی یک ماشین یونیکس در آمریکا با FTP آپلود می‌کردند می‌گوید: «SSH اختراع نشده بود فایل را می‌گذاشتیم با اجرا کردن یک کامند ایمیل‌ها می‌رفت بعد جوابش که می‌آمد سر یک ساعت دانلود می‌کردیم». می‌گوید: «حالا بگذریم که چقدر باگ داشت و ایمیل یکی را در میل باکس دیگری می‌گذاشتیم» و این ۵۰ کاربر می‌دانستند که ساعت شش بعد از ظهر باید وصل شوند و ایمیل‌هایشان را بگیرند. می‌گوید: «۸ تا مودم داشتیم با ۸ تا خط تلفن گاهی کسی تاجر بود و زنگ می‌زد که تو رو خدا الان وصل شوید ایمیل مهم دارم».

این دوران به سرعت باد می‌گذرد و پای اینترنت ماهواره‌ای به کشور باز می‌شود. از شرکت داده‌پردازی یک خط ارسال کننده دایال‌آپ و یک دریافت‌کننده ماهواره‌ای می‌خرند. ایمیل آنلاین می‌شود ولی اینترنت هنوز آفلاین است برای همین با سرویس‌هایی به نام web to mail gateway این مشکل را بر طرف می‌کنند. کافی بود به یک آدرس خاص URL را ایمیل کنید تا فایل‌های اچ‌تی‌ام‌ال را به ایمیل اتچ کرده و برای شما بفرستد. مدتی بعد که کارشان می‌گیرد دفتر را از بازار به قیطریه منتقل می‌کنند و تعدادشان به ۴ نفر افزایش می‌یابد.

مکعب مرموز

یک روز یک مکعب کوچک مسیر زندگی حرفه‌ای بابک را تغییر می‌دهد. یکی از شرکت‌های همکار با یکی از کامپیوترهایش به مشکل می‌خورد و آن را برای آپادانا می‌فرستند تا شاید آن‌ها بتوانند دستگاه را درست کنند. مشکل از این قرار است که فلاپی دیسک در دستگاه گیر کرده و هیچ دکمه‌ای هم روی آن نیست تا آن را با فشردن دکمه خارج کنند. ورود ناگهانی یک SPARCstation IPX به مجموعه آن‌ها که تشنه آموختن هستند اوضاع را به هم می‌ریزد. یک هفته با این دستگاه کار می‌کنند تا متوجه مشکل



آن‌ها را به صورت اتفاقی به لینوکس سوزی می‌رساند. ملیت و زبان آلمانی این توزیع در دسر ساز می‌شود. از روی ناچاری و با علاقه شروع به یادگیری آلمانی می‌کند تا شاید زبان این توزیع را بفهمد و بتواند طعم لینوکس را بچشد. مدتی بعد با پذیرش شکست اما همچنان مشتاق به دنبال توزیع دیگری می‌رود و با Slackware آشنا می‌شود. این نخستین توزیعی بود که توانست در سال ۷۵ به لطف محلی که در آن کار می‌کرد از اینترنت دانلود کند. آن زمان روش نصب یک توزیع به این صورت بود که ابتدا باید یک کرنل یک و نیم مگابایتی را از روی فلاپی بوت می‌کردند تا آن هسته لینوکس، درایور سی‌دی رام دستگاه را لود کند. می‌گوید: «کرنل ۲ تازه آمده بود، برای همین کار چهار پنج روز سر و کله زدیم» لینوکس هیچ کدام از سخت‌افزارهای آن‌ها را نمی‌شناسد و با مصیبت یک کارت شبکه Novel NE2000 از بازار رضا می‌گیرند تا در نهایت به اینترنت وصل می‌شوند. کارت گرافیک هم شناسایی نمی‌شود. با خنده می‌گوید: «هی ما کامپایل می‌کردیم از اون موقع من کامپایل کردن ام خوب شد» در همین روزها یکی از دوستانش یک ردهت نسخه ۴ را از آمریکا می‌آورد. می‌گوید: «رابط کاربری‌اش بهتر بود و کمی آدم حسابی‌تر بود هنوز

توضیح می‌دهد. می‌گوید: «مثلا سان فایر ۲۵۰۰۰ یک چیزی اندازه یخچال شاید بای ساید است که درش باز می‌شود و می‌توانید داخلش شوید مثلا ۲۵۶ سی‌پی‌یو دارد، با کنسول کار می‌کنید که می‌گوید سی‌پی‌یو شماره ۵۳ از کار افتاد روی هر سیستم دیگر وقتی سی‌پی‌یو از کار می‌افتد یعنی سیستم از کار افتاده، اما این جا شما سی‌پی‌یو را عوض می‌کنید دوباره روی کنسول می‌نویسد سی‌پی‌یو شماره ۵۳ آنلاین شد. دستگاه‌های سان معروف است که وقتی روشنش می‌کنی ۴۰ دقیقه طول می‌کشد بالا بیاید و دیگر خاموشش نمی‌کنی» درباره مزایای سولاریس می‌گوید: «دستگاه سی‌تی اسکنی trusted solaris وجود ندارد که درش سولاریس نباشد یا تراست سولاریس سیستم‌عاملی است که پنتاگون استفاده می‌کند برای اون قدیمی‌تری که با سولاریس کار کرده لینوکس عجیب است لینوکس یک هیپی به حساب می‌آید وقتی پیشش از مزایای لینوکس صحبت کنید ناراحت می‌شود»

با همکاری سپهر محمدی سعی می‌کنند یونیکس را روی رایانه‌های معمولی PC هم نصب کنند برای همین راهی مغازه‌های تهران می‌شوند و در یک مغازه سی‌دی فروشی در چهارراه ولی عصر دست تقدیر

به وجود آمده می‌شوند. سیستم‌عامل این دستگاه Sun OS ۲/۶ بود و برای خروج فلاپی باید دستور eject را در کامندلاین می‌نوشتند. با این حال طعم یک فناوری جدید و یک دنیای متفاوت از ویندوز برای آن‌ها شیرین است و سعی می‌کنند با زیرکی آن را مدت بیشتری پیش خود نگه دارند. چشمانش برقی می‌زند و با شور می‌گوید: «خیلی هیجان انگیز بود» و واژه «خیلی» را چند ثانیه می‌کشد با این امید که ما هم میزان اشتیاقش در آن روز را درک کنیم.

شروع به یادگرفتن سولاریس می‌کنند و تصمیم می‌گیرند یک وب سرور برای آن بنویسند چون نمونه اصلی چندان راضی کننده نیست. از آن به بعد تا امروز دورادور سولاریس را دنبال کرده و تا قبل از سال ۹۰ هر کس که پروژه‌ای در این زمینه دارد پیش او می‌آید. می‌گوید: «من خیلی به سان علاقه داشتم چون فائدر سان آقای بیل جوی را دوست دارم، بیل جوی یک آدم خیلی باهوش است که Cshell و Vim را ساخت انصافا هم سولاریس یک سیستم‌عامل پدر مادر دار روی سخت‌افزار پدر مادر دار است» رم، سی‌پی‌یو و هارد دستگاه‌های سولاریس هات پلاگ است یعنی می‌توان اجزای رایانه را در حالی که روشن است تغییر داد. این مفهوم را خودش خیلی بهتر

سی‌دی‌اش را دارم»

با این حال ترجیح می‌دهند سیستم‌های شرکت روی ویندوز NT بمانند و برنامه‌ها را تحت ویژوال استودیو توسعه دهند. می‌گوید: «بعد که به شرکت دیگری رفتم کم‌کم این اعتماد به نفس را پیدا کردم که ساختار را روی یونیکس بیاورم» هرچند با جاوا هم برنامه‌نویسی می‌کند اما این زبان به هیچ وجه نمی‌تواند او را راضی کند و از محیط گرافیکی AWT به بدی یاد می‌کند و وقتی می‌بیند سی تمام نیازهایشان را پاسخ می‌دهد ترجیح می‌دهند به ترند آن روزها توجه‌ای نکنند. می‌گوید: «جاوا از آن تکنولوژی‌هایی بود که از زمان خودش جلوتر بود و به درد نیاز ما نمی‌خورد».

در آپادانا با CGI یک بولتن برد طراحی می‌کنند و نامش را نمایه می‌گذارند. این سرویس یکی از برگ برنده‌های آپادانا می‌شود، یک سایت داخل شبکه که مردم در آن لینک به اشتراک می‌گذاشتند. دیگر محصولان یک سرویس چت است که سرورش را با جاوا می‌نویسند و کلاینت ویندوزش را با دلفی. این‌ها ایده‌هایی بود که از شبکه‌های BBS به ارث گرفتند. چند سال بعد ICQ و بعدش YAHOO pager یا همان مسنجر معروف وارد رقابت شد.

تا این که یک روز در گشت و گذارهایش با NetBSD آشنا می‌شود. می‌گوید: «خیلی پایدار تر بود و حداقل من راحت‌تر با آن کار می‌کردم» FreeBSD 3.3 آغاز ماجرای او با خانواده یونیکس است. با این که لینوکس و سولاریس را هم دنبال می‌کرده و حتی در توسعه سولاریس مشارکت کرده اما از آن سال به بعد تا امروز سیستم‌عامل مورد علاقه‌اش برای سرور FreeBSD می‌شود. به واسطه همین پروژه با آدم‌های مختلفی در کشورهای دیگر دوست می‌شود. می‌گوید: «الان هم هر کاری انجام می‌دهم یک ربطی به BSD دارد».

3 مهاجرت

مدتی بعد حدود سال ۸۰ شرکت آپادانا را ترک می‌کند و به داتک می‌رود و از همان ابتدا تمام سیستم‌ها را بر پایه یونیکس بنا می‌گذارد. روزی که آپادانا را ترک می‌کند سیستم اینترنت تازه از ۱۴ کیلوبیت به ۲۸ کیلوبیت رسیده است. آن روزها تازه موزیلا مرورگرش را عرضه کرده. می‌گوید: «آن موقع از همه بهتر اینترنت اکسپلورر بود، عالی بود، اکتیو ایکس و تمام فناوری‌ها رو هم ساپورت می‌کرد و موزیلا خیلی درب و داغان بود حتی من آن زمان اولین دستگاه اپل را که دیدم بهترین مرورگرش اینترنت اکسپلورر بود» هرچند زمانه به این نرم‌افزار وفا نکرد و اکنون جزو بدترین مرورگرهای دنیا است.

۴ سال در قامت مدیر فنی در داتک می‌ماند. می‌گوید: «یک پسر جوان کم سن و سال بودم که چون از این چیزها سر در می‌آوردم اسمم شد مدیر فنی و گرنه نه مدیریت بلد بودم نه چیزی». با این حال موفق می‌شود تمام ساختار سیستم‌های رایانه‌ای شرکت داتک را تحت BSD نصب کند. درباره ساختار BSD می‌گوید: «تر و تمیزتر است شما در لینوکس برنامه که نصب می‌کنید تنظیماتش را در /etc می‌ریزد اما بی‌اس‌دی می‌گوید تمام روت مال سیستم‌عامل است برای همین لینوکس هم مثل ویندوز بعد از یک مدت چاق می‌شود و همه چیز قاطبی می‌شود بعد از مدتی می‌روید در etc حجم زیادی configuration فایل می‌بینید می‌روید در bin تعداد زیادی فایل باینری می‌بینید اما در بی‌اس‌دی و سولاریس این طوری نیست همه می‌رود در /usr/local و هر وقت خواستی این پوشه را کلا پاک می‌کنی سیستم‌عامل نو می‌شود» او درباره مزیت‌های بی‌اس‌دی ادامه می‌دهد: «چیزی به نام port داریم و من نرم‌افزارهایم را از فایل‌های باینری نصب نمی‌کنم از سورس کد هم نصب نمی‌کنم که پیچیدگی‌های وابستگی داشته باشد یک چیزی بین پکیج و سورس است موقع کامپایل تصمیم می‌گیری چه آپشن‌هایی روشن باشد و به چه کتابخانه‌ای لینک شود لینوکس هم مزایای خودش را دارد اما با BSD خیلی خوشحال‌ترم» با این حال منکر برتری‌های لینوکس هم نمی‌شود و معتقد است وقتی بخواهد از پایگاه داده‌های اوراکل استفاده کند هیچ وقت سراغ BSD نمی‌رود.

در نزدیکی سال ۸۱ مخابرات سرویس E1 را شروع می‌کند و اوضاع دایال آپ مقداری بهتر می‌شود. در همان سال‌ها داتک با مشارکت شرکتی دیگر، ISP شده و بحث PAP و اینترنت ADSL به اتفاق روز جامعه فناوری اطلاعات ایران تبدیل می‌شود. بابک در نوشتن طرح ADSL هم مشارکت می‌کند اما همان روزها از داتک بیرون می‌آید.

3 ازدواج

سال ۸۴ به شاتل می‌رود هرچند دو سال بیشتر آن‌جا نمی‌ماند. به قول خودش شاتل خیلی زود برایش ملایم شد و دیگر آن‌جا کار کردن جذابیتی نداشت. این را می‌اندازد تقصیر این که شرکت بیش از آن که به مسائل فنی اهمیت دهد، دنبال بیزینس خودش بود. به محض ورودش روی تمام ساختار آن‌جا FreeBSD نصب می‌کند. می‌گوید: «این طوری شد که هر جا که رفتم یک ردپایی به جا گذاشتم» مجبور می‌شود هرآنچه می‌داند را به دیگران هم یاد بدهم تا

در صورت ترک شرکت مشکلی پیش نیاید. این یاد دادن‌ها باعث می‌شود تا سال ۸۹ شروع به آموزش بی‌اس‌دی با همکاری BSD Certification Group که مقرر اصلی‌اش در آمریکا بود بکند. نماینده رسمی این مؤسسه می‌شود. روند هم به این صورت است که امتحان می‌گرفتند و پاسخ‌ها را برای مؤسسه اصلی ارسال می‌کردند. با همین روش اکنون ۳۰ نفر در ایران مدرک رسمی BSD-A دارند و بعد از آمریکا ایران دومین کشور از نظر جمعیت متخصصان BSD است. با این حال مشغله کاری نمی‌گذارد این کار را ادامه دهد.

در همان سال‌هایی که در شاتل است با حنا حاج سید جوادی آشنا می‌شود و در سال ۸۵ ازدواج می‌کند. این ازدواج بعدها در زندگی کاری‌اش هم تأثیر مثبتی می‌گذارد. از نظر او بابک یک گیگ است. می‌گویم زندگی با یک گیگ چگونه است؟ سکوت سختی می‌کند. شور و شوق اولیه از صدایش رخت بر می‌بندد و آرام می‌گویی: «واپلش خیلی سخت بود» بعد از کمی سکوت ادامه می‌دهد: «من خیلی با مهندسین همکار بودم ولی بابک با بقیه متفاوت بود اون هیچ وقت ساعت کاری‌اش تمام نمی‌شود. روز تعطیل و مسافرت برایش تعریف ندارد» می‌خندد: «البته الان خیلی بهتر شده» او ادامه می‌دهد: «با گذر سال‌ها من درک کردم که باید این فضا را به او داد، تفاوت‌های بین ما خیلی زیاد است با وجود این به یک نقطه مشترک رسیدیم من نه با خودش با کارش هم ازدواج کردم» بابک می‌گوید: «من با این تضاد همیشه دست به گریبانم این که برای پیشرفت باید خودت را پرزنت کنی و در جمع باشی خیلی وقت‌ها مقاله می‌دهم وقتی موقع سخنرانی می‌شود پشیمان می‌شوم»

حنا حاج سید جوادی در شاتل مهندس نرم‌افزار است که بعد به کیش ویر می‌رود و مدیریت پایگاه داده‌های اوراکل آن‌جا را در دست می‌گیرد. چهار سال بعد وقتی شرکت ایمن پردیس را راه‌اندازی می‌کنند به خواست بابک مدیرعامل شرکت می‌شود تا همچنان فقط کارهای فنی در دست بابک بماند. می‌گوید: «نظرش این بود که من آدم اجرایی هستم» می‌خندد: «شاید هم می‌خواست همه چیز را گردن من بیندازد». بابک می‌گوید: «من حوصله کارهای اجرایی را نداشتم و خودم را در این اتاق حبس کردم تا جایی که امکان داشته باشد سعی می‌کنم پای کامپیوترم باشم و خیلی با آدم‌ها سر و کله زنم»

سال ۸۶ بابک فرخی تصمیم به نوشتن کتاب Network Administration with FreeBSD 7 می‌گیرد و



این کار یک سال طول می کشد. همسرش می گوید: «آن کار خیلی زمان بر و سخت بود یک سال از عمرش را گرفت ساعت هشت از شرکت می آمد و بعد از کمی استراحت تا چهار پنج صبح روی این کتاب کار می کرد و بعد دوباره هفت صبح می رفت. هدف هر دوی ما این بود که این کار انجام شود». وقتی می پرسیم چرا کتاب را به زبان انگلیسی نوشت می گوید: «این برای من یک چالش بود دنبال پول و معروفیت نبودم فقط دنبال این بودم که این کار دشوار را انجام دهم می دانستم خیلی کار سختی است». کتاب هنوز هم فروخته می شود و هر ماه انتشارات مبلغی را برای فرخی می فرستد. اصرار انتشارات برای نسخه دوم کتاب متناسب با فری بی اس دی ۱۰ مؤثر واقع نمی شود و بابک دیگر سراغ نوشتن کتاب نمی رود.

۳ سلطان BSD

تجربه زیادش در کار با BSD و تولید بسته های متعدد برای این سیستم عامل باعث می شود یکی از توسعه دهندگان اصلی به او پیشنهاد بدهد که رسماً توسعه دهنده بی اس دی شود و به CVS این پروژه دسترسی داشته باشد. او طی این مدت نزدیک به ۸۰ پکیج را نگهداری می کرد.

بعد از شاتل دوباره به داتک بر می گردد و چهار سال هم در آن جا می ماند. درباره این تغییر شغل های متعدد می گوید: «هر کجا کار روتین می شد دوست نداشتم می خواستم همیشه شیبی داشته باشد و شاتل خیلی سریع شیبش تمام شد» در داتک تصمیم می گیرد مشاور باشد و دورادور نظارت داشته باشد اما به اصرار شرکت دوباره معاون فنی می شود و چهار سال بعد متوجه می شود درگیر همان چیزی شده که نمی خواسته است. سال ۸۸ از داتک هم خارج می شود و برای دو سال با شرکت بیستاک کار می کند. می گوید: «بیستاک کار VoIP می کرد و این برای من یک فضای جدید بود» آن جا مدیر فنی می شود اما با این تفاوت که این بار برخلاف شاتل و داتک از ابتدای کار با شرکت نیست و ساختار فنی شرکت پیش از این درست شده است. می گوید: «وقتی من رسیدم قسمت فنی کار تمام شده و فقط نگهداری مانده بود» روی سرورهای آن جا فدورا نصب است اما به درخواست بابک فرخی و به خاطر رفتن مسئول سیستم ها، تمام آن ها پاک شده و فری بی اس دی می شود. دیدگاه مدیریتی و طبعاً محافظه کارانه اش در نگهداری سیستم به خوبی نمایان است. می گوید: «من با فدورا هیچ وقت نتوانستم ارتباط برقرار کنم چون خیلی شکننده بود من از دنیای BSD

و سولاریس به پایداری عادت داشتم و فدورا خیلی روی لبه فناوری بود تا چیزی را آپگرید می کردی سیستم خراب می شد من اگر بخواهم از لینوکس استفاده کنم حداقل دبیان استفاده می کنم»

آنجا هم خیلی زود شیبش تمام شد می گوید: «دیدم هر جا می روم شیبش تمام می شود گفتم چه کار کنم که تمام نشود» تصمیم می گیرد شرکت خودش را راه بیندازد. بنابراین شرکت ایمین پردیس را که دو سال قبل کارهای ثبتش را انجام داده بودند فعال می کنند. می گوید: «من اصلاً به مدل استارت آپی اعتقاد ندارم این که یک نفر با ایده بیايد پیش سرمایه گذار بعد او هم پول بدهد یک دفتر بخرد با میز پینگ پونگ و فوتبال دستی این جا جواب نمی دهد. ما مشتری داشتیم کار هم داشتیم برای این که جنبه قانونی داشته باشیم شخصیت حقوقی مان را ثبت کردیم یک جایی باید به جمعیتان اضافه می کردیم که مجبور شدیم دفتر بخریم همیشه اعتقاد داشتم هسته شرکت باید کوچک بماند اصلاً هم دنبال رشد قارچی و فضای استارت آپی و سر و صدا نیستیم»

کارش را با مشاوره دادن شروع می کند بعد کلاس های متعددی برگزار می کند. می گوید: «اصلاً فکر نمی کردم این قدر استقبال بشود من پنجشنبه جمعه ها هم می آمدم درس می دادم یهو از نیروی انتظامی زنگ می زدند که می خواهیم ۲۰ نفر را بفرستیم BSD یاد بگیرد» شهرستانی ها در روزهای تعطیل درخواست بوت کمپ می کنند و هر شرکتی که می خواهد به کارمندانش سولاریس یا BSD یاد دهد اول سراغ بابک فرخی می رود. تجربه چندین ساله کار در شرکت های مختلف برای او دستاورد بزرگی شده است. کار آن قدر سنگین و انرژی بر است که بعد از اجرای موفق چندین دوره آموزشی آن را متوقف می کند. می گوید: «آموزش را خیلی دوست دارم هر از چندگاهی سعی می کنم بچه های شرکت را جمع کنم به آن ها چیزهای جدیدی را که آشنا شده ام یاد بدهم».

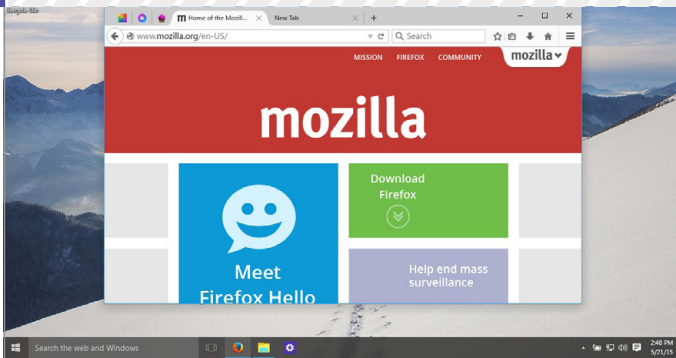
علت استفاده نکردنش از لینوکس دلایل متفاوتی دارد. گاهی کاملاً فنی است و گاهی اخلاقی. از نظر او لینوکس به نسبت BSD بیش از حد ناپایدار و آشفته است و برای سرور مناسب نیست. دلیل دیگرش را این طور می گوید: «من یکی از ناراحتی هایم این است که این آقای توروالدز خیلی بد دهن و بی تربیت است اصلاً خوشم نمی آید. در BSD یک سری آدم جنتمن با شخصیت اصلاً نه حاشیه دارند نه برای مطرح کردن خودشان به کسی فحش می دهند. مثلاً دکتر مارشال کک مک کیوستیک و آقای

جرج نویل و آقای دکتر رابرت واتسون کتابی به نام طراحی و پیاده سازی سیستم عامل فری بی اس دی دارند. کتاب هایشان را می خوانی لذت می بری من چون با این سه نفر در ارتباطم بسیار جنتمن و اهل کمک هستند تو سؤال احماقانه بپرس با حوصله جواب می دهد» ادامه می دهد: «سیستم بی اس دی را بیشتر می پسندم که قائل به شخص نیست و سیستم دموکراتیک است در لینوکس، توروالدز با آلن کاکس دعوايش می شود و بهبودی را که کاکس در هسته انجام داده را چون با هماهنگی او نبوده بر می گرداند و دوباره همان اشکال بد را می گذارد». از نظر او لایسنس بی اس دی بهتر از جی پی ال است. در این باره می گوید: «چون وقتی می گویی آزاد یعنی آزاد، جی پی ال آزادی مشروط دارد که از نظر من جالب نیست. بی اس دی می گوید اگر اسرار تجاری داری سورس کد را منتشر نکن ما زورت نمی کنیم برای همین سونی و خیلی های دیگر از آن استفاده می کنند با این حال اکثر شرکت هایی که از کدهای تحت لایسنس بی اس دی استفاده می کنند روی همان پروژه ها مشارکت می کنند و مشکلی با کانتربیتیوت کردن ندارند و حتی ریپوزیتوری فورک شده ندارند اما این لایسنس حالشان را خوب می کند»

مشخص است که تمام انتخاب هایش تخصصی و به دور از تعصب است. می گوید: «بیشتر از ده سال است که زندگی ام در مک است و ویندوز را به خاطر آفیس دارم فن ویندوز نیست اما معتقد هم نیستم ویندوز اشغال است لزوماً از نرم افزارهای متن باز و آزاد استفاده نمی کنم یکسری نرم افزار تجاری دارم که روی ویندوز و مک است و می توانم خودم را وفق دهم اما نمی آیم خودم را عذاب بدهم می روم ۲۰ دلار پول می دهم تا پست باکس را جای تاندربرد استفاده کنم. هیچ وقت روی میز کارم BSD نصب نمی کنم چون مک بهتر است و قطعاً هیچ وقت نسخه سرور مک یا ویندوز را روی سرورم نصب نمی کنم، قطعاً اوراکل را روی BSD نصب نمی کنم NetBSD روی ۳۰ پلتفرم مختلف اجرا می شود این ها هر کدام قطعات پازل هستند که باید در جای خودش بگذاری».

۴ از آبا دانا تا ایمین پردیس

بابک فرخی حالا مدیر فنی شرکت «ایمن پردیس» است. شرکتی که محصول «فرعی اش» یعنی «بیسفون» معروف ترین اپلیکیشن ارتباطی ایرانی است. خیلی ها این نرم افزار، رسانه اصلی شان شده و برای خود صفحه رسمی ساخته اند. تعدادی از



نامه سرگشاده مدیر عامل موزیلا به مدیر عامل مایکروسافت



داوود بهزادی

ساتیا: من دارم درباره یکی از جنبه‌های بسیار مخرب ویندوز ۱۰ به تو نامه‌نگاری می‌کنم. به‌طور صریح‌تر این که به نظر می‌رسد تجربه به‌روز رسانی به گونه‌ای طراحی شده که انتخابی که کاربران دربارۀ تجربه اینترنتی‌ای که می‌خواهند کرده‌اند را دور انداخته و با تجربه اینترنتی‌ای که مایکروسافت می‌خواهد داشته باشند جایگزین کند. هنگامی که نخستین بار تجربه به‌روز رسانی ویندوز ۱۰ را دیدیم که کاربران را با پایمال کردن موثر ترجیحات کاربری موجودشان برای مرورگر وب و دیگر کارهایشان، از انتخاب‌های خود بازمی‌دارد، با تیم شما تماس گرفتیم تا در مورد این مشکل بحث کنیم. متأسفانه این امر منجر به هیچ پیش‌رفت معنی‌داری نشد.

ممنونیم که هنوز به صورت فنی این امکان وجود دارد که تنظیمات و پیش‌گزیده‌های پیشین افراد نگاه داشته شود، ولی طراحی تجربه کلی به‌روز رسانی و تنظیمات و رابط‌های برنامه‌نویسی پیش‌گزیده به گونه‌ای تغییر یافته که این کار را مبهم‌تر و سخت‌تر کند. حالا افراد برای برگرداندن انتخاب‌هایی که پیش‌تر در نگارش‌های قدیمی‌تر ویندوز ایجاد کرده بودند نیاز به بیشتر از دو بار کلیک موشی، پیمایش در میان محتویات و مقداری مهارت فنی دارند. این کار گیج‌کننده و دشوار است و به آسانی قابل از بین رفتن.

فلسفه وجودی موزیلا، آوردن انتخاب، کنترل و فرصت برای همه است. فایرفاکس و دیگر محصولاتمان را برای این منظور ساخته‌ایم. موزیلا را به عنوان یک سازمان غیرانتفاعی برای این منظور ساخته‌ایم. و کار می‌کنیم تا تجربه اینترنتی، و رای محصولاتمان تا جایی که می‌توانیم این ارزش‌ها را ارائه دهند.

گاهی پیش‌رفت‌های بزرگی می‌بینیم، جایی که محصولات مصرفی به افراد و انتخاب‌هایشان احترام می‌گذارند. با این حال، با به بازار آمدن ویندوز ۱۰ عمیقاً ناامید شدیم که دیدیم مایکروسافت چنین گام چشم‌گیری رو به عقب برداشته است.

این تغییرات ما را به دلیل این که سازمانی هستیم که فایرفاکس را می‌سازد ناراحت نمی‌کند. این‌ها به این دلیل ناراحت‌کننده هستند که میلیون‌ها کاربر، ویندوز را دوست دارند و همان‌هایی هستند که انتخاب‌هایشان نادیده گرفته شده است، و به دلیل پیچیدگی افزوده شده‌ای که در راه هر کسی که هرگاه بخواهد انتخابی متفاوت از آن چه مایکروسافت می‌پسندد داشته باشد، گذاشته شده است.

قویا به شما توصیه می‌کنیم که تاکیک تجاری خود را مورد بازنگری قرار داده و با ساده‌تر، واضح‌تر و شهودی کردن حفظ انتخاب‌هایی که مردم تا کنون کرده‌اند در تجربه به‌روز رسانی، دوباره به حق آن‌ها برای انتخاب و کنترل آسان‌تر تجربه برخطشان احترام بگذارید. تنظیم انتخاب‌ها و ترجیحات جدید باید برای مردم ساده‌تر باشد، نه فقط برای محصولات دیگر مایکروسافت، بلکه از طریق رابط‌های برنامه‌نویسی تنظیمات پیش‌گزیده و رابط‌های کاربری.

لطفاً در ویندوز ۱۰، انتخاب و کنترلی که کاربران مستحق آن هستند را بهشان بدهید. خالصانه،

کریس بیرد | مدیر عامل موزیلا

کانال‌ها روزی ۴ هزار دنبال کننده جدید دارند. بیسفون ۴ سال پیش بر اساس یک ایده بلند پروازانه شکل گرفت، آن را با شرکت بیستاک در میان گذاشتند و شروع کننده یک جریان جدید شدند. می‌گوید: «مدیر بیستاک هم انسان جسوری بود و از این که به ما بخندند نمی‌ترسید با چراغ کاملاً خاموش کاری را شروع کردیم که در آن تخصصی نداشتیم چون نرم‌افزار بک‌اند و سروری خوب تولید می‌کنیم اما در حوزه موبایل اصلاً. رفتیم سراغ کسانی که بلد بودند اما دیدیم آن‌هایی که ادعا می‌کردند خیلی در این حوزه سوار کار نیستند برای همین خودمان آستین‌ها رو بالا زدیم.» او حتی برای این کار خودش برنامه نویسی موبایل یاد می‌گیرد و شروع به کدنویسی می‌کند. می‌گوید: «خود من از اسمارت فون فراری بودم اما با Objective-C روی اپل کد نوشتم الان هم تیم ۱۸ نفره تقریباً تمامشان از این جا شروع کردند». درباره جو شرکت ایمن پردیس می‌گوید: «اتمسفر ما خیلی اتمسفر خاصی است همه با آغوش باز می‌خواهند به هم چیز یاد بدهند».

حالا بدون این که از بیسفون درآمدی داشته باشند به خوبی در حال توسعه آن هستند. می‌گوید: «بیسفون به عنوان سرگرمی شروع شد و هنوز هم همین‌طور است و به شدت هم برایش هزینه می‌کنیم ولی پولمان را از جای دیگری در می‌آوریم البته بیسفون هم قرار است پول درآورد.» کار اصلی شرکت فروش یک سیستم CGN است که در سطح Carrier عملیات Nat انجام می‌دهد. اپراتورهای موبایل و شرکت‌های مهمی از این سرویس استفاده می‌کنند. کار این محصول این است که برای مهاجرت به IPv6 زمان می‌خرد. می‌گوید: «مهاجرت به نسخه ۶ زمان‌بر است برای این زمان جابجایی اپراتورها دچار کمبود آی‌پی هستند و باید نت کنند منتهی نت برای کاربر نهایی طراحی شده نه در مقیاس بزرگ. ما محصولی درست کردیم که در ابعاد بزرگ می‌تواند نت انجام دهد بدون این که سرویس کاربر را تحت تأثیر قرار دهد و این یعنی میلیون‌ها نشست و چند ده میلیون بسته در ثانیه بدون این که کیفیت پایین بیاید در دنیا این کار انگشت شمار است» در ۱۸ سال گذشته کار اصلی بابک فرخی شبکه و برنامه نویسی بوده در حوزه‌هایی هم مثل روتینگ شبکه آشنایی پیدا کرده است.

یکی از محصولات شرکت Net OS نام دارد که یک سیستم‌عامل مبتنی بر FreeBSD است و برای کارهای شبکه و روتینگ بهینه شده است. یک فلش کوچک را میان دو انگشتانش می‌گیرد و به ما نشان می‌دهد. می‌گوید: «اگر از اینترنت رایتل استفاده می‌کنید از این رد می‌شود» این سیستم‌عامل برای پردازنده‌های زئون اینتل بهینه شده تا از دستورالعمل‌های بیشتری پشتیبانی کند و مثلاً محاسبات و کنترل به جای نرم‌افزار درون پردازنده پردازش شود. به ترمینالی که روی سیستمش باز است اشاره می‌کند و می‌گوید: «۳ میلیون بسته در ثانیه را روی هر دستگاه فوروارد می‌کند».

مصاحبه تمام شد. عکاس جدیدی که آمده بود هم عکس‌هایش را گرفت و آماده رفتن شدیم. سیگاری دود کرد و تعارفی زد که مرا با موتورش تا جایی برد. عکاس خبرگزاری مشرق است و در مهر و فارس هم کار کرده. ترک موتورش که نشسته بودم گفت: «آدم گاهی می‌ماند ما با این همه ادعای حزب‌اللهی بودن هیچ کاری برای مملکت نکردیم و این‌ها بدون هیچ ادعایی چقدر برای کشور مفید بودند» از صداقتش خوشم آمد. گفتم: «نه فقط برای ایران برای کل جهان». گفت: «برای کل جهان».





GNU+Linux



موسسه

آرزوی ادامه استفاده از نرم افزار آشنا

تعویض سیستم عامل یک مهاجرت عظیم است و دوره‌ی گذار آن قابل اجتناب نیست. با این حال بسیاری از کاربران بالقوه در صورتی اقدام به مهاجرت می‌کنند که بتوانند به استفاده از نرم افزارهای همیشگی خود ادامه بدهند. احتمالاً دلیل این نگرش، این است که محصولات اصلی شرکت‌هایی چون ادوبی و مایکروسافت، روی سیستم‌عامل‌های ویندوز و اواس ۱۰ یکسان‌اند. بیشتر کاربران پس از مهاجرت فقط تفاوت‌های دو سیستم عامل را می‌آموزند. پس انتظار دارند اوضاع در گنو/لینوکس هم، با وجود تاریخچه و اهداف متفاوت مشابه باشد.

دقت نکردن به ویژگی‌ها

افراد انتظار دارند گنو/لینوکس در برابر ویندوز و اواس ۱۰ بدوی و ابتدایی باشد، در نتیجه زمانی که یک ویژگی در فهرستی که انتظارش را دارند قرار نگرفته باشد یا نام متفاوتی داشته باشد، به جای جست‌وجو، استدلال می‌کنند که اصلاً

بهانه‌های استفاده نکردن از گنو/لینوکس

در زمانی که گنو/لینوکس تازه معروف شده بود، مقالات مختلفی کمبودهای گنو/لینوکس را محکوم می‌کردند. به سختی ممکن بود ماهی بگذرد، بی آن که کسی توضیح بدهد گنو/لینوکس چه کم دارد، یا برای قابل استفاده بودن به چه ویژگی، برنامه یا سرویسی نیاز دارد. و مثل اغلب موارد، شکایت‌ها گمراه کننده بودند. مسلماً نرم افزارهای آزادی که روی گنو/لینوکس اجرا می‌شوند، مثل تمام نرم افزارهای دیگر دارای برخی کاستی‌ها و نواقص هستند. مثلاً کماکان شما نمی‌توانید فرم‌های پی‌دی‌اف را پر کنید، یا در برخی کشورها مالیات خود را با گنو/لینوکس محاسبه کنید. در موارد دیگر، مثل تشخیص نویسه نوری یا تشخیص گفتار نرم افزارهای آزاد وجود دارند، ولی مقابل رقبای انحصاری ابتدایی به نظر می‌رسند. البته هر ساله کمبودها و نقایص کم‌تر و کم‌تر می‌شوند و شکایت‌ها اغلب به خاطر ناآگاهی است تا هر چیز دیگر. اما چه نوع از ناآگاهی؟ این‌جا جایی است که بحث جذاب می‌شود.



نویسنده:
بروس بائیلد



درباره‌ی پیدا نشدن فسیل‌های حلقه‌ی مفقوده می‌اندازد. تکامل‌گرایان اغلب به فرم‌های انتقالی اشاره می‌کنند، در ادامه خلقت‌گرایان از آن‌ها درخواست فرم‌های انتقالی بین آن فرم‌های انتقالی را می‌کنند تا جایی که طرف مقابل از این بحث فرسایشی خسته شده و به آن پایان دهد. آنگاه خلقت‌گرایان اعلام پیروزی می‌کنند. به طریق مشابه، طرفداران نرم‌افزار آزاد می‌توانند شواهدی را علیه این بهانه‌ها ارائه کنند. در واکنش به این شواهد، بهانه‌سازان به جای عقب‌نشینی از موضعشان، همان ایراد را به گروه دیگری از نرم‌افزارها اعمال می‌کنند. تا زمانی که دیگر طرفداران نرم‌افزار آزاد بحث با چنین کسی را بی‌فایده ببینند. آن موقع است که بهانه‌سازان ادعای پیروزی می‌کنند و متقاعد می‌شوند که اجتناب آن‌ها از گنو/لینوکس کاملاً توجیه شده است.

البته این بهانه‌ها در مورد انتظارات معقول نیستند، حتی در مورد گنو/لینوکس و نرم‌افزار آزاد هم نیستند. جز در موارد معدودی که کاستی‌هایی وجود دارد، این بهانه‌ها فقط توجیه‌هایی برای درست نشان دادن خود هستند. شک دارم چیزی که بهانه‌سازان بیان می‌کنند، واقعاً آرزویی برای جلوگیری از آزار دیدنشان توسط آزادی نرم‌افزار و راحت گذاشتنشان برای انجام رایانشان به روش‌هایی که همیشه انجامش می‌داده‌اند باشد. ■

نصب نرم‌افزار در گوشی تلفنشان که همان تجربه‌ی موجود در گنو/لینوکس را برای نصب نرم‌افزار ارائه می‌دهد، ندارند.

۳ فرضیات مبتنی بر اطلاعات قدیمی
عمده دلایلی که از استفاده گنو/لینوکس توسط کاربران جلوگیری می‌کنند، مبتنی بر اطلاعاتی هستند که زمانی درست بوده‌اند، ولی سال‌ها است که غلط هستند. نمونه‌هایی از این دلایل عبارتند از: شما باید هر چیزی را از طریق خط فرمان اجرا کنید، نصب آن دشوار است و پشتیبانی‌ای در کار نیست. تمامی این دلایل بیش از یک دهه است که دیگر درست نیستند.

۳ کمبود ابزارهای انحصاری
برای سال‌ها تازه واردان می‌گفتند چیزی که گنو/لینوکس واقعاً نیاز دارد، ابزارهای انحصاری استاندارد است، که بیشتر اوقات منظورشان فوتوشاپ بود. باید توجه داشته باشیم که برای بسیار از افراد، نکته اصلی در نرم‌افزارهای آزاد استفاده نکردن از ابزارهای انحصاری است. البته مقصر، تولیدکنندگان نرم‌افزاری هستند. این تولیدکنندگان همچنان به دنبال یافتن فهمیدن روش پول درآوردن از طریق فروش نرم‌افزار هستند و پس از تلاش‌هایی مانند فریم‌میکر ادوبی در سال‌های ۲۰۰۰ و ۲۰۰۱ به این نتیجه رسیده‌اند که نمی‌توانند.

به هر حال ده سال پیش، فروشگاه‌های نرم‌افزاری آزاد به خاطر راحتی کاربران، ابزارهای انحصاری را هم می‌پذیرفتند. اما به خاطر نداشتن پشتیبانی، شروع به ایجاد نرم‌افزارهای جایگزین کردند. برای مثال امروزه دیگر کسی وجود ندارد که با وجود لیبره‌افیس، خود را برای استفاده از فریم‌میکر به دردسر بیندازد. در بسیاری از موارد اگر هنوز اندک شانس برای نرم‌افزارهای انحصاری روی گنو/لینوکس وجود داشته باشد نیز، احتمالاً در چند سال آینده توسط نرم‌افزارهای آزاد جایگزین گرفته خواهد شد.

۳ مغالطه‌های بی‌پایان
روشی که افراد در استفاده از این بهانه‌ها دارند مرا به یاد استدلال‌های خلقت‌گرایان

وجود ندارد.

این نگرش به حدی قوی است که بخش صفحات گسترده لیبره‌افیس، احساس کرد که باید DataPilot را به Pivot Tables تغییر نام بدهد تا با اکسل مایکروسافت هم‌گام شود. البته اغلب اوقات، کاربران چیزی را که در انتظارش هستند می‌یابند.

۳ عدم آگاهی از جایگزین
نرم‌افزارهای آزاد به خاطر این که توسط انگیزه‌های اقتصادی محدود نشده‌اند، معمولاً دارای چندین جایگزین در هر گروه هستند. در اغلب موارد کاربران جدید از تمام این جایگزین‌ها آشنا نیستند. برای مثال، یک کاربر جدید ممکن است گیمپ را به دلیل عدم پشتیبانی مستقیم از مدل رنگ سی‌ام‌و‌ای کی یا تجزیه رنگ، به عنوان جایگزینی برای فوتوشاپ نپذیرد. ولی شروع به جست‌وجو نمی‌کند تا دریابد کریتا هردوی این کارها را انجام می‌دهد. در عوض کاستی‌های گیمپ را به عنوان اشکال نرم‌افزارهای گرافیکی گنو/لینوکس توصیف می‌کند.

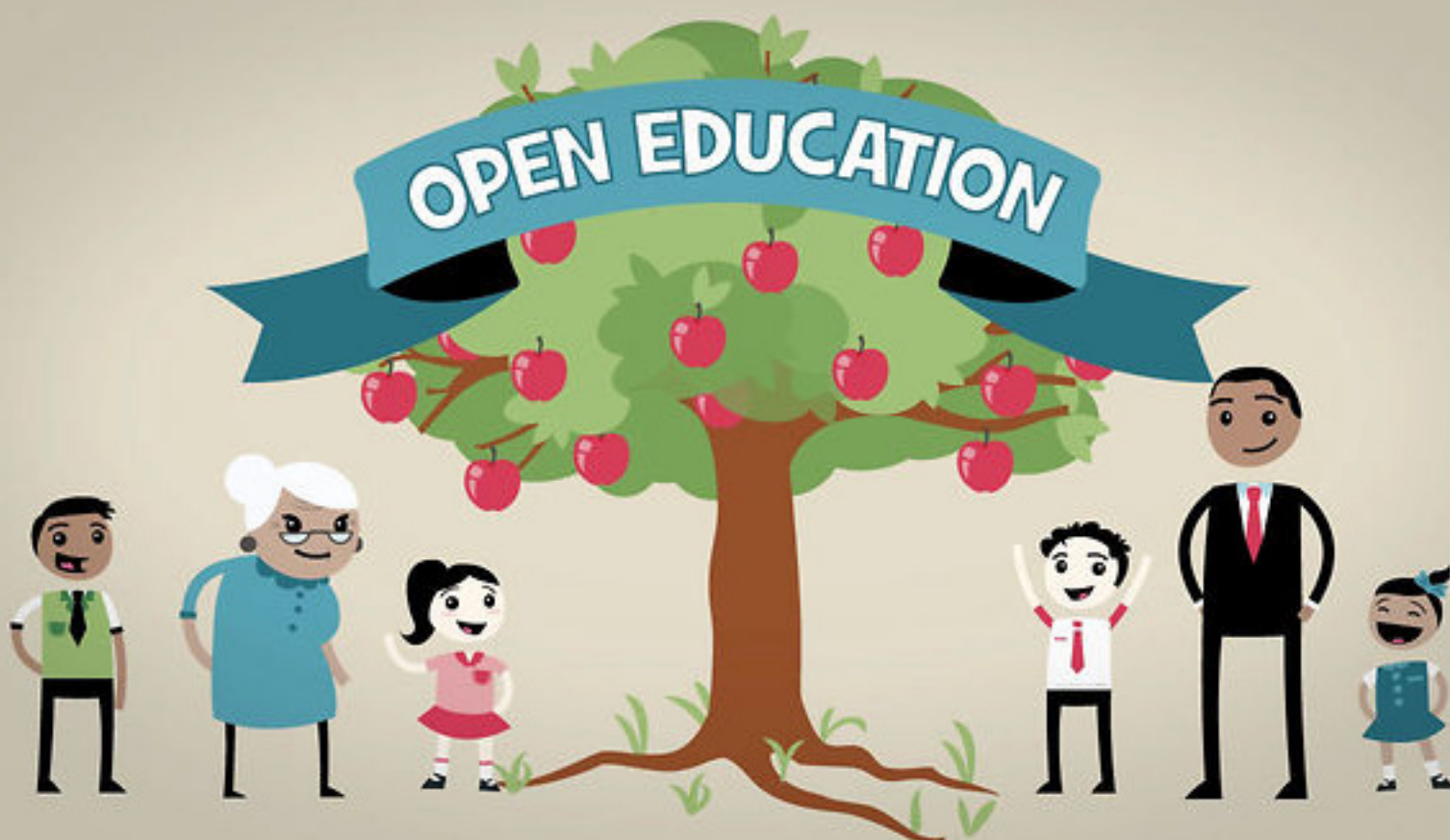
۳ موجود نبودن یک سرویس
گاهی اوقات کاربران تاکید می‌کنند که به پشتیبانی از یکی از خدمات برخط بر روی گنو/لینوکس نیاز دارند. خدمتی که اخیراً بیشتر به آن اشاره می‌شود گوگل درایو است. در واقع راه‌حلی برای داشتن گوگل درایو روی گنو/لینوکس وجود دارد، اگرچه برخی از آن‌ها خیلی موقتی به نظر می‌رسند. ولی حتی اگر این راه‌کارها هم وجود نداشته باشند، چرا اصرار بر گوگل درایو دارید؟ این تنها خدمات ابری موجود نیست.

۳ فرضیات مبتنی بر اطلاعات نادرست
رایج‌ترین مثال برای این بهانه، ادعایی است که می‌گوید گنو/لینوکس فاقد راهی آسان برای نصب نرم‌افزارهای جدید است. در عوض مدعی هستند که شما باید همه نرم‌افزارهایتان را از کد مبدأ کامپایل کنید. این بهانه وقتی جذاب می‌شود که همان افرادی که بهانه را ساخته‌اند مشکلی با



عمده دلایلی
که از استفاده
گنو/لینوکس
توسط کاربران
جلوگیری
می‌کنند، مبتنی بر
اطلاعاتی هستند
که زمانی درست
بوده‌اند، ولی
سال‌ها است که
غلط هستند





❖ کمی در باره خودتان بگویید چگونه به دنیای متن باز و آموزشی وارد شدید؟

یک معلم حرفه‌ای ریاضی هستم، با این حال سرگرمی دیگری را نیز همیشه دنبال می‌کنم؛ این سرگرمی‌ها برنامه‌نویسی و کامپیوتر هستند. اما برنامه‌نویسی و رایانه برایم همیشه یک فعالیت ذهنی فعال بوده‌اند. در طول سال‌ها، از چندین برنامه متن‌باز استفاده کرده‌ام، و یا به پروژه‌های کوچکی پیوسته‌ام، مانند ASCIIMathML. در حدود ۹ سال پیش، هنگامی که انجام تکالیف به‌صورت اینترنتی به جریانی اصلی مبدل شد؛ با خود گفتم که این موضوع واقعا ایده خوبی است، اما نمی‌توانم چنین هزینه‌ای را به دانشجویان تحمیل کنم. طبیعتاً دنبال این رفتم تا گزینه‌های متن‌باز و ارزانی را به جای موارد فعلی استفاده نمایم. در ابتدا با پروژه WebWork آشنا شدم. اما بعد متوجه شدم تنظیم آن بسیار پیچیده است. این‌زمانی بود که من IMathAS را به وجود آوردم؛ تا بسیار راحت‌تر قابل تنظیم باشد. این برنامه پروژه‌ای بود که به راحتی با LAMP کار می‌کرد به این خاطر که من علاقه داشتم از پروژه‌های متن‌باز استفاده کنم. پس‌از آن و با گذشت زمان، وقتی که این سیستم به خوبی پیاده‌سازی شد؛ به استفاده از آن اقدام کردیم.

آشنایی با استادی که باعث صرفه‌جویی زیادی برای دانشجویان شد

صرفه‌جویی میلیون دلاری و دانش آزاد

در ابتدا با دیوید لیمپن از طریق مقاله‌ای که در TeamOpen منتشر کرده بود، آشنا شدم؛ و بعد از آن بود که احساس کردم که باید با او در مورد کارش در آموزش‌های آزاد و متن‌باز صحبت کنم. دیوید یک استاد در کالج پیرس است؛ که باعث شده است دانشجویانش میلیون‌ها دلار به دلیل استفاده از کتاب‌های اشتراکی و دانش آزاد، صرفه‌جویی مالی داشته باشند. او همچنین برنامه‌ای متن‌باز و آزاد به نام IMathAS را توسعه داده است، که برای آموزش ریاضی و پلتفرم آموزشی، تهیه شده است.



نیکو زنگنه
نویسنده



سپس بعد از چند سال نظرم به این موضوع جلب شد و با خود گفتم، که چرا هر فرد باید ۱۵۰ دلار پول کتابی را بپردازد که بعداً ممکن است به آن نیازی نداشته باشد؟ این جا بود که بر روی مفاهیم کتب متنی باز و آموزش آزاد (باز) تمرکز کردم. پروژه کتابخانه دروس آزاد در واشنگتن به من و همکارانم زمانی را برای کار بر روی این ایده‌ها قرار داد تا به کار بر روی آنان بپردازیم. همچنین با همکاری با دیگر دانشگاه‌ها و مدارس، می‌توانستیم به مواد دست اول و عالی، مورد نیاز دسترسی داشته باشیم.

ما به بیش از ۳۰۰۰ دانشجو در هر ترم، خدمات می‌دهیم.

ابزار آموزشی متن باز مورد علاقه شما چیست؟

من واقعا نمی‌دانم چه چیزهایی واقعا به عنوان ابزار آموزشی باز (آزاد) هستند، اما واقعا به وبسایت UWN علاقه دارم. و همچنین کاری که NC Campus و Lumen برای توسعه Press- book انجام دادند.

ابزار متن باز مورد علاقه شما چیست (به جز ابزار خودتان)؟

از خیلی از آنان استفاده می‌کنم. Edit، ابزار اصلی برای کد نویسی است. همچنین از Gimp و Inkscape برای ویرایش تصاویر و طراحی استفاده می‌کنم. MathJax و jsxGraph هم کتابخانه‌های بسیار خوبی هستند، که در پروژه‌ی خودم از آنان استفاده می‌کنم.

درباره پیشرفت کارتان چه نظری دارید؟ و آیا به نظر شما، این پروژه می‌تواند به تعداد افراد بیشتری منفعت برساند؟

این خیلی باحال است، علاقه‌مند هستم تا افراد بیشتری آن را استفاده کنند اما ما هیچ وقت فکرش را نمی‌کردیم که تعداد افرادی که از این برنامه استفاده می‌کنند، تا این حد پیشرفت کنند. این واقعا جالب است؛ وقتی که می‌بینیم برخی افراد به ما پیام‌هایی را ارسال می‌کنند، که با ما نشان دهند، استفاده از این برنامه را آغاز کرده‌اند. در هر صورت با استفاده از گسترش دهنده‌های OpenStax، این امید وجود دارد که برنامه ما

MathAs، را چگونه ارزیابی می‌کنید؟

بعد از این که اولین نسخه را ایجاد کردم. آن را به فرد دیگری از جامعه دانشجویان کالج در ایالتان نشان دادم. او گفت که "فکر می‌کنی که پروژه قرار است چه گونه پیش رود؟" و به او گفتم که می‌خواهم این پروژه در کل ایالت قابل دسترسی باشد. سپس او رفت و با بودجه مالی برای انجام پروژه و ایجاد میزبانی اینترنتی برای قرار دادن آن در بستر وب، بازگشت. به همین خاطر اکثر دانشگاه‌ها در سراسر ایالت شروع به استفاده از این برنامه کردند. در ابتدا فقط دانشکده‌های بزرگ و آموزشکده‌ها به استفاده از آن پرداختند. مانند کسانی که نمی‌خواستند هزینه‌های زیادی را مانند چیزی که فعلا پرداخت می‌کردند، پرداخت کنند، اما بعداً این برنامه توسط عده‌ی بیشتری، حتی افراد حقیقی نیز مورد استفاده قرار گرفت.

بعد از آنکه تعداد زیادی دوره توسط کتب متنی انجام شد، بسیار مشتاق بودم این کتاب‌ها و ابزار را در خارج از واشنگتن نیز منتشر نمایم. بنابراین یک دامنه با آدرس myopenmath.com را اندازه‌ی کردم تا این گزینه در خارج از ایالت نیز قابل دسترسی باشد. تعداد کمی کالج وجود داشت که تصمیم گرفتند؛ نمونه‌های خود را میزبانی کنند، اما اکثر مردم استفاده از خدمات میزبانی شده را بسیار راحت‌تر یافتند. خوشبختانه ما توانستیم با یک شرکت تجاری همکاری کنیم که برای ما پشتیبانی فنی و نرم‌افزاری را انجام می‌دهد و همچنین هزینه‌ها را هم متقبل شد. در پاسخ به سوال شما، به طور کلی آن را خوب ارزیابی می‌کنم، تعدادی دانشکده هستند که در حال حاضر؛ از این برنامه بهره می‌برند. با این حال

به صورت گسترده‌تری مورد پذیرش قرار گیرد.

آیا تا به حال این موضوع را در نظر گرفته‌اید که از سیستم کنترل نسخه‌ای مانند gitHub برای پروژه خود استفاده کنید؟

ما تصمیم گرفتیم که کتاب‌های خود را در Word بنویسیم. به این خاطر که مخاطبان و اکثر جامعه‌ی دانشگاهی و آموزشکده‌ها بسیار بیشتر از TeX به Word علاقه داشتند. در حالی که ما می‌توانیم به راحتی از GitHub استفاده کنیم، حتی برای قرار دادن فایل Word. ولی از آن استفاده نخواهیم کرد؛ شاید افراد نتوانند از GitHub استفاده کنند؛ بنابراین ما آرشیوی را در وبسایتمان قرار داده‌ایم تا به راحتی به تمام نسخه‌ها دسترسی داشته باشند. در هر صورت اگر نسخه‌ای HTML از کتب متنی آماده شد آنان را در GitHub قرار خواهیم داد.

آیا پیشنهاد خاصی برای مربیان آموزشی دارید تا وارد جنبش آموزش باز (آزاد) شوند؟

برای آداب‌تورها، امکانات بسیاری بیشتری نسبت به آن چه در سابق بود، وجود دارد. و به راحتی می‌توان گام‌های زیادی را به جلو برداشت بدون آن که وقت زیادی را در این زمینه هدر داده باشید. توصیه می‌کنم که به عنوان مهمان ابتدا وارد سایت MyOpenMath.com شده و ببینید با کمی تلاش؛ چه چیزهایی به دست آمده است. همچنین برای دانشگاه‌ها در رشته‌های دیگر پیشنهاد می‌کنم تا از پروژه‌های کتب متنی باز، OpenStax و آموزش Lumen و... دیدن نمایند تا ببینند چه چیزهایی در این سایت‌ها قرار دارد.

اینجا بود که بر روی مفاهیم کتب متنی باز و آموزش آزاد (باز) تمرکز کردم. پروژه کتابخانه دروس آزاد در واشنگتن به من و همکارانم زمانی را برای کار بر روی این ایده‌ها قرار داد تا به کار بر روی آنان بپردازیم.





آیا نرم افزار امنیتی آزاد امنیت پایینی دارد؟

این سوالی تکراری است که ما معمولا در بنه تک «Benetech» در ارتباط با مارتوس می شنویم. ابزار قدرتمند و رایگان ما برای رمزنگاری، برای امن کردن جمع آوری و مدیریت اطلاعات حساس است که توسط برنامه حقوق بشر بنه تک^۱ ایجاد و آماده شده است. سوال مهمی است برای ما و همه کسانی که توسعه دهنده نرم افزارهای متن باز آزاد هستند؛ آن هم در عصر پسا-اسنودن که کاربران زیادی نگران درز اطلاعات شخصی یا شنود اطلاعاتشان توسط سرویس های امنیتی هستند. با این حال ما بر این باوریم که نه تنها نرم افزارهای آزاد امنیت بالایی دارند بلکه برای آن ضروری هستند.

۱. اجازه دهید تا قیاس زیر را تعریف کنیم:

رمزنگاری ترکیبی امنیتی است که باعث حفاظت از اطلاعات شما می شود و شما تنها کسی هستید که از آن ترکیب استفاده می کنید. یا این که این عمل را به یکی از همکارانی که به شما نزدیک است، می سپارید که هدف شما دور نگاه داشتن افراد غیرمجاز از دسترسی به داده های شماست. آن ها ممکن است سارقینی باشند که قصد دزدیدن اطلاعات مرتبط با کسب و کار شما را دارند یا این که کارمندان هم رده شما که قصد اطلاع از حقوقتان را دارند. یا موکلی که قصد دزدیدن اطلاعات مرتبط با کلاهبرداری را دارد؛ در تمام این موارد شما مجبورید اطلاعات خود را از آنان محفوظ داشته و از دسترسی افراد غیرمجاز جلوگیری کنید.

حال اجازه دهید بگویم مکانی امن برای اشیاء با ارزشم یافته ام. دو گزینه پیش رو دارم گزینه اول که از گاوصندوقی در تبلیغات مطلع می شوم که نیم اینچ عمق دیواره آن بوده و یک اینچ ضخامت درب آن است. همچنین برای نگهداری اطلاعات از ۶ قفل پیچیده برخوردار است. که توسط آژانس امنیتی مستقلی تایید شده است که تا دو ساعت در آتش دوام می آورد. گزینه دوم این است که از گاوصندوقی استفاده کنم که سازنده آن گفته است روش امن نگاه داشتن چیزهایی که درون است به صورت سری است که می تواند اطلاعات شما را با تخته سه لا و استیلی نازک امن نگاه دارد. پس این که به نظر شما کدام یک امن تر است؛ بسیار بدیهی است.

حال باید به دلیل آن که اطلاعات طراحی گاوصندوق اول در اختیار است و می توان به راحتی با تجهیزات و مواد اولیه لازم همانند آن طراحی کرد، گفت که شفافیت، باعث ناامن بودن گزینه اول خواهد بود؟ خیر، امنیت اطلاعات شما در گاو صندوق اول به دو دلیل است. یکی استفاده از ترکیبات و قدرت بالای آن در حفاظت اشیاء و ترکیبات امنیتی فوق العاده بالا است و دیگری به خاطر در دسترس بودن اطلاعات گاوصندوق برای کارشناسان که منجر به این می شود که آنان با کمک به طراحی و ارائه راهکارهایی جدید همیشه آن را بهتر از قبل نگاه می دارند. این گزینه باعث می شود که راهی مخفی در گاوصندوق برای گشودن آن وجود نداشته باشد. مثلا اگر مشکلی یا ایرادی در گاوصندوق وجود داشت، که با اطلاع از آن ساق می توانست آن را به سرعت باز کند، فردی خبره آن را متوجه شده و راهکار آن را ارائه خواهد کرد؛ زیرا که روش طراحی آن برای افراد در دسترس است.

۲. هدف ما از امنیت چیست؟

چیزی نیست که کامل مطلق باشد با این حال تبلیغات در حال اغراق هستند، به عنوان مثال هیچ روش امنیتی وجود ندارد تا اجازه دزدی از اشیاء سرقت شده را به سارقان ندهد و یا کاملاً آنان را متوقف کند، بلکه آن ها فقط قادرند هزینه دزدی از شما را برای سارقین بالا برند و سرقت از شما هزینه ای گزاف داشته باشد. این که بدانید آن روش امنیتی چگونه کار می کند آن را غیر امن نمی سازد. ممکن است علم به مواد اولیه و نوع فلز و یا فولاد به کار رفته در گاوصندوق باعث شود سارقین از ابزار مناسب به آن بهره جویند ولی چنین چیزی نیز باعث نمی شود که بگویم چنین روش هایی برای سارقین کم هزینه باشند. گاوصندوقی با طراحی و ترکیبات رمزی سخت می تواند هر مهاجمی را دلسرد کند.

قیاس ما از گاوصندوق و مکان امن، به راحتی می تواند موضوع نرم افزارهای امنیتی متن باز آزاد را برای شما روشن کند.

همانند امنیت گاوصندوق مذکور امنیت نرم افزار قوی رمزنگاری به دلیل آزاد بودن برای شما پرخطر نیست. در واقع کد منبع نرم افزار امنیتی با دیده شدن توسط دیگران تقویت شده و بر حسب نیاز امنیت و حریم

خصوص کاربران افزایش خواهد یافت. این همان دلیلی است که مرتباً اشاره می شود که نرم افزارهای آزاد با بررسی توسط کاربر نهایی، می تواند از راه های پنهان و درهای پشتی عاری باشد. در دنیایی که نظارت بر کاربران در حال تبدیل به یک هنجار است؛ بدیهی است کاربران خواهان شفافیت در عرضه محصولات امنیتی باشند و لزوم حفاظت از حریم خصوصی و بالا بردن امنیت بسیار حیاتی است. این موضوع برای فعالین حقوق بشر، روزنامه نگاران، فعالین گروه های مدنی و اجتماعی «NGOها» و دیگران بازیگران عرصه عدل و عدالت که امنیت و حریم خصوصی برای آنان از نزدیک ملموس است، ضروری است.

شاید موضوع متن باز بودن کد منبع برنامه های امنیتی منجر به بالا رفتن امنیت برنامه می شود را یک پارادوکس بدانید، با این حال با توجه به قیاس مذکور در ابتدای مقاله نرم افزاری که به صورت متن بسته عرضه می شود امنیت را بالاتر نخواهد برد؛ زیرا که امنیت به نحوه کار ترکیبات رمزی، الگوریتم های مورد استفاده و دیگر موارد موثر وابسته است. با این حال همانطور که در قیاس ذکر شد مهاجمان با مطالعه عمیق کدهای برنامه متن باز می توانند به مواردی دست یابند که باعث حمله به امنیت کاربران شود ولی امنیت به معنای عدم حمله و نبود ضعف نیست بلکه باعث پرهیز از شدن حمله و سرقت می شود.

ما در بنه تک بر این عقیده هستیم که استفاده از مدل متن باز برای توسعه به جامعه کاربران کمک خواهد کرد تا ما را در بهبود کدهایمان یاری دهند. در رابطه با موضوع مارتوس ما دیگر به اختراع مجدد چرخ نیازی نداریم و می توانستیم به راحتی از ابزارهای موجود مانند ابزار تور^۲ «Tor» برای ناشناس ماندن و دیگر ابزار متن باز و آزاد بهره جوییم.

موضوع نرم افزارهای امنیتی برای فعالین حقوق بشر به موضوعی مهم برای فعالین تبدیل شده است که به طور روزافزونی خواهان پشتیبانی از فعالین امنیتی در این بخشها هستند. برخی از آنان بمناسبت بنیاد فن آوری باز که کاربران را برای بررسی کدها تشویق می نماید. به طور جامع استفاده از نرم افزارهایی با شفافیت بالا، مسئولیت پذیری بیشتر و اثبات پذیری قویتر منجر به بهبود امنیت نرم افزار خواهد شد که تمامی این موارد در نرم افزارهای آزاد قابل دسترسی هستند. ■

□

۱. Benetech Human Rights Program

برنامه ای برای ناشناس ماندن در اینترنت است که از شبکه ای از کاربران در اینترنت شکل گرفته است





پرونده

- | ۵۲ | آشنایی با مدل‌های مدیریت داده‌ها
- | ۵۶ | با انواع پایگاه‌داده‌های پرکاربرد آشنا شوید
- | ۶۰ | ACID در مقابل BASE
- | ۶۲ | فهرست‌گذاری در پایگاه‌داده چگونه کار می‌کند؟
- | ۶۶ | موتورهای پایگاه‌داده در مای‌اس‌کی‌وال
- | ۶۹ | ارتباط پایگاه‌داده‌ها با دنیای خارج
- | ۷۴ | سرگذشت یک پروژه متن‌باز
- | ۷۵ | نکاتی که در فارسی باید رعایت کنیم
- | ۷۶ | سه محدودیت NoSQL



آشنایی با مدل‌های مدیریت داده‌ها

در دهه‌های گذشته، ذخیره‌سازی موثر داده‌ها و بدون تناقض آن‌ها جنبه مهم مدیریت داده‌ها بوده است. امروزه مساله ذخیره‌سازی داده‌ها سهم کمتری در میان مساله مدیریت داده‌ها دارد. مدیریت داده‌ها و استفاده از آن‌ها حتی نسبت به ده سال قبل، شکل دیگری به خود گرفته است. شبکه‌های اجتماعی، سیستم‌های امنیتی و هوشمند، میزبان‌های رسانه (Media Servers)، دوربین‌های مدار بسته، سیستم‌های مدیریت نقشه‌ها، شبکه‌های سنسوری و... هر یک به نوعی خاص نیازمند استفاده و مدیریت داده‌ها هستند. به عبارتی هر جنبه کیفی استفاده از داده‌ها مانند امنیت و سرعت دسترسی، امکان تغییر و توسعه داده‌ها به شکل‌های گوناگون و... باعث شدند مدل‌های ذخیره‌سازی متفاوتی برای داده‌ها به وجود آید. علاوه بر این موارد، حجم عظیم داده‌های مبادله شده باعث ایجاد مساله‌ای تازه به نام داده‌های بزرگ (BigData) شده است. بنابراین حوزه مطالعاتی در زمینه داده‌ها بسیار وسیع است. در این بخش سعی شده که به معرفی اجمالی از چند مدل ذخیره‌سازی داده‌ها و سیستم‌های مدیریت مربوط به آن مدل‌ها پرداخته شود. از آن‌جا که مدل داده‌ای یک پایگاه داده‌ها، رابطه مستقیمی با سیستم مدیریت آن پایگاه داده دارد (البته به جز نوع بدون قالب آن)، در این نوشته گاهی اوقات از واژه «مدل» و «پایگاه داده‌ها» به جای «سیستم مدیریت پایگاه داده‌ها» استفاده شده است. هر چند که هر کدام مفاهیمی جدا هستند.

خود را نمایش داد و یک شکاف بین منطق رابط‌های و منطق شیء‌گرا حس شد. برنامه‌نویسان گاهی مجبور بودند همان‌طور که اشیاء داده را در برنامه‌شان استفاده می‌کردند، آن‌ها را در پایگاه‌های داده ذخیره و استفاده‌ی مجدد کنند و به این ترتیب زمان و هزینه توسعه را کاهش دهند. در ادامه با مدل‌هایی که سعی کردند این شکاف را برطرف کنند آشنا می‌شوید.

سیستم‌های مدیریت پایگاه‌های داده شیئی گرا (OODBMS):

اواخر دهه ۸۰ میلادی را می‌توان زمان ظهور پایگاه‌های داده‌ی شیء‌گرا دانست که در واقع پاسخی به نیازمندی‌های برنامه‌های CAD بود که با اشیاء داده‌ی پیچیده و تودرتو سروکار دارند. مساله اینجاست که برنامه‌ی CAD یک برنامه‌ی روالی (procedural) است که در هر لحظه یک ورودی می‌گیرد و یک خروجی پیچیده تولید می‌کند، اما این با ساختار رابط‌های در تضاد است و پیاده‌سازی آن در منطق رابط‌های مستلزم اجرای دستورات join متعدد و پرس و جو (query)های طولانی، آن هم برای گرفتن تنها یک سطر خروجی است.

سیستم‌های مدیریت پایگاه‌های داده شیء‌گرا (OOD-BMS) برای حل این مشکلات به وجود آمد. خصوصیات این سیستم را می‌توان بطور خلاصه این‌طور بیان کرد:

- ❖ پشتیبانی از نوع داده کاربر (User Data Type)
- ❖ امکان ذخیره‌سازی اشیاء تودرتو
- ❖ پشتیبانی از لیست اشیاء (مانند مجموعه‌ها، آرایه‌ها، دسته‌ها) و bag (لیستی از لیست‌ها)، هر کدام به عنوان یک شیء مستقل
- ❖ پشتیبانی از متدهای اشیاء (معادل روال‌های ذخیره شده (Stored Procedure) در پایگاه‌های رابط‌های):

پایگاه‌های داده، مساله‌ای ضروری و مهم است. این موضوع در مدل ذخیره‌سازی رابط‌های، تا حدودی مساله‌ای چالشی است. گرچه معرفی سطوح نرمال‌سازی و روش‌های نرمال‌سازی پایگاه‌های داده، گام‌های مورد نیاز را برای این امر فراهم می‌آورد، اما طراحی درست و نرمال یک پایگاه داده‌های رابط‌های خبرگی و مهارت خاص خود را می‌طلبد.

هر دستور یا مجموعه دستورات وابسته به هم که از طریق کاربران به پایگاه داده ارسال می‌شود، تحت پوشش یک تراکنش اجرا می‌شود. مدیریت تراکنش‌ها در این نوع شیوه ذخیره‌سازی، مساله‌ای جدا از خود مدل ذخیره‌سازی است؛ به عبارتی سیستم‌های مدیریت پایگاه‌های داده رابط‌های (RDBMS) به طور معمول موظفند در زمان اجرای تراکنش‌ها، خواص را به نوعی رعایت و پیاده‌سازی کنند. بدین منظور هر یک از این سیستم‌ها شیوه و سیاست خاص خود را برای مدیریت تراکنش‌ها و پیاده‌سازی خواص ACID عرضه کرده‌اند. سیستم‌هایی مانند MySQL از شیوه قفل‌گذاری در سطح سطرها جدول به این هدف می‌رسند. اما در سیستمی مثل PostgreSQL، این امکان توسط چند نسخه‌سازی صورت می‌گیرد.

مدل رابط‌های دارای محدودیت‌های خاص خود است. کاربر نمی‌تواند به طور معمول و موثر، هر نوع ساختار داده دل‌خواه خود را در یک جدول پایگاه‌های رابط‌های ذخیره کند. به علاوه این که هر RDBMS برای خود مجموعه نوع داده (Data Type)های خود را ارائه کرده است که جدای از ناخوانایی با یکدیگر، محدود هستند

با معرفی منطق برنامه‌نویسی شیء‌گرا و زبان‌های مرتبط با آن و همه‌گیر شدن استفاده از این منطق در طراحی سیستم‌های نرم‌افزاری، این مشکل بیشتر

سیستم‌های مدیریت پایگاه‌های داده رابط‌های (RDBMS):

این مدل را می‌توان جزو پرکاربردترین و شناخته شده‌ترین مدل مدیریت داده‌ها و نیز بالغ‌ترین مدل بین سایر مدل‌ها دانست. شالوده مدل رابط‌های، منطق رابط‌های است. این منطق توسط ادگار کاد (Edgar Codd)، دانشمند علوم رایانه در سال ۱۹۷۰ معرفی شده و در واقع تفسیر دیگری از نظریه مجموعه‌ها و جبر رابط‌های است. هر قلم داده (Data Entity) در مدل رابط‌های، معادل یک سطر از جدول است که با یک کلید از سایر سطرها آن جدول متمایز می‌شود. منطق رابط‌های در واقع یک جبر بسته است. به این ترتیب که همه چیز، اعم از جداول و سطرها جدول، یک رابطه است و حاصل عملگرهای جبری بر روی رابطه‌ها، باز هم یک رابطه است. به خاطر این شالوده قدرتمند، مدل رابط‌های توانست جایگزین مدل‌های ناکارآمدی چون مدل سلسله مراتبی و مدل شبکه‌ای شود که پیمایش بین داده‌ها از طریق اشاره‌گرها و زیرروال‌های پیمایشگر صورت می‌گرفت؛ بر خلاف مدل رابط‌های که هر سطر به طور منظم و مستقل در جایگاه خود قرار دارد و یافتن آن‌ها نیازمند اشاره‌گرهای نسبی و فراخوانی زیرروال‌های پیمایشگر نیست. بسته بودن منطق رابط‌های، باعث به کارگیری پرس‌وجوها (query) به طور تو در تو و به طور موثر می‌شود. در این مدل، عملگرهای رابط‌های مانند join، باعث اتصال جدول‌ها با یکدیگر و استفاده‌ی موثر از داده‌ها می‌شود. زبان دستکاری و به کارگیری داده‌ها (DML) در این مدل، معمولاً زبان SQL است که یک زبان سطح بالا و اخباری (declarative) است. از آن‌جا که روند تدریجی توسعه سیستم‌های نرم‌افزاری می‌تواند باعث تغییرات آتی در ساختار پایگاه داده‌ها شود، طراحی درست یک



مهدی حمیدی
نویسنده

متدها جزئی از منطق شیء گرا هستند. به جای آن که برنامه درگیر اجرای query جهت دریافت اطلاعات اطلاعات از پایگاه داده‌های و محاسبات آن و ارسال نتیجه به پایگاه داده شود، یک متد شیء می‌تواند به طور موثر در سمت سرور پایگاه داده‌های اجرا شود و تغییرات را در همان جا، روی شیء اعمال کرده و نتیجه را به برنامه برگرداند.

❖ و یکی از مهم‌ترین خصوصیات این سیستم، اشتراک در نوع داده در برنامه و پایگاه داده‌ها است که باعث اعمال قوانین بررسی قوی نوع (Strong Type Checking) در زمان انتقال داده‌ها بین برنامه و پایگاه داده‌ها، به طور موثر می‌شود.

ادغام با زبان‌های برنامه‌نویسی: از آن‌جا که ساختار داده‌ها در پایگاه داده‌ها، مانند همان ساختار برنامه کاربر است، ذخیره و بازیابی اطلاعات می‌تواند از طریق دستورات DML درون خود برنامه یا زبان برنامه‌نویسی صورت بگیرد، به جای آن که دستورات DML از طریق یک واسطه محدود مانند Connection String یا دستورات خاص، به پایگاه داده‌ها، به عنوان یک سیستم جدا ارسال شود.

❸ مشکلات سیستم‌های مدیریت پایگاه‌های داده شیء گرا:

❖ پیمایش بین داده‌ها از طریق روال‌های جداگانه: در منطق پایگاه داده شیء گرا، اشیاء داده به وسیله‌ی اشاره‌گرها به یکدیگر متصل هستند. این منطق وجود یک زیرروال پیمایشگر را موجب می‌شود که می‌توان آن را یک گام به عقب توصیف کرد.

❖ نقض encapsulation: خاصیت encapsulation یکی از پایه‌های مفهوم شیء گرایی است. از آن‌جا که با اجرای query می‌توان به داده‌های درون یک شیء درون پایگاه داده‌ها دست یافت، می‌توان این طور گفت که پایگاه داده‌ها شیء گرا، مفهوم شیء گرایی را به طور کامل پشتیبانی نمی‌کنند. گاهی اوقات نقض این encapsulation ضروری است. به عنوان مثال نیازمند داده‌ای از یک شیء داده هستیم که متد آن پیاده‌سازی نشده است و دستیابی به این مقدار، تنها یک بار صورت می‌گیرد. در واقع منطقی نیست که برای دستیابی به این مقدار بخواهیم یک متد جدا درون شیء داده‌ی درون پایگاه داده‌ها بنویسیم. بنابراین همواره یک مصالحه بین نقض encapsulation و نوشتن متدهای بیشتر برای اشیاء داده وجود دارد.

❖ بسته بودن جبر رابط‌های موجود در منطق رابط‌های در این‌جا وجود ندارد و به کارگیری پرس‌وجو‌های تودرتو که حاصل این بسته بودن بود، در این‌جا از دست می‌رود. به خصوص که در این مدل، اخباری

بودن و سطح بالا بودن زبان دسترسی به داده‌ها (DML) تضعیف می‌شود.

❖ در نهایت آن استخوان‌بندی و شالوده قدرتمند ریاضی که منطق رابط‌های بر آن استوار بود، در منطق پایگاه داده‌های شیء گرا وجود ندارد و این به معنی از دست رفتن مجموعه‌ای از ابزارهای تحلیل و استنتاج مبتنی بر این منطق است.

ایده‌ای که در پشت پیاده‌سازی پایگاه داده‌های شیء گرا نهفته بود این بود که برای توسعه‌دهندگان بسیار مفید خواهد بود که نه تنها ذهنشان درگیر پیاده‌سازی پشت رابط اشیاء داده نشود، بلکه حتی درگیر این مساله نباشند که آن شیء چطور در پایگاه داده‌ها ذخیره می‌شود.

سه اشتباه که تکرار شد:

شرکت‌های توسعه‌دهنده و پشتیبان ایده پایگاه‌های داده شیء گرا متوجه تکرار اشتباه خود شدند. اولین اشتباه آن‌ها نزدیک کردن بیش از حد طراحی پایگاه داده‌ها به طراحی برنامه‌ها بود. آن‌ها دوباره فهمیدند که این نزدیک شدن با معماری چند لایه توسعه نرم‌افزارها که باعث انعطاف در توسعه نرم‌افزارها می‌شود تناقض دارد.

آن‌ها همچنین دوباره فهمیدند که استفاده از زبان‌های اخباری مانند SQL-۹۲ چنان بهرهوری را بالا می‌برد که شرکت‌های توسعه‌دهنده نرم‌افزار ترجیح می‌دهند که به جای درگیری با مفاهیم جدید، هزینه تامین منابع سخت‌افزاری را پرداخت کنند؛ به عبارتی شما همواره می‌توانید سخت‌افزار بخرید، اما زمان را هرگز! آن‌ها همچنین دوباره فهمیدند که استاندارد نبودن یک مدل داده‌ای منجر به خطا در طراحی و بروز ناسازگاری در داده‌ها می‌شود. نگهداری از یک سیستم مدیریت پایگاه‌های داده شیء گرا کاری بسیار سخت بود.

❸ سیستم مدیریت پایگاه‌های داده شیء-رابط‌های (ORDBMS):

گرچه تا سال ۹۵ میلادی، بحث شیء گرایی در پایگاه‌های داده مساله داغی بود، اما هنوز یک توافق عام بر سر این که یک پایگاه داده‌های شیء گرا باید شامل چه خصوصیات باشد، شکل نگرفت! در نهایت همگان بر این موضوع توافق کردند که پشتیبانی کمی شیء گرایی در پایگاه‌های داده بسیار مفید است! به عبارتی گرچه پایگاه‌های داده شیء گرا نتوانستند موفقیت قابل انتظاری را کسب کنند، اما پیشرو در ارائه چند ایده موفق و کاربردی بودند که بعدها در پایگاه داده‌های شیء-رابط‌های نیز توانستند پیاده‌سازی شوند. همچنین توسعه‌دهندگان سیستم‌های مدیریت پایگاه داده‌های رابط‌های نیز متوجه وجود کمبود در این

سیستم‌ها شدند، سعی کردند به هر طریقی پشتیبانی از اشیاء داده را در سیستم‌های خود و با رعایت انطباق با نسخه‌های قبلی وارد کنند. که در منبع چهارم این نوشته با دشواری‌های زیاد این رویکرد آشنا خواهید شد. نهایتاً گرایش به فواید موجود در مدل‌های ORD-BMS و OODBMS باعث به وجود آمدن مفهومی با عنوان پایگاه‌های داده شیء-رابط‌های (Object-Relational) شد. در این مفهوم به طور موثری تلاش شده که نه تنها نقاط قوت هر دو سیستم پیاده‌سازی شده و نقاط ضعف آن‌ها برچیده شود، بلکه این سیستم بیانگر یک راهکار نوین در زمینه مدیریت داده‌ها محسوب شود. سیستم مدیریت پایگاه داده‌های که در این زمینه پیشرو بوده است، PostgreSQL است. PostgreSQL را می‌توان یک سیستم پایگاه داده‌های شیء-رابط‌های بسیار قدرتمند و انعطاف‌پذیر دانست که علاوه بر معرفی راهکاری در جهت پشتیبانی از اشیاء داده‌ها، استانداردهای روز پایگاه داده‌های رابط‌های را پیاده‌سازی می‌کند. مشخصات کلی یک سیستم شیء-رابط‌های را می‌توان به این صورت بیان کرد:

❖ پشتیبانی از داده‌های پیچیده

❖ پشتیبانی از ارث‌بری نوع داده

❖ پشتیبانی از رفتار اشیاء

منظور از رفتار اشیاء همان متدهای تعریف شده درون اشیاء داده است و این رفتار را می‌توان به عنوان حتی یک برنامه‌ی مستقل گسترش داد. نکته جذاب این است که با این دیدگاه می‌توان بخش زیادی از لایه منطق (Business Logic) برنامه را در سمت سرور اجرا کرد، بدون آن که مانند گذشته، محاسبات در سمت کلاینت صورت بگیرد و تاخیر در انتقال داده‌ها باعث تاخیر در محاسبات گردد. یک سیستم مدیریت پایگاه داده‌های شیء-رابط‌های مانند یک سکو یا یک مثال بد، مانند یک سیستم عامل عمل می‌کند، به این ترتیب که می‌تواند منابع سیستم را برای این برنامه‌ها و نیز ارتباطات بین برنامه‌های را کنترل و مدیریت کند. این قدرت و انعطاف‌پذیری، باعث شده است که بسته‌های مختلفی برای این نوع سیستم نوشته شود که نوع داده‌ها و متدهای خود را وارد ORDBMS می‌کنند و کاربر می‌تواند در کنار استفاده‌ای که قبلاً از یک DBMS به عنوان یک سیستم رابط‌های استاندارد می‌کرد، از این متدها و نوع داده‌های جدید در برنامه خود استفاده نماید.

در این نوع پایگاه داده‌های، پشتیبانی از شیء گرایی به طور موثری از طریق جداول مسطح صورت می‌پذیرد و پشتیبانی از نوع داده‌هایی مانند کلکسیون‌ها، خلی در مفهوم نرمال بودن پایگاه داده‌های از دید رابط‌های آن وارد نمی‌کند.



❶ معرفی سیستم‌های مدیریت پایگاه‌داده‌های NoSQL و مخازن داده:

گرچه مدل رابطه‌ای بسیار قدرتمند و انعطاف‌پذیر است، اما همان‌طور که اشاره شد، کار کردن با این مدل داده مهارت خاص خود را می‌طلبد. جدای از این، خیلی از کاربران، مشکلات و نیازمندی‌هایی دارند که راهکار پایگاه‌داده‌های رابطه‌ای و حتی شیء-رابطه‌ای برایشان مناسب نیست. در دهه اخیر سیستم‌هایی موسوم به NoSQL و برنامه‌های مربوط به آن‌ها بسیار پرطرفدار و فراگیر شده‌اند، با این شعار که در کنار معرفی کارکردهای جدید، بسیاری از دشواری‌های کار کردن با مدل رابطه‌ای را نیز برطرف کنند. در واقع هدف این است که با ریشه‌کن کردن محدودیت‌های سخت‌گیرانه‌ای که قبلاً در سیستم رابطه‌ای معرفی شده بود، بتوان به راحتی و انعطاف بالا در نگهداری، پرس‌وجو و استفاده از داده‌ها رسید. پایگاه‌داده‌های NoSQL با استفاده از شیوه‌های ساخت‌نیافته (یا ساخت‌یافته در زمان عمل) به حذف محدودیت‌های رابطه‌ها کمک می‌کند و راه‌های متعدد و مختلفی را جهت موارد استفاده‌ی خاصی هم چون ذخیره «تمام متن» اسناد (full-text document storage) ارائه می‌کند. برخلاف آنچه که در مقدمه آورده شده، در سیستم‌های پایگاه‌داده‌های NoSQL هیچ مدل داده‌ای همانند سیستم‌های پایگاه‌داده‌های رابطه‌ای، مورد استفاده یا نیاز نیست. هر سیستم مدیریت پایگاه‌داده‌های NoSQL، به منظورهای خاص و هر یک به شیوه خاص، این منطق را پیاده‌سازی کرده‌اند. این پیاده‌سازی‌ها و راهکارهای بدون قالب (schema less) هم اجازه‌ی پشتیبانی از انواع نامتناهی از فرم‌های داده‌ای را فراهم می‌آورد و هم این‌که به طور ساده و بسیار موثری، از قالب «کلید-مقدار»ی که در مدل رابطه‌ای بود، پشتیبانی می‌کنند. پشتیبانی از قالب کلید-مقدار، بیشتر برای تعداد داده‌های کوچک و معمولاً به منظور cache کردن داده‌ها به کار می‌رود که در بخش مقایسه سیستم‌های NoSQL به آن پرداخته می‌شود.

بر خلاف پایگاه‌داده‌های رابطه‌ای، قادر خواهیم بود تا کلکسیون (collection) از داده‌ها را درون یک سیستم پایگاه‌داده‌های NoSQL همانند MongoDB داشته باشیم. این پایگاه‌داده‌های با به عبارت دیگر این «مخازن اسناد» (document stores) هر داده را در کنار سایر داده‌ها، تحت عنوان یک کلکسیون (یا همان سند) در پایگاه‌داده‌های نگهداری می‌کند. این اسناد می‌توانند تحت عنوان اشیاء داده مستقل مانند قالب JSON به نمایش درآیند و همچنین بر اساس صفت‌هایش

پرس‌وجو شوند.

سیستم‌های پایگاه‌داده‌های NoSQL بر خلاف سیستم رابطه‌ای که از زبان SQL استفاده می‌کرد، روش‌های مشترکی را برای پرس‌وجو از داده‌ها ارائه نمی‌کنند و هر سیستم NoSQL راهکار پرس‌وجو یا دسترسی خاص خود به داده‌ها را دارا است.

❷ مقایسه سیستم‌های مدیریت پایگاه‌داده‌های مبتنی بر SQL (رابطه‌ای) و NoSQL

جهت ساده‌سازی در ارائه‌ی مفهوم، سعی می‌گردد تا تفاوت‌های این دو سیستم بررسی گردد

❶ ساختار و نوع داده نگهداری شونده: در سیستم رابطه‌ای، نیازمند یک ساختار از صفات برای نگهداری از داده‌ها هستیم، بر خلاف NoSQL که معمولاً اجباری را تحمیل نمی‌کند.

❷ پرس‌وجو: همه پایگاه‌داده‌های رابطه‌ای استاندارد SQL را تا حدودی پیاده‌سازی کرده‌اند و می‌توانند از طریق SQL مورد پرس‌وجو قرار گیرند. در NoSQL شرایط عکس است. همان‌طور که گفته شد، هر پیاده‌سازی، روش خودش را برای کار با داده‌ها دارد.

❸ بسط‌پذیری: هر دو این راهکارها قابل رشد به صورت عمودی هستند. (مثلاً افزایش منابع سیستم). هرچند با پیشرفت و ساده‌تر شدن برنامه‌ها، راهکارهای NoSQL ابزارهای راحت‌تری را در جهت رشد افقی ارائه می‌کنند (مثلاً ساختن یک cluster از چندین ماشین) قابلیت اتکا: هنگامی که مسأله‌ی قابلیت اطمینان در داده‌ها و تضمین تراکنش‌های انجام شده مطرح باشد، همچنان بهتر است بر روی پایگاه‌های داده‌های رابطه‌ای شرط ببندید!

❹ پشتیبانی: پایگاه‌داده‌های رابطه‌ای، قدمتی طولانی دارند. همان‌طور که گفته شد، بسیار پرطرفدار هستند و پیدا کردن پشتیبانی برای آن‌ها چه به صورت رایگان و هزینه‌ای به راحتی امکان‌پذیر است. طبیعی است که حل مشکلات نیز در سیستم سنتی بسیار سریع‌تر از سیستم‌های تازه به ظهور رسیده مانند MongoDB که گفته می‌شود ذاتاً پیچیده است، خواهد بود.

❺ داده‌های پیچیده و نیاز به نگهداری و پرس‌وجو: به طور ذاتی، پایگاه‌داده‌های رابطه‌ای از منطق go to جهت پرس‌وجو‌های پیچیده و برآورده نمودن نیاز نگهداری از داده‌ها استفاده می‌کنند که در این حوزه پیشرو بوده و بسیار موثرترند.

❻ گرچه در ادامه به جزئیات بیشتری از سیستم‌های NoSQL پرداخته می‌شود، لیکن موارد استفاده با عدم استفاده از این سیستم‌ها را می‌توان به طور کلی به این شکل عنوان کرد:

❶ اندازه کار: اگر با دسته‌های عظیم از داده‌ها

سرور کار دارید، بسط دادن این داده‌ها توسط خانواده پایگاه‌داده‌های NoSQL راحت‌تر صورت می‌گیرد.

❷ سرعت: پایگاه‌داده‌های NoSQL معمولاً سریع‌ترند، و بعضی وقت‌ها در زمان عملیات نوشتن به شدت سریع‌ترند! عملیات خواندن هم بسته به نوع پایگاه داده NoSQL و داده‌های مورد پرس‌وجو می‌تواند بسیار سریع صورت بگیرد.

❸ طراحی بدون قالب: همان‌طور که گفته شد، پایگاه‌داده‌های NoSQL قابلیت انعطاف بالایی دارند.

❹ بسط‌پذیری و تکراری‌سازی آسان: بر خلاف پایگاه‌داده‌های رابطه‌ای، پایگاه‌داده‌های NoSQL به راحتی قابل بسط و تکرار هستند

❺ آزادی عمل در انتخاب: تنوع در پیاده‌سازی‌های مختلف از مفهوم NoSQL، این امکان را به کاربران می‌دهد تا با توجه به مسأله و داده‌ای که با آن کار می‌کنند، سیستم مدیریت پایگاه‌داده‌های مناسب را انتخاب کنند. در ادامه با این موارد قابل انتخاب آشنا می‌شوید.

❸ مقایسه سیستم‌های مدیریت پایگاه‌داده‌های NoSQL و مدل‌های آن‌ها

در این بخش سعی بر آن است که سیستم‌های مدیریت پایگاه‌داده‌های NoSQL معرفی گشته و کارکرد و اهداف آن‌ها تشریح شود تا بتوان در صورت نیاز، انتخابی درست و موثر انجام داد.

به طور کلی، می‌توان سیستم‌های مدیریت پایگاه‌های داده NoSQL را به چهار دسته زیر تقسیم نمود:

❶ مبتنی بر مقدار-کلید

❷ مبتنی بر ستون

❸ مبتنی بر سند

❹ مبتنی بر گراف

❶ مبتنی بر کلید-مقدار:

این مدل می‌تواند به عنوان ساده‌ترین مدل در بین سایر مدل‌های این دسته و نیز زیرساخت مفهوم NoSQL شناخته شود. این نوع از پایگاه‌داده‌های با تطابق کلیدها و مقادیر کار می‌کنند، چیزی شبیه dictionary. هیچ چیزی به عنوان ساختار یا رابطه وجود ندارد. پس از اتصال به پایگاه داده (مانند Redis)، یک برنامه می‌تواند یک کلید را عنوان کند (مانند the_answer_to_life) و یک مقدار متناسب (مثلاً ۴۲) را دریافت کند. سیستم‌های مدیریت پایگاه‌داده‌های مبتنی بر مقدار-کلید بطور معمول برای ذخیره‌ی سریع داده‌های اولیه به کار می‌روند. آن‌ها بسیار کارا، موثر و معمولاً بسط‌پذیرند.

نکته: وقتی صحبت از دیکشنری در دنیای کامپیوتر

می‌شود، منظور دسته‌ای خاص از اشیاء داده است که از آرایه‌ای از کلکسیون‌ها با کلیدهای مجزا تشکیل شده اند و هر کلید، با یک مقدار تطابق دارد.

برای سیستم‌های مدیریت پایگاه‌داده‌های NoSQL مبتنی بر کلید-مقدار می‌توان موارد زیر را معرفی نمود:

- ❖ Redis: درون حافظه اصلی قرار می‌گیرد و امکان ماندگار بودن در حافظه را نیز دارد.

- ❖ Riak: قابلیت بسط‌پذیری بالا، و قابلیت چند نسخه‌سازی

- ❖ Memcached / MemcacheDB: قابلیت بسط‌پذیری

❖ موارد مناسب جهت استفاده:

- ❖ caching: سرعت ذخیره بالا، و گاهی تکراری داده‌ها و با هدف استفاده در آینده نزدیک

- ❖ در صف قرار دادن: برخی از سیستم‌ها مانند Redis از لیست‌ها، دسته‌ها و صف‌ها و... پشتیبانی می‌کنند.

- ❖ توزیع عملیات و اطلاعات: می‌توانند جهت پیاده‌سازی الگوی معماری Publish-Subscribe موثر باشند

- ❖ نگهداری از اطلاعات زنده: برنامه‌هایی که نیازمند نگهداری از «حالت» (state) هستند، می‌توانند از این سیستم‌ها استفاده کنند.

❖ ۲) مبتنی بر ستون:

سیستم‌های مدیریت پایگاه‌های داده مبتنی بر ستون، بر اساس قدرتمندسازی همان طبیعت کلید-مقدار شکل گرفته‌اند. برخلاف آن تصور عامه در اینترنت که گفته می‌شود یادگیری این پایگاه‌داده‌ها مشکل است، این نوع پایگاه‌داده‌ها به طور خیلی ساده و بر اساس ساخت کلکسون‌هایی از یک یا چند جفت کلید-مقدار کار می‌کنند که با یک رکورد مطابقت دارد. برخلاف تعاریف قدیمی از قالب (schema) در سیستم‌های رابطه‌ای، یک سیستم مبتنی بر ستون نیازمند جدولی از پیش تعیین شده جهت کار کردن با داده‌ها در آن نیست. هر رکورد با یک یا چند ستون که دربردارنده اطلاعات هستند، ارائه می‌شود و هر ستون از هر رکورد می‌تواند متفاوت باشد.

به طور ساده، این نوع پایگاه‌داده‌ها، آرایه‌هایی دو بعدی هستند که به موجب آن، هر کلید (در اینجا: معادل سطر یا رکورد)، یک یا چند جفت کلید-مقدار را به همراه دارند که به آن‌ها متصل هستند. این سیستم، اجازه استفاده و نگهداری حجم عظیمی از داده‌های ساخت‌نیافته را می‌دهد. (مثلاً یک رکورد با مقادیر عظیمی از اطلاعات)

این سیستم معمولاً زمانی استفاده می‌شود که جفت‌های کلید-مقدار کافی نیستند و ذخیره تعداد بسیار زیاد از رکوردها با تعداد اطلاعات بسیار بزرگ،

یک ضرورت محسوب می‌شود. این سیستم قابلیت بسط‌پذیری بسیار خوب و بالایی را دارد.

برای سیستم‌های مدیریت پایگاه‌داده‌های مبتنی بر ستون می‌توان موارد زیر را معرفی نمود:

- ❖ Cassandra: مخزن داده مبتنی بر BigTable و Dy-namoDB

- ❖ HBase: مخزن داده سکوی Apache Hadoop، مبتنی بر ایده‌ی BigTable

❖ موارد مناسب جهت استفاده:

- ❖ نگهداری از داده‌های ساخت‌نیافته و اطلاعات غیرفراز: اگر دسته‌ای عظیم از مقادیر و صفات نیازمند به نگهداری برای مدت طولانی باشند، این سیستم بسیار کاربردی است.

- ❖ بسط‌دهی: این سیستم‌ها به طور ذاتی قابلیت بسط‌پذیری بسیار بالایی دارند و می‌توانند با مقادیر بسیار بسیار زیادی از اطلاعات سروکار داشته باشند.

❖ ۳) مبتنی بر سند:

سیستم مدیریت پایگاه‌داده‌های مبتنی بر سند را می‌توان آخرین دیوانه بازی دانست که توانسته سیل عظیمی از مردم را به استفاده از خود جوگیر کند! این سیستم همانند سیستم مبتنی بر ستون عمل می‌کند، گرچه امکان رسیدن به تودرتوسازی بیشتری را می‌دهد: وجود یک سند در یک سند دیگر، در سند دیگر..

سندها بر محدودیت یک یا دو سطح از تودرتوسازی سیستم‌های مبتنی بر ستون فائق آمده‌اند. به طور ساده، هر ساختار پیچیده و دلخواه می‌تواند تشکیل یک سند را بدهد و این سند قابلیت ذخیره‌سازی در این سیستم را دارد.

علیرغم ذات قدرتمند قابلیت پرس‌وجو از رکوردها با کلیدهای مختلف، این سیستم مشکلات و افت‌هایی را در مقایسه با بقیه همتایان خود دارد. به عنوان مثال، دریافت یک مقدار از یک رکورد به معنی دریافت مقدار زیاد آن رکورد است. مشابه این سناریو در زمان به‌روزرسانی داده‌ها نیز صادق است. این موارد باعث افت کارایی می‌شوند.

برای سیستم‌های مدیریت پایگاه‌های داده مبتنی بر سند می‌توان موارد زیر را معرفی نمود:

- ❖ MongoDB: پایگاه‌داده بسیار پرتعداد و بسیار کارا
- ❖ CouchDB: یک مخزن داده‌ی پیشرو و سنت شکن
- ❖ Couchbase: مبتنی بر JSON، قابلیت ذخیره در حافظه اصلی

❖ موارد مناسب جهت استفاده:

- ❖ اطلاعات تودرتو: این سیستم‌ها اجازه کار با داده‌های بسیار پیچیده و تودرتو را می‌دهد.

❖ کار با جاوا اسکریپت: یکی از حیاتی‌ترین کارکردهای این نوع سیستم‌ها، روشی است که در آن با برنامه‌های جاوا اسکریپتی روبرو می‌شوند، مثلاً استفاده از قالب JSON مناسب برای جاوا اسکریپت!

❖ ۴) مبتنی بر گراف:

در نهایت، می‌توان به سیستم جذاب مدیریت پایگاه‌داده‌های مبتنی بر گراف اشاره کرد. این سیستم‌ها داده‌ها را به شکلی کاملاً متفاوت از سه مدل قبلی که به آن اشاره شد، ارائه می‌کنند. آن‌ها ساختار درختی را همانند آن‌چه که در گراف‌ها هست ارائه می‌کنند، با راس‌ها و یال‌هایی که به وسیله‌ی رابطه‌ها به یکدیگر متصل هستند.

مشابه تصویری ریاضی گراف‌ها و به لطف طبیعت گراف‌ها (مانند اتصال و دسته‌بندی اطلاعات مرتبط با هم)، برخی از عملیات خاص در این سیستم‌ها، بسیار ساده‌تر صورت می‌پذیرند. (مانند ارتباط آدم‌ها)

این پایگاه‌داده‌های به طور معمول در برنامه‌هایی استفاده می‌شوند که برقراری مرزهای مشخص در اتصالات ضروری است. به عنوان مثال هنگامی که وارد یک شبکه اجتماعی شویم، اتصال دوستانان به شما و نیز اتصال دوستانان به شما از طریق سیستم‌های مدیریت پایگاه‌داده‌های مبتنی بر گراف ساده‌تر است.

❖ برای سیستم‌های مدیریت پایگاه‌داده‌های مبتنی بر گراف می‌توان موارد زیر را معرفی نمود:

- ❖ OrientDB: یک پایگاه‌داده‌های ترکیبی NoSQL از گراف و سند و بسیار سریع که با جاوا نوشته شده و قابلیت کار در مدل‌های کاری مختلف را دارد.

- ❖ Neo4J: پایگاه‌داده‌های بسیار پرتعداد و قدرتمند مبتنی بر جاوا

❖ موارد مناسب جهت استفاده:

- ❖ سروکار داشتن با داده‌های رابطه‌ای پیچیده: همان‌طور که گفته شد، این سیستم برای کار با ساختارهای پیچیده و رابطه‌ای بسیار موثر و راحت است. مانند ارتباط بین دو موجودیت و هر درجه از ارتباط بین موجودیت‌هایی که به طور غیر مستقیم با یک موجودیت رابطه دارند.

- ❖ مدل‌سازی و سروکار داشتن با موجودیت‌های دسته‌بندی شده: این سیستم‌ها در هر زمینه‌ای که مفهوم روابط انسانی مطرح است، برتری دارند. دسته‌بندی اطلاعات بر طبق روابط انسانی می‌توانند به طور خیلی خوبی توسط این سیستم‌های انبار داده صورت گیرند. ❖



mongoDB

با انواع پایگاه داده‌های پر کاربرد آشنا شوید

MongoDB

نوع: NoSQL



محمّد خالقی
نویسنده

مونگودی بی یک پایگاه داده مستقل از بستر است که بر خلاف پایگاه داده‌های معمول که بر پایه جدول و پایگاه داده‌های رابطه‌ای (relation-al database) پیاده‌سازی می‌شوند، مبتنی بر سند (document-oriented) پیاده‌سازی شده و با استفاده از یک قالب پویای شبیه JSON به یک پارچه‌سازی داده‌ها می‌پردازد. مونگودی بی این قالب را BSON می‌خواند که در انواع خاصی از کاربردها، سریع‌تر از پایگاه داده‌های رابطه‌ای معمول عمل می‌کند.

لازم به ذکر است که مونگودی بی جزو پایگاه داده‌های NoSQL دسته‌بندی می‌شود.

مونگودی بی نخستین بار در ۲۰۰۷ توسط شرکت MongoDB inc ساخته و توسعه داده شد. این شرکت از سال ۲۰۰۹ به مدل توسعه آزاد روی آورد و از طریق پشتیبانی خدمات خود برای شرکت‌ها، کسب درآمد می‌کند.

مونگودی بی با استفاده از زبان‌های برنامه‌نویسی سی، سی‌پلاس‌پلاس و جاوا اسکریپت نوشته شده است.

تاکنون مونگودی بی توسط مراکز و شرکت‌های بزرگی چون لینکدن، سورس‌فورج، ای‌بی، فوراسکوئر، نیویورک تایمز و به خدمت گرفته شده است. به علاوه مونگودی بی معروف‌ترین پایگاه داده‌ها NoSQL محسوب می‌شود.

تفاوت اصلی مونگودی بی با پایگاه داده‌های معمول در نوع نگهداری داده‌ها و رابطه میان آن‌هاست. مونگودی بی به جای این که از جدول‌ها برای نگهداری داده‌های خود استفاده کند، از چندین سند ساده برای این کار استفاده می‌کند. برای مثال به جای این که اطلاعات نویسنده و عنوان و قیمت را در جدول‌های متمایز مربوط به هم نگهداری کنیم، در مونگودی بی همه اطلاعات در یک

سند نگهداری می‌شود.

همچنین مونگو قابلیت بالایی در مجموعه‌های تکراری دارد، مجموعه تکراری مجموعه‌ای است که شامل بیش از دو نسخه از یک داده باشد. معمولاً از این قابلیت به این صورت استفاده می‌شود که یک عضو اصلی، تمامی اعمال مربوط به خواندن و نوشتن را انجام می‌دهد و عضوهای دیگر، از عضو اصلی رونوشت تهیه می‌کنند. در صورتی که عضو اصلی در انجام کار خود ناکام بماند، دیگر عضوها طی یک فرآیند رأی‌گیری، یکی از عضوها را به عنوان عضو اصلی انتخاب می‌کنند تا عملیات عضو اصلی را انجام دهد.

مونگو از سیستم sharding برای پخش کردن بار پردازشی سامانه استفاده می‌کند که موجب می‌شود هنگام بروز خطای سخت افزاری، سامانه با مشکل مواجه نشده و به کار خود ادامه دهد.

از دیگر ویژگی‌های مونگو می‌توان به مجموعه‌های پوش اشاره کرد. این مجموعه‌ها اندازه‌های ثابتی دارند و زمانی که مجموعه به اندازه مورد نظر رسید، مانند یک فهرست دایره‌ای عمل می‌کند.

مونگو هم‌چنین از کاهش نگاشت پشتیبانی می‌کند. مزایای استفاده از مونگو را می‌توان در موارد زیر خلاصه کرد:

- ❖ استفاده از ساختار سندمحور که باعث می‌شود اطلاعات مربوط به یک نمونه، در سند آن و با استفاده از قالبی شبیه JSON ذخیره شوند.
- ❖ ساختار هر شی مشخص است.
- ❖ مونگودی بی از ساختارهای پیچیده به دور است.
- ❖ استفاده از حافظه داخلی دستگاه برای ذخیره‌سازی کارها که موجب دسترسی سریع‌تر به داده‌ها می‌شود.

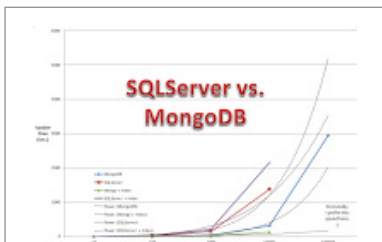
معایب مونگودی بی

نمایه‌گذاری فیلدها به دلیل استفاده از درخت دودویی، موجب اشغال فضای نسبتاً زیادی از حافظه سامانه می‌شود.

اندازه هر سند به دلیل تغییرپذیری احتمالی در آینده، ابتدا بسیار بالا در نظر گرفته شده و در مواردی که یک سند با اطلاعات کم هم داشته باشیم، حجم زیادی فضای اشغال می‌شود.

❖ مونگو از تراکنش‌ها پشتیبانی نمی‌کند.

❖ در برخی موارد، انعطاف‌پذیری پرس‌وجو در مونگو از رقبای مبتنی بر SQL آن کم‌تر است.



MySQL

نوع: سامپادرا (RDBMS)

مای‌اس کیوال در رده سامپادرا یا سامانه مدیریت پایگاه داده رابطه‌ای قرار می‌گیرد. مای‌اس کیوال دومین سامپادرای بزرگ جهان است که بزرگ‌ترین سامپادرای آزاد جهان هم محسوب می‌شود.

مای‌اس کیوال با پروانه جامع همگانی گنو منتشر شده است. این پایگاه داده در ابتدا توسط یک شرکت سوئدی به نام MySQL AB ساخته و توسعه داده شد و اکنون اوراکل مالک آن به حساب می‌آید. نخستین نگارش مای‌اس کیوال در ۱۹۹۵ انتشار یافت.

مای‌اس کیوال محبوب‌ترین پایگاه داده برای استفاده



SQLite

نوع: سامپادرا (توکار)

اس کیوالایت یک سامپادرا با حجم بسیار کم است. اس کیوالایت به صورت یک خدمت جدا از برنامه اصلی که روی میزبان خود اجرا شود نیست، بلکه در واقع یک کتابخانه در زبان سی است که به عنوان جزئی از برنامه اصلی اجرا می شود. به همین دلیل، برخلاف دیگر انواع پایگاه داده ها، اس کیوالایت به مراتب سبک تر از همتایانش به حساب می آید. اس کیوالایت تا حد خوبی، اکثر دستورهای SQL را پیاده سازی می کند، ولی تمامی ویژگی های SQL را ندارد. این پایگاه داده، تمامی اطلاعات مربوط را روی یک پرونده واحد نوشته و به همین دلیل، اجازه خواندن هم زمان از روی پایگاه داده ها را داریم، ولی اجازه نوشتن هم زمان داده ها روی پایگاه داده ها در دسترس نخواهد بود. با توجه به سبکی و این موضوع که فرآیندی جدید روی دستگاه اجرا نمی کند، اس کیوالایت به صورت گسترده در برنامه های سمت کاربر مورد استفاده قرار می گیرد.

برای مثال، مرورگرهای اینترنتی برای ذخیره تاریخچه بازدیدهای خود، از اس کیوالایت استفاده می کنند. همچنین خود سیستم عامل ها به صورت گسترده از اس کیوالایت استفاده می کنند. با توجه به استفاده بسیار گسترده اس کیوالایت در برنامه های سمت کاربر، به نظر می رسد اس کیوالایت، پرکاربردترین پایگاه داده موجود باشد.

اس کیوالایت توسط دکتر ریچارد هیپ که در سال ۲۰۰۰ برای نیروی دریایی ایالات متحده کار می کرد، به وجود آمده است. در آن زمان، دکتر هیپ روی برنامه مربوط به موشک های هدایت شونده که از روی ناوهای دریایی شلیک می شدند، کار می کرده و تا پیش از اس کیوالایت، از پایگاه داده شرکت IBM با نام اینفورمیکس استفاده می کردند. هدف از ساخت این پایگاه داده این بود که برنامه ها، بدون نصب پایگاه داده، به صورت مستقل اجرا شوند.

در باب مزایا و معایب این پایگاه داده، در بالا تا حد خوبی اشاره شد. با توجه به تفاوت مفهومی استفاده از اس کیوالایت، طبیعتاً این پایگاه داده، با پایگاه داده های معمول که در کارسازهای وب مورد استفاده قرار می گیرند، قابل قیاس نیست و به شدت ضعیف تر

مستقل از بستر بوده و روی همه سیستم عامل ها اجرا می شود. با این حال، بیشتر روی سیستم عامل های گنوالینوکس و ویندوز استفاده می شود. پستگرس، نخستین بار در دانشگاه برکلی و در ادامه پروژه اینگرس که یک سامپادرا بود در سال ۱۹۸۶ کلید خورد و اکنون توسط جامعه کاربری پستگرس که شامل خیل فراوانی از داوطلبان هستند، توسعه داده می شود. در سال ۱۹۹۴ اندرو یو و جولی چن، مفسر اس کیوال را به پستگرس اضافه کرده و آن را با نام پستگرس ۹۵ منتشر نمودند.

در مقایسه با سامپادراهای معمول، پستگرس از بُعد قابلیت، با نیاز عمیقش به شی گرائی، از مفهوم رابط های متمایز می شود.

بر اساس همین زیرساخت قوی است که پستگرس، توانایی خوبی در به دست گرفتن بهینه تعداد زیادی عملیات دارد.

از دیگر مزایای این پایگاه داده، می توان به این نکته اشاره کرد که پستگرس، به شدت توسعه پذیر بوده و با استفاده از روش های شخصی سازی، می توان از آن برای اجرای ساده عملیات پیچیده و تکراری استفاده کرد.

مزایای پستگرس کیوال

هم چون دیگر پایگاه داده های معروف، یکی از نقاط قوت پستگرس نیز جامعه کاربری بالای آن است و همین طور نرم افزارهای سوم شخصی که برایش منتشر شده و کاربری آن را آسان نموده اند.

توسعه پذیری

از ویژگی های مهم پستگرس می توان به توسعه پذیری بالای آن اشاره کرد. پستگرس تنها یک سامپادرا نیست، بلکه یک سامانه مدیریت پای شی گرا-رابطه ای محسوب می شود (سامپادراش).

معایب پستگرس

با استناد به ادعای پایگاه دیجیتال اوشن، در شرایطی که عملیات خواندن سنگین روی پستگرس اعمال شود، در رقابت با پایگاه داده هایی مانند مای اس کیوال، عمل کرد ضعیف تری از خود نشان می دهد.

زمان هایی که باید از پستگرس استفاده کنیم:

پستگرس ابزاری عالی برای مواقعی است که احتیاج به پایداری بالای اطلاعات داریم و تحت هیچ شرایطی نمی توان این موضوع را نادیده گرفت. در این مواقع، پستگرس نسبت به رقبای انتخاب بهتری است.

هم چنین در شرایطی که خدمت ما، طراحی بسیار پیچیده ای دارد، پستگرس نسبت به رقبای پیش است.

در وب است. برای مثال سامنه های مدیریت محتوای معروفی مانند جوملا، وردپرس، دروپال و... از آن استفاده می کنند. همچنین خدمات بزرگ اینترنتی نظیر گوگل، فیس بوک، توییتر، فلیکر، یوتیوب و... نیز از مای اس کیوال به عنوان پایگاه داده خدمات خود استفاده می کنند.

مای اس کیوال با استفاده از سی و سی پلاس پلاس پیاده سازی شده و تجزیه گر آن با یک نوشته شده است.

مای اس کیوال روی بسیاری از بن سازه های موجود مثل گنوالینوکس، خانواده بی اس دی، مک او اس ده، ویندوز و... کار می کند.

مزایای مای اس کیوال

فرآیند نصب و راه اندازی مای اس کیوال بسیار ساده است و نرم افزارهای سوم شخص بسیاری که برایش نوشته شده است، کار کردن با آن را چندین برابر آسان تر و سریع تر کرده است. همین طور استفاده از خود پایگاه داده بسیار ساده است.

در واقع اصلی ترین چالش کار با مای اس کیوال، یادگیری زبان SQL و موتورهای مختلف آن است.

مای اس کیوال یک استاندارد صنعتی و البته پایگاه داده بسیار محبوب است. این امر موجب گسترده بودن جامعه کاربری این سامانه شده است که پشتیبانی فنی قوی این محصول، ناشی از همین جامعه کاربری بزرگ است.

معایب مای اس کیوال

وجود برخی ناپایداری های جزئی در مای اس کیوال در برابر دیگر سامپادراها.



PostgreSQL

نوع: سامپادراش (ORDBMS)

پستگرس کیوال پایگاه داده هایی از نوع سامپادراش محسوب می شود که هدف اصلی آن تطبیق با استانداردها و توسعه پذیری بالا است، این پایگاه داده ها



cassandra

Cassandra

نوع: سامپاد توزیع شده

کاساندرایک سامانه مدیریت پایگاه داده توزیع شده است که برای پردازش حجم زیادی از داده روی تعداد زیادی کارساز طراحی شده است. کاساندرایا جابجا توسعه داده شده و دسترسی بسیار بالایی را بدون گلوگاه نرم‌افزاری ارائه می‌کند به این معنا که در کاساندرایا هیچ بخشی پیدا نمی‌شود که در صورت به وجود آمدن مشکلی در آن بخش، تمام سامانه از کار بیفتد.

آپاچی کاساندرایا در ابتدا به منظور بهبود جست‌وجو در صندوق‌های پستی کاربران در فیس‌بوک ساخته شد و در ۲۰۰۸ به عنوان یک پروژه آزاد روی گوگل کُد منتشر شد، و از ۲۰۰۹ تا کنون، توسط بنیاد آپاچی پشتیبانی می‌شود.

با توجه به کارایی بسیار بالای این پایگاه داده، بسیاری از شرکت‌های بزرگ فن‌آوری جزو کاربران آن هستند که در این میان، شرکت اپل با ۷۵۰۰۰ گره و ذخیره بیش از ۱۰ پتابایت داده، به عنوان بزرگ‌ترین شرکت استفاده‌کننده از این پایگاه داده، مطرح می‌شود. همچنین در فهرست دیگر استفاده‌کنندگان از کاساندرایا، نام‌های معتبری چون نت‌فلیکس، ای‌بی، گو ددی، گیت‌هاب، اینستاگرام، سرن و بیش از ۱۵۰۰ شرکت بزرگ دیگر، دیده می‌شوند.

کاساندرایا کارایی بسیار بالایی ارائه می‌دهد. برای مثال در سال ۲۰۱۲، دانشگاه تورنتو طی تحقیقاتی که روی سامانه‌های NoSQL انجام داد، در حوزه مقیاس‌پذیری، کاساندرایا را به عنوان برنده بلامنازع در میان دیگر پایگاه داده‌ها اعلام کرد. در طی آزمایش‌های آن‌ها، کاساندرایا بیش‌ترین توان عملیاتی را برای بیش‌ترین تعداد گره در تمام آزمایش‌ها ارائه داد، گرچه این موفقیت با هزینه‌ی زمانی بالای خواندن و نوشتن همراه بود.

کاساندرایا از مدل داده ردیف‌های بخش بندی شده استفاده کرده و در آن، ردیف‌ها در جدول‌ها دسته‌بندی می‌شوند. کلید نخستین موافقه‌ها هر جدول، کلید بخش آن است و در داخل هر بخش،

در باب معایب این پایگاه داده هم می‌توان به وجود برخی محدودیت‌ها در رأس‌های گراف اشاره کرد و این که نتوفورجی از سامانه sharding پشتیبانی نمی‌کند. به صورت خلاصه از سامانه sharding برای بخش کردن بار پردازشی سامانه در فرایندهای حجیم و مقیاس‌پذیری افقی سامانه استفاده می‌شود.



Redis

نوع: ذخیره کلید-مقدار

ردیس یک پایگاه داده درون حافظه‌ای است. این پایگاه داده‌ها از حافظه اصلی کارساز برای ذخیره داده‌ها استفاده می‌کنند که موجب سرعت بیشتر پایگاه داده و استفاده کم‌تر از توان پردازشی کارساز می‌شود، هرچند که طبیعتاً پایداری کم‌تری را نیز به همراه می‌آورد. با وجود فن‌آوری سخت‌افزاری non-volatile random access memory حتی در صورت قطع شدن برق نیز، از داده‌ها محافظت می‌شود.

ردیس از ساختار کلید-مقدار برای ذخیره داده‌ها بهره برده و کلیدها در آن با پایداری اختیاری ذخیره می‌شوند. به این معنی که تضمین می‌شود زمانی که یک تراکنش در پایگاه داده‌ها ثبت شد، تا مدت مشخص تعیین شده، این تراکنش در سامانه باقی مانده و از بین نمی‌رود.

ردیس نخستین بار در ۲۰۰۹ منتشر شد و توسط VMware و Pivotal Software پشتیبانی می‌شد، ولی از ۲۰۱۵ توسط آزمایشگاه‌های ردیس پشتیبانی و حمایت می‌شود.

از بُعد کارایی و سرعت، در شرایطی که احتیاجی به پایداری داده‌ها نباشد، ردیس فوق‌العاده سریع‌تر از پایگاه‌داده‌های معمول که هر تغییری را روی دیسک ذخیره می‌کنند، عمل می‌کند. ولی در صورت فعال بودن پایداری داده‌ها، به این دلیل که داده‌ها به صورت مقطعی، روی دیسک سخت پشتیبان‌گیری می‌شوند، کارایی سامانه مقداری پایین می‌آید.

هم‌چنین به این دلیل که ردیس به صورت تکررشته‌ای و تک‌فرایندی عمل می‌کند، یک اجرا از ردیس، توانایی انجام وظایف موازی را ندارد.

استفاده از حافظه اصلی در ردیس، با این که موجب سرعت بیشتر می‌شود، ولی با توجه به هزینه بالاتر حافظه اصلی نسبت به دیسک سخت، هزینه بیش‌تری روی دست کاربران خود می‌گذارد.

عمل می‌کند. در ضمن، با توجه به این موضوع که اس‌کیوالایت اجازه نوشتن هم‌زمان روی خود را نمی‌دهد، مطلقاً گزینه جالبی برای سمت کارساز به حساب نمی‌آید. ولی با توجه به قدرت بالا در عین سبکی، به شدت در برنامه‌های سمت کارخواه مورد استفاده قرار گرفته، که با توجه به استفاده کم از منابع میزبان در قیاس با دیگر پایگاه داده‌ها، کاربردهای فراوانی در این حوزه دارد.



Neo4j

نوع: پایگاه داده گرافی

نتوفورجی یک پایگاه داده گرافی است که توسط جابجا پیاده‌سازی شده است. نتوفورجی از زبان توسعه‌دهندگان خود آن چنین نامیده می‌شود: یک پایگاه داده توکار دیسک-محور و یک موتور مبادلاتی ماندگار جابجا، که داده‌ها را به جای این که به شیوه معمول در جدول‌های رابطه‌مند ذخیره کند، در گراف ذخیره می‌کند.

نتوفورجی با پروانه‌ارائه می‌شود: نگارش جامعه‌محور آن با GPLV3 ارائه شده و تعدادی از مولفه‌های اضافی آن با Affero GPL یا AGPL منتشر می‌شوند. هم‌چنین نگارش تجاری آن نیز موجود است. نگارش ۱۰ نتوفورجی در ۲۰۱۰ انتشار یافت و نگارش ۲۰ آن نیز در ۲۰۱۳ منتشر شد.

این پایگاه داده که به عنوان محبوب‌ترین پایگاه داده گرافی شناخته می‌شود، در شرکت Neo Technology inc واقع در سانفرانسیسکو ایالات متحده و مالموی سوئد ساخته و توسعه داده شده است.

در نتوفورجی همه داده‌ها به صورت یک یال، یک رأس یا یک مشخصه ذخیره می‌شود. هر رأس و یال می‌توانند به تعداد دل‌خواه مشخصه داشته باشند. هم‌یال‌ها و هم‌رأس‌ها می‌توانند برچسب بخورند. برچسب‌ها می‌توانند برای محدود کردن جست‌وجوها به کار روند.

از مزایای نتوفورجی می‌توان به این موارد اشاره کرد:

- ❖ یک ابزار خوب برای نشان دادن ارتباط میان داده‌ها است و با توجه به گراف محور بودن پایگاه داده، این کار بسیار ساده و قابل فهم انجام می‌شود.
- ❖ این پایگاه داده برای انجام پیمایش‌های مختلف روی داده‌های مرتبط به هم، بسیار ساده و سریع عمل می‌کند
- ❖ از مدل داده‌های بسیار ساده‌ای بهره می‌برد.

ردیف‌ها توسط ستون‌های کلید باقی‌مانده، خوشه بندی شده‌اند.



HBase 3

نوع: سامپاد توزیع شده غیر رابط‌های

اچ بیس هم از دیگر پروژه‌های آپاچی است که به عنوان بخشی از پروژه هدوپ توسعه داده شده و روی بستر سامانه پرونده توزیع شده هدوپ (HDFS) اجرا می‌شود. اچ بیس یک پایگاه داده توزیع شده غیر رابط‌های است که پس از پروژه جدول بزرگ گوگل، تحت زبان جاوا ساخته و توسعه داده شده است. این پایگاه داده در واقع قابلیت‌های جدول بزرگ گوگل را برای هدوپ به ارمغان آورده و یک روش تحمل پذیری خطا را برای داده‌هایی که توسط تعداد زیادی داده خالی و یا داده بی‌ارزش احاطه شده‌اند، ارائه می‌دهد.

اچ بیس نخستین بار در شرکت پاورست برای اهداف پردازش زبان طبیعی ساخته شد و اکنون توسط بنیاد آپاچی پشتیبانی می‌شود.

از ویژگی‌های اچ بیس می‌توان به فشرده‌سازی، انجام عملیات در حافظه کارساز و یا فیلتر بلوم به ازای هر ستون یاد کرد. این ویژگی‌ها براساس مقاله‌ای که در مورد جدول بزرگ از سمت گوگل انتشار یافت، پیاده سازی شده است.

جدول‌ها در اچ بیس به عنوان ورودی و خروجی برای کاهش نگاشت در هدوپ عمل می‌کنند.

اچ بیس به عنوان یک جایگزین سامانه‌های SQL بیان نمی‌شود، هرچند که اخیراً بهبودهای بسیار قابل توجهی روی کارایی آن انجام شده است و اکنون به عنوان پایگاه داده بسیاری از پایگاه‌های وب داده‌محور، از جمله خدمت پیام‌رسانی فیس‌بوک عمل می‌کند.



Microsoft SQL Server 3

نوع: سامپاد را

اس کیوال سرور مایکروسافت یک سامپاد را است که با سی‌سی‌پلاس‌پلاس نوشته شده است. این پایگاه داده، یک نرم‌افزار آزاد نبوده و متعلق به مایکروسافت است و برخلاف دیگر پایگاه داده‌ها، روی تمامی سیستم‌عامل‌ها اجرا نشده و فقط روی بن‌سازه‌های ویندوزی اجرا می‌شود.

در سال ۱۹۸۸، مایکروسافت به شرکت‌های استون تیپ و سی بیس پیوست تا نوع دیگری از پایگاه داده سی بیس اس کیوال سرور را برای IBM OS/2 بسازد. این پایگاه داده، نخستین نگرش اس کیوال سرور بود و موجب ورود مایکروسافت به بازار فروش پایگاه داده‌ها و آغاز رقابت با اوراکل و IBM و بعدها سی بیس شد.

اس کیوال سرور مایکروسافت در ۷ نگرش اصلی به این نام‌ها منتشر شده است: مرکز داده، شرکتی، استاندارد، وب، کسب‌وکار، کارگروهی و نگرش اکسپرس. همچنین نگرش‌های خاص دیگری نیز وجود دارد. برای مثال می‌توان به نگرش‌های توسعه‌دهنده، یا نگرش آرر که به عنوان یک سامانه ابری از سوی مایکروسافت استفاده می‌شود، اشاره کرد.

برای اس کیوال سرور، یک لایه رابط خارجی طراحی شده است که موجب می‌شود هر عملیاتی را که بخواهیم با آن انجام دهیم، با استفاده از قالب از پیش تعریف شده مایکروسافت به نام TDS استفاده کند. TDS یک لایه نرم‌افزاری است که برای تبادل اطلاعات بین پایگاه داده‌ها و کاربر به کار می‌رود.

Hive 3

نوع: پایگاه داده زیر ساخت

هایو یک پایگاه داده زیرساخت است که برای ارائه خلاصه‌ای از داده‌ها و تحلیل آن‌ها روی هدوپ اجرا می‌شود. هایو نخستین بار توسط فیس‌بوک ایجاد شد، ولی اکنون توسط دیگر شرکت‌ها مانند نت‌فلیکس و یا آمازون توسعه می‌یابد.

هایو از تحلیل تعداد بسیار زیادی داده که روی HDFS ذخیره شده‌اند پشتیبانی می‌کند. البته اکنون از سامانه پرونده Amazon S3 نیز پشتیبانی می‌کند.

هایو به صورت پیش‌گزیده فراداده‌ها را به صورت توکار در آپاچی دربی یا هر پایگاه داده مای‌اس کیوال مانند ذخیره می‌کند.

هایو برای سرعت بیش‌تر، تمامی نوع داده‌ها، حتی داده‌های نگاشت بیتی را شاخص گذاری می‌کند. همچنین هایو اطلاعات مربوط به خود را روی پرونده یا حتی روی دیگر پایگاه‌داده‌ها مانند اچ بیس ذخیره می‌کند.

هایو برای انجام عملیات روی داده‌ها، از الگوریتم‌های

BWT و اسنپی استفاده می‌کند.

زبان پرس‌وجوهای هایو بسیار شبیه SQL بوده و به سادگی قابل تبدیل به پرس‌وجوهای اسپارک و تز است.



solr 3

نوع: واسط برنامه نویسی جست‌وجو و نمایه گذاری

سولر یک بستر آزاد برای جست‌وجو در پایگاه داده‌ها است که با جاوا نوشته شده و منشأ اصلی آن، پروژه لوسین آپاچی بوده است. ویژگی‌های اصلی آن، جست‌وجو بین متن‌ها و نمایه گذاری آنی محتوا است و حتی بین پرونده‌های متنی موجود در پایگاه داده‌ها مثل pdf و پرونده‌های اداری نیز جست‌وجو می‌کند. سولر یک سامانه جست‌وجوی توزیع شده است و برای مقیاس پذیری و تحمل خطا طراحی شده است. لازم به ذکر است که سولر معروف‌ترین موتور جست‌وجوی تجاری نیز محسوب می‌شود.

Elasticsearch 3

نوع: جست‌وجو و شاخص گذاری

الستیک سرچ یک جست‌وجوگر کارساز است که با استفاده از جاوا و بر مبنای پروژه لوسین ساخته شده و یک موتور جست‌وجوی توزیع شده با ظرفیت چندگانه است. الستیک سرچ، پس از سولر، دومین موتور جست‌وجوی سمت کارساز تجاری معروف به حساب می‌آید.

الستیک سرچ نیز قابلیت جست‌وجو میان تمامی پرونده‌ها و اسناد اداری، pdf و دیگر اسناد متنی را دارا بوده و یک موتور جست‌وجوی توزیع شده است. به این معنا که تعدادی نمایه به shardهای مختلف تقسیم می‌شوند و هر shard می‌تواند صفر و یا بیشتر از خودش رونوشت بگیرد. در کارساز، هر گره، یک یا بیش از یک shard را میزبانی می‌کند و به عنوان هماهنگ کننده، هر عملیات را به shard مورد نظر هدایت می‌کند.

در میان کاربران الستیک سرچ نام‌های معتبری چون ویکی مدیا، گیت‌هاب، موزیلا، کیورا، نت‌فلیکس، سرن و... دیده می‌شود. ■



BASE در مقابل ACID

«تراکنش‌های حالت سیستم» که مجبور به داشتن CAD به عنوان خواص ذاتی یا «محدودیت‌های قابلیت اطمینان سیستم» بودند، مشخص کرده بود.

«بروس لیندی» و همکارانش مقاله‌ی «یادداشت‌هایی بر پایگاه داده‌های توزیعی» را در ۱۹۷۹ منتشر کردند که بر روی کار «گری» ایجاد شده بود. او اصول را برای دستیابی به قابلیت اطمینان و استانداردهای اولیه تکرار پایگاه داده ایجاد کرد. در سال ۱۹۸۳ «آندریاس روتر» و «تئو هارد» مقاله «مفاهیم بازبایی پایگاه‌داده‌های تراکنش‌گرا» را منتشر کرد و به طور رسمی اصطلاح ACID را به کار برد. این اصطلاح دو دهه بعد به صورت زیر معنی شد:

تمامیت: هیچ وظیفه (یا همه وظایف) درون یک تراکنش اجرا نمی‌شوند مگر این که همه آن‌ها انجام شوند. این اصل «همه یا هیچکس» است، اگر یکی از اجزای تراکنش با شکست روبرو شد همه تراکنش‌ها با شکست روبرو می‌شود.

قابلیت اطمینان: تراکنش باید تمام پروتکل‌ها یا قوانین تعریف شده توسط سیستم در تمام زمان‌ها را تأمین کند. تراکنش نباید آن پروتکل‌ها را نقض کند و پایگاه‌داده‌ها باید در یک حالت پایدار در آغاز و پایان تراکنش بمانند. هرگز هیچ گونه تراکنش نیمه انجام شده‌ای وجود ندارد. انزوا: هیچ تراکنشی به تراکنش دیگر که در یک حالت میانی و ناتمام است دسترسی ندارد. بنابراین هر تراکنش به خودی خود مستقل است. این موضوع برای کارایی و انزوای تراکنش‌ها در یک پایگاه‌داده‌ها نیاز است.

پایانی: زمانی که یک تراکنش کامل شد، در حالت انجام شده باقی می‌ماند و امکان بازگشت آن وجود ندارد. این تراکنش از شکست سیستم، قطع شدن برق و انواع دیگر از کارافتادگی سیستم مصون خواهد بود.

اتم‌های هیدروژن در محلول است. محلول‌های اسیدی فعالیت بیشتری دارند و محلول‌های قلیایی کمتر فعالند.

در پردازش کردن تراکنش پایگاه‌داده‌ها، داده‌ها اتم هیدروژن هستند که در سیستم جریان دارند و حرف «p» ویژگی‌هایی است که پردازش شدن این تراکنش‌ها را با داشتن قابلیت اطمینان تضمین می‌کند.

ACID چیست؟

اگر از هر کسی که در حوزه داده‌ها حرفه‌ای است این سوال را کنید، احتمالاً به خوبی می‌تواند شرح دهد. مفهوم ACID برای سالیان وجود داشته است و تا همین اواخر نیز یک معیار اصلی بود که همه پایگاه‌های داده در تلاش برای کسب کردن آن بودند. اگر پایگاه‌داده‌ای ACID را نمی‌داشت، قابلیت اطمینان آن زیر سوال می‌رفت.

جیم گری اولین بار در سال ۱۹۷۰ این ایده را به تصویر کشید و پس از آن مقاله بدیع «مفهوم تراکنش: خواص و محدودیت‌ها» را در ژوئن ۱۹۸۱ منتشر کرد. در این مقاله تنها اصطلاحات تمامیت (یا همه یا هیچ)، یکپارچگی و پایانی مورد بحث قرار گرفته بودند و پس از آن بود که اصطلاح انزوا نیز اضافه شد.

در آن مقاله بر روی سطح تراکنش تمرکز شده بود و تراکنش را به عنوان قرارداد یا هر تعداد از

در علم شیمی pH یک کمیت برای مشخص کردن نسبت اسیدی یا بازی بودن یک محلول

در آب است. تغییرات pH در بازه ۰ تا ۱۴ قرار دارد و محلول بشدت اسیدی مانند اسید باطری عدد ۰ و محلول بشدت بازی مثل NaOH عدد ۱۴ را کسب می‌کند. این در حالی است که آب در دمای ۲۵ درجه سلسیوس دارای pH ۷ است و این حالت را خنثی می‌نامند.

مهندسی داده با هوشمندی این استعاره را از شیمی‌دان‌ها وام گرفتند و آن را به یک اختصار تبدیل کرده‌اند. البته شاید معنی واقعی خود را نداشته باشد، اما زمانی که بحث درباره قابلیت اطمینان پردازش کردن تراکنش است همچنان نشان‌دهنده رویدادهای مربوط به یک سیستم پایگاه داده است. برای گسترش دادن بیشتر این استعاره مفید اجازه بدهید اول نگاهی به ایده pH بکنیم.

اولین بار «سورنسن» در حالی که در آزمایشگاه کارلسبرگ در ۱۹۹۰ در حال کار بود، این مفهوم را توسعه داد. بنابراین ایده مربوط به زمان حاضر نیست. در این جا حرف «H» به اتم‌های هیدروژن اشاره دارد و در مورد حرف «p» بحث و اختلاف فراوانی وجود دارد. برخی آن را از واژه فرانسوی «puissance»، برخی آن را از واژه آلمانی «potenz» و برخی دیگر آن را از واژه لاتین «pondus» یا «potentia» می‌دانند. اما هر چه باشد همه این‌ها به معنی پتانسیل یا توان (power) است. بنابراین pH به معنای قدرت، پتانسیل یا فعالیت

جنبه‌های بسیار دیگری برای این تعاریف وجود دارد و تامین نیازمندی‌های واقعی ACID مختص به هر پایگاه داده‌ها خاص است. اما در کل در دنیای RDMS همه چیز ACID است و بدون ACID قابلیت پایداری زیر سوال می‌رود.

عدد pH مربوط به ACID پایین است، تقریباً چیزی شبیه به اسید باطری (۰) و یا شاید سرکه (۲)، در این حالت داده و محدودیت‌های آن بسیار فعال هستند. بنابراین در هر میکروثانیه از یک پایگاه داده‌ها که از ACID به عنوان سیستم محدودیت استفاده می‌کند، همه داده‌ها (اتم‌های هیدروژن) تحت بررسی ثابت هستند تا از این برآورده شدن محدودیت‌ها اطمینان حاصل کنند. چنین محدودیت‌هایی برای سالیان در دنیای رابطه‌ای، نرمال، طرح واره محور، مقیاس پذیر افقی و کوچک پیش از شبکه‌های اجتماعی به خوبی کار می‌کرد.

این بدیهیات گذشته دیگر مطرح نیستند و اکنون داده بدون ساختار، بزرگ داده، ساختارهای داده غیر رابطه‌ای، سیستم‌های محاسباتی توزیع شده و قابلیت اطمینان نهان اکنون متداول شده‌اند. این نیازمندی‌های جدید به کلمات اختصاری جدید و pH جدید نیاز دارد.

❸ قضیه CAP:

در سال ۲۰۰۰ «اریک بروور» (Eric Brewer) سخنرانی خود را در همایش ACM که در مورد مفاهیم محاسبات توزیع شده بود ارائه داد و آن جا بود که قضیه CAP متداول شد.

اکثر مردم در جهان خارج هیچ‌گاه حتی اسم این قضیه را نشنیده‌اند و اهمیتی هم برایشان ندارد. آن‌ها فقط می‌خواهند رایانه‌شان کار کند، اینترنت کار کند، شبکه‌های اجتماعی کار کنند و البته همه این‌ها با در دسترس بودن قابلیت اطمینان فایل هایشان.

قضیه CAP، که با نام قضیه بروور هم شناخته می‌شود، بعدها تجدید نظر شد و از طریق کار «شیت گیلبرت» و «نانسی لیچ» از MIT در سال ۲۰۰۲ تغییر کرد.

اصل اساسی قضیه می‌گوید که سه نیازمندی اساسی در سیستم وجود دارد که این اصول برای یک طراحی، پیاده‌سازی و استقرار موفق برنامه‌ها در سیستم‌های محاسباتی توزیع شده ضروری هستند. این نیازمندی‌ها قابلیت اطمینان، دسترس پذیری و تحمل قسمت‌بندی است، که به اختصار CAP نامیده می‌شود.

قابلیت اطمینان به این مساله اشاره می‌کند که آیا یک سیستم کاملاً عمل می‌کند یا نه؟ آیا قابلیت اعتماد سیستم از قوانین مشخص شده در برنامه‌نویسی‌اش با توجه به این مقررات تعریف شده پیروی می‌کند؟ آیا همه گره‌ها در یک خوشه، همه

داده‌های مفروض برایشان را می‌بینند؟ این همان ایده معرفی شده در ACID است.

دسترس پذیری به معنای همان چیزی است که انتظارش می‌رود. آیا سیستم یا سرویس داده شده زمانی که درخواستی به آن ارسال می‌شود، حاضر است؟ آیا هر درخواست پاسخی خارج از شکست یا موفقیت را دریافت می‌کند؟

تحمل قسمت‌بندی، نشان دهنده این حقیقت است که یک سیستم داده شده، در شرایط از دست رفتن اطلاعات یا خرابی سیستم به عمل کردن ادامه می‌دهد. شکست یک گره نباید باعث سقوط کل سیستم شود.

این‌ها تنها تعاریف ساده‌ای از سه جنبه قضیه CAP بودند. مقالات فراوانی در این باره وجود دارد که بسیاری از تفاسیر، تحلیل‌ها و مشکلات پیچیده موجود در کاربردهای دنیای واقعی این قضیه را بحث کرده‌اند.

دلیل اصلی معرفی قضیه این نکته است که در اکثر نمونه‌ها و سیستم‌های توزیع شده تنها می‌توانند دو ویژگی از این ویژگی‌ها را تضمین کنند. در این جا چشم‌پوشی می‌تواند نتایج فاجعه باری داشته باشد که شامل احتمال شکست خوردن تمام سه عامل به صورت همزمان و جداگانه است.

محدودیت‌های قضیه CAP با قابلیت اطمینان پایگاه داده‌ها برای سیستم‌های بزرگ مقیاس، توزیع شده، غیررابطه‌ای همچنان پابرجا است. بیشتر اوقات آن‌ها به دسترس پذیری و تحمل قسمت‌بندی نیاز دارند. بنابراین قابلیت اطمینان فدا می‌شود و ACID سقوط می‌کند. یک عبارت مناسب این جا این است که «برای تپه‌ها بران» (هر چه می‌توانی برو، سریع برو)

❸ BASE خودش را معرفی می‌کند

خوشبختانه در دنیای سیستم‌های محاسباتی توزیع شده، مهندسين باهوش وجود دارند. چگونه سیستم‌های گسترده جهانی مانند بیگ تیل گوگل (Google's BigTa-ble)، آمازون دینامو و کاساندرای فیسبوک با از دست دادن قابلیت اطمینان سر و کله می‌زنند و قابلیت اطمینان سیستم را نیز حفظ می‌کنند؟

پاسخ این سوال در حالی که قطعاً ساده نیست، اما جواب BASE (Basically Available, Soft state, Eventual consistency) است.

در یک سیستم در حالی که BASE نیازمندی اصلی برای قابلیت اطمینان است، فعالیت یا پتانسل (p) از تغییرات داده (H) اساساً آرام کردن آن است. در مقیاس گذاری pH، یک BASE نزدیک تر به آب صابونی است (۱۲) یا شاید نمک دریاچه بزرگ (۱۰).

ادعای چنین اظهار نظری این نیست که میلیون‌ها تراکنش به سرعت رخ نمی‌دهند، بلکه رخ می‌دهند

و تنها محدودیت‌های اعمال شده بر آن‌ها تغییر کرده است. آن محدودیت‌ها در زمان‌های متفاوت با قوانین متفاوت رخ می‌دهند.

اساساً در دسترس: این محدودیت بیان می‌کند که این سیستم آماده بودن داده‌ها را با توجه به قضیه CAP تضمین می‌کند و یک پاسخ به هر درخواستی وجود دارد. اما این پاسخ می‌تواند «شکست» برای داده درخواست شده باشد، و یا ممکن است این داده‌ها در یک وضعیت نامناسب یا در حال تغییر باشد. این مساله مشابه اتفاقاتی است که برای پاس شدن یک چک ممکن است اتفاق بیفتد.

حالت نرم: حالت سیستم می‌تواند در طول زمان تغییر کند. حتی در طول زمانی که هیچ ورودی وجود ندارد ممکن است تغییرات برای رسیدن به «قابلیت اطمینان نهایی» انجام شود. بنابراین این حالت از سیستم همیشه «گرم» است.

قابلیت اطمینان نهایی: سیستم در نهایت قابلیت اطمینان خواهد شد، پس از آن که دریافت ورودی به پایان برسد. دیر یا زود داده به همه جا گسترش پیدا می‌کند، اما سیستم به دریافت ورودی ادامه می‌دهد و قابلیت اطمینان هر تراکنش را پیش از آغاز تراکنش بعدی بررسی می‌کند. جزئیات کامل این موضوع در مقاله «قابلیت اطمینان نهایی - بازبینی شده» آمده است.

❸ نتیجه‌گیری: حرکت به جلو

pH جدید پردازش کردن تراکنش پایگاه داده‌ها، امکان مقیاس گذاری عمومی را با سطوح مقرون به صرفه کارآمد می‌کند. بررسی قابلیت اطمینان هر تراکنش منفرد در هر لحظه از هر تغییر هزینه عظیمی را به یک سیستم که دارای ترلیون‌ها تراکنش است اضافه می‌کند. نیازمندی‌های محاسباتی حتی بیشتر نجومی هستند

«قابلیت اطمینان نهایی» این امکان را به سازمان‌هایی چون باهوا، گوگل، توییتر، آمازون و صدها یا میلیون‌ها سازمان دیگر می‌دهد تا با کاربران‌شان در سراسر جهان تعامل داشته باشند، در حالی که هزینه‌های خود را پایین نگه می‌دارند، سیستم‌هایشان کار می‌کند و کاربران‌شان خوشحال هستند. علاوه بر این‌ها خواهان داشتن قابلیت اطمینان کامل همیشگی هستند. اما همان طور که «دن پری‌جت» در مقاله «BASE یک جایگزین برای ACID» بیان کرده است، باید توازن وجود داشته باشد و قابلیت اطمینان نهایی برای توسعه موثر سیستم‌ها با وجود افزایش نمای داده حاصل از، شبکه‌های اجتماعی، محاسبات ابری و سایر پروژه‌های بزرگ داده وجود داشته باشند. ■



فهرست گذاری راه حلی برای دسترسی سریع به اطلاعات به جای پویش سطر به سطر پایگاه داده؛

فهرست گذاری در پایگاه داده چگونه کار می کند؟

برای یافتن شماره بلوک دیسکی که رکورد مرتبط در آن است مراجعه کند و سپس تنها به آن بلوک دیسک مراجعه کرده و رکورد را استخراج کند. «
بیا بید موضوع را با بحثی آغاز کنیم که اصولاً چرا باید از فهرست گذاری در پایگاه های داده استفاده کنیم. برای فهم بیشتر این موضوع از مثالی ساده شروع می کنیم. فرض کنید که ما دارای جدولی در یک پایگاه داده فرضی و با نام کارمند «Employee» ساخته ایم؛ این جدول دارای سه ستون است که هر یک از ستون ها مشخص کننده ویژگی های خاصی از یک کارمند است که به صورت ساده ای نام، سن و آدرس کارمندان در این فیلدها ذخیره می شود. به طور مشخص هر فیلد دارای نام های «Employee_Age»، «Employee_Name» و «Employee_Address» در بانک اطلاعاتی هستند. حال اگر بخواهیم با استفاده از یک پرس و جو یا کوئری (Query) ساده به اطلاعات داخل آن جدول دست پیدا کنیم که نام آنان عیسی «Jesus» است باید پرس و جوی اس کیوال زیر را اجرا کنیم.

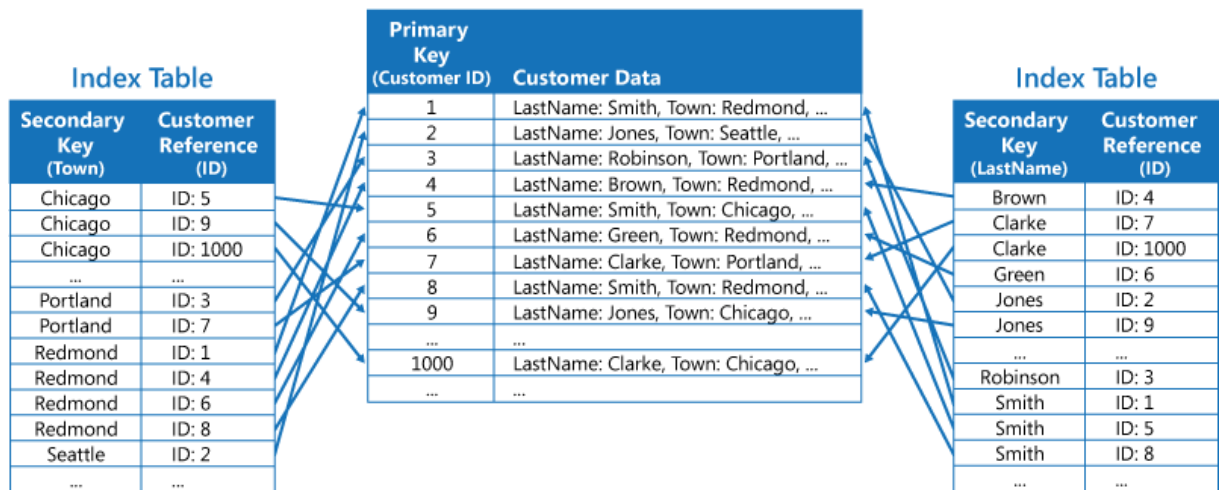
```
SELECT * FROM Employee WHERE Employee_Name = 'Jesus'
```

فهرست گذاری روشی در پایگاه های داده است که برای دسترسی به اطلاعات جداول و رکوردهای ذخیره شده در یک پایگاه داده ها به کار می آید. این روش، روشی ساده است که در زندگی روزمره ما نیز مورد استفاده است. در این روش ما جدولی از تمامی داده ها و شاخص آنان را می سازیم و برای دسترسی به اطلاعات ابتدا به آن مراجعه کرده و با پیدا کردن شاخص به آن شاخص مراجعه کرده و اطلاعات را می خوانیم. در ویکی پدیا فارسی تعریفی از این مفهوم به شکل کلی بیان شده است که در زیر بیان می شود.

«فهرست گذاری (Indexing) در پایگاه داده ها روشی برای ارایه دسترسی سریع به مقادیر یک یا چند ستون است. این بهبود در بازیابی، در ازای هزینه بیشتر در هنگام تغییر / درج / حذف رکوردها است. به بیانی دیگر، استفاده از فهرست، فرایند بازیابی (Retrieval) را سرعت می بخشد، ولی در هنگام حذف و اضافه و یا تغییر مقادیر، هم داده ها باید تغییر کند و هم فهرست باید به روز شود، و این به معنای هزینه بیشتر است.

به عنوان مثال، برای به دست آوردن رکورد حساب با داشتن شماره حساب، به جای آن که در بین تمامی حساب ها جستجو شود، سامانه پایگاه داده ها می تواند ابتدا به فهرست

Fact Table



که حرف اول نام خانوادگی آنان «ت» است در صفحات ۶ تا ۸ هستند. این فهرست برای دستیابی به اطلاعات ایجاد شده است و با دسته‌بندی آنان و معرفی مکان هر دسته، یافتن اطلاعات را آسان‌تر کرده‌ایم.

فهرست‌گذاری و شاخص‌ها در پایگاه‌های اطلاعاتی نیز به این شکل عمل می‌کنند. به این شکل که جدولی از شاخص‌ها و دسته‌بندی‌ها ایجاد می‌شود که شامل محل هر شاخص است؛ سپس اگر در پایگاه‌داده‌ها به دنبال نام شخصی بگردیم همانند مثال مذکور، ابتدا به جدول فهرست رفته و با یافتن دسته مورد نظر، تنها به جستجوی آن دسته خواهد پرداخت و دیگر نیازی به پوشش تمامی اطلاعات در پایگاه‌داده‌ها وجود نخواهد داشت. البته مورد مذکور فقط یک روش از فهرست‌گذاری است و روش‌های دیگری نیز برای این موضوع وجود دارد اما هدف و ساختار کلی آنان به یک شکل است. با وجود این که استفاده از فهرستی از شاخص‌ها باعث بهینه شدن پایگاه‌داده‌ها و کارایی بیشتر در زمان جستجو خواهد بود. اما در برخی موارد بهتر است از این روش استفاده نشود. به طور کلی در استفاده از فهرست‌گذاری باید موارد زیر را در نظر داشت.

هنگامی که با جداول نسبتاً کوچک سر و کار دارید شاخص‌ها کارایی زیادی ندارند. به صورت کلی شاخص‌ها عملکرد را زمانی بهبود می‌بخشند که بر روی ستون‌هایی تعریف شوند که در الحاق «Join» جداول استفاده شده است.

به‌طور دقیق‌تر زمانی که ما پرس‌وجوی فوق را برای جستجو در جدول مذکور برای یافتن فردی با نام عیسی جستجو می‌کردیم، در پشت صحنه دقیقاً چه اتفاقی در حال رخ دادن بوده‌است؟ در این زمان سامانه مدیریت پایگاه‌داده‌ها مجبور است که تمامی سطور و افراد موجود در جدول را یکی به یک برگردد تا این که سطرایی که فیلد نام کارمند در آن عیسی بود را بیابد. همچنین به این خاطر که ما تمامی کارمندانی را که نامشان عیسی است را می‌خواهیم؛ سامانه مدیریت پایگاه‌داده‌ها باید تمامی اطلاعات را یک به یک برگردد. ممکن است در جدول فوق چندین سطر با مشخصه نام کارمند عیسی وجود داشته باشد که نمی‌توان فقط به یکبار جستجو و یافتن یک فرد اکتفا کرد. در چنین مواقعی که ما نیازمند جستجو در یک جدول به دنبال فرد خاص هستیم باید در دیگر سطرها نیز به دنبال فرد مورد نظر بگردیم. در این پرس‌وجو سامانه مدیریت پایگاه‌داده‌ها از سطر اول تا سطر آخر در جدول را به دنبال کارمندان که با نام عیسی هستند، گشته و تمامی آنان را خواهد یافت. بنابراین برای جستجوی چند فرد با نام عیسی، تمامی جدول و اطلاعات ذخیره شده یک به یک توسط مدیر پایگاه‌داده‌ها پوشش شده‌اند. در چنین مواقعی است که می‌گوییم یک پوشش کامل از جدول فوق صورت گرفته‌است.

۳ فهرست شاخص‌ها چه تأثیر بر کارایی پایگاه‌داده‌ها دارند؟

ممکن است تصور کنید که چنین روش جستجویی در جداول پایگاه‌های اطلاعاتی بسیار غیر بهینه هستند. طبیعتاً این روش به این شکل کارایی پایینی داشته و همانند جستجو در فهرست قبول‌شدگان کنکور در روزنامه‌ها به صورت پوشش بالا و خط به خط است که جستجو و پیدا کردن اطلاعات در آن ممکن بهینه نیست. این درست همان جایی است که مبحث فهرست‌گذاری وارد عمل می‌شود و فهرست‌ها مطرح می‌شوند. همان طور که از نام این مطلب مشخص است، فهرست‌ها در بانک‌اطلاعاتی راه‌حل مشکل فوق است. این روش خواهد توانست جستجو در پایگاه‌داده‌ها را بسیار سریع‌تر کند. نکته اساسی و کلیدی در فهرست‌گذاری همان قطع جستجوی خط به خط و پوشش تمام جداول برای یافتن گزینه دلخواه است که با فهرست کردن و فهرست‌گذاری اطلاعات باعث دستیابی سریع به اطلاعات مورد جستجو می‌شود.

۳ فهرست و شاخص چیست؟

برای درک بیشتر این موضوع که فهرست در پایگاه‌داده‌ها دقیقاً چیست، بیایید یک دفترچه تلفن را در نظر بگیریم. برای یافتن اطلاعات و افرادی که در آن قرار دارند می‌توان آن افراد را دسته‌بندی کرد. معمولاً در یک دفترچه تلفن ما آنان را در شاخه‌هایی با اسم اول فامیل آنان مشخص می‌کنیم که از الف تا ی دسته‌بندی شده است. حال در فهرستی از تمامی دسته‌بندی‌های انجام شده در اول آن مشخص شده‌است که افرادی

۳ چه نوع از ساختارهای داده‌ها را می‌توان به عنوان فهرست در نظر گرفت؟ در فهرست‌ها اطلاعات معمولاً به شکل درخت بی (B-tree) نگاهداری می‌شود. بر اساس تعریفی که در ویکی‌پدیا فارسی از درخت بی ذکر شده است، در علوم کامپیوتر، یک درخت بی یا بی‌تری داده‌ساختاری درختی است که داده‌ها را به صورت مرتب‌شده نگه می‌دارد و جستجو، درج و حذف را در زمان مصرفی لگاریتمی میسر می‌سازد. برخلاف درخت‌های جستجوی دودویی متوازن (Balanced binary search tree)، این داده‌ساختار برای سیستم‌هایی که بلاک‌های عظیم اطلاعات را خوانده و می‌نویسند بهینه‌سازی شده است. این ساختار داده معمولاً در پایگاه‌های داده‌ها و سیستم پرونده استفاده می‌شود. یکی از دلایلی که باعث شده‌است که استفاده از درخت بی در جداول فهرست‌ها است به دلیل کارایی بالای آنان و تأثیری است که بر زمان جستجو، حذف و اضافه از آن است که تمامی این اعمال در زمانی لگاریتمی انجام می‌شوند. از دلایل مهم دیگری که درخت بی را به یک روش محبوب تبدیل کرده است می‌توان به امکان مرتب بودن داده‌ها در آن اشاره داشت. به طور کلی سامانه مدیریت پایگاه‌داده‌های رابطه‌ای، تعیین‌کننده ساختاری است که برای فهرست‌ها استفاده می‌شوند اما در برخی شرایط برای برخی برخی از سامانه‌های مدیریت پایگاه‌داده‌های رابطه‌ای می‌توانید مشخص کنید که چه ساختمان داده‌هایی را می‌خواهید در زمان ایجاد فهرست در پایگاه اطلاعاتی خود ایجاد کنید.



❶ فهرست جدول هش چگونه کار می کند؟

در علوم رایانه، جدول درهم سازی یا جدول «هش» نوعی ساختمان داده ها است که مقدارهایی که باید ذخیره شوند را به وسیله تابع هش (درهم سازی) با کلیدهای ویژه ای مرتبط می سازد. عملیات اولیه ای که این جدول تسهیل می کند عمل مراجعه است. به این معنی که کاربر می تواند با سرعتی کارآمد داده مورد نظرش را در آن بیابد. در جدول های هش همچنین افزودن داده های جدید در زمان کم امکان پذیر است. زمان لازم برای جستجو و افزودن وابسته به نوع جدول و میزان داده ها هستند. این زمان می تواند با انتخاب جدول مناسب به مرتبه زمانی O برسد.

جداول هش نیز نوع دیگری از ساختمان داده ها است که ممکن است برای فهرست گذاری برخی فهرست ها «Indexes» به کار رفته باشد. این فهرست ها معمولاً با نام فهرست های هش شناخته می شوند. یکی از دلایل که باعث استفاده از هش در فهرست شده است، به این دلیل است که در زمان هایی تنها به جستجو در مقادیر نیاز داریم. استفاده از هش بسیار بهینه تر از دیگر روش ها است. بنابراین اجرای کوثری یا پرس و جوی دلخواه برای یافتن رشته ای مساوی رشته خاص و مورد نظر ما همانند مثال مذکور در اوایل مطلب بسیار سریع تر خواهد بود. برای نمونه اجرای کوثری ابتدای مقاله با استفاده از فهرست هش بر روی نام کارمندان جدول هش منفعت برده و با سرعت بهتری به نتیجه دلخواه خواهد رسید. به خاطر این موضوع، جداول هش به طور گسترده در نرم افزارها، به خصوص در آرایه های شرکت پذیر، ساختار فهرست پایگاه داده، حافظه های نهان و مجموعه ها استفاده می شوند.

در جداول هش معمولاً به این شکل است که اطلاعات و مقادیر هر فیلد و رکود را در یک آرایه که به سادگی جدول هش نام دارد می گذارد که دارای مشخصه های کلید، مقدار و طول است. جدول هش اندیس یک کلید را محاسبه می کند و از آن اندیس برای جای دادن آن کلید در جدول استفاده می کند. به طور کلی تصور کنید که می خواهید اطلاعات چندین هزار نفر را در یک پایگاه داده ذخیره کنید؛ راه حل ابتدایی و ساده این است که تمامی آنان را در یک جدول ساده ریخته و ذخیره کنیم. بعد این همان طور که ذکر شد در صورتی که بخواهیم به دنبال فردی بگردیم باید n بار که n تعداد افراد داخل پایگاه داده است را یک به یک بگردیم. در این صورت مرتبه زمانی n خواهد بود. راه حل جدول هش در این زمان با ایجاد کلیدی که از آدرس مقادیر ذخیره شده را در پایگاه داده به صورت آدرس حافظه ای محل ذخیره نام ها به بدست آورده است و ذخیره آن به عنوان کلید در جدول هش کلیدی را به هر یک از آنان نسبت می دهد سپس نام آنان را نیز در کنار هر یک از کلیدها متناسب با نام قرار می دهد. مثلاً کلید و نام متناظری که برای فرد مذکور با نام عیسی ساخته شده است به این شکل در جدول هش قرار می گیرد.

“Jesus => 0x28939

برای جست و جو و یافتن فردی با نام عیسی در جدول با استفاده از کوثری نوشته شده در اوایل مطلب، اگر از جدول هش استفاده کرده باشیم، ابتدا مدیر پایگاه داده ها به جدول فوق مراجعه کرده و بر اساس نام و کلید یافته شده به آدرس حقیقی و مکان نام-کارمند در داخل جدول کارمند مراجعه خواهد کرد. طبیعی است که مراجعه به جدول فوق برای یافتن آدرس آن فرد سریع تر از پویش جدول و سطور است و دستیابی به اطلاعات سریع تر خواهد بود.

❷ معایب فهرست درهم سازی در چیست؟

عیبی که برای جدول هش عنوان می کنند، مرتب نبودن اطلاعات در این روش است و همچنین تعداد زیادی از پرس و جوی مورد استفاده برای استفاده در پایگاه های داده وجود دارند که جداول هش کمکی در بهبود اجرای آنان ندارند. به عنوان نمونه، قصد

دارید با استفاده از یک پرس و جوی ساده اس کیوال که اکثراً آن آشنایی دارید؛ به دنبال افرادی در کارمندان بگردید که سن آنان از ۴۰ سال کمتر باشد. در نظر بگیرید در جدول فیلدی با نام سن-کارمند (Employee Age) نیز وجود دارد. آیا خواهید توانست این پرس و جو را با استفاده از روش جدول هش انجام دهید؟ خوب، این کار ممکن نیست؛ زیرا جدول هش در مواقعی کاربرد خوبی از خود نشان خواهد داد که بخواهید به دنبال موارد کاملاً مشخصی چون همانند رشته یا رقم خاصی باشید مثلاً عنوانی آیا مساوی با عبارت خاصی است یا آیا شباهت به آن عبارت دارد. (مشابه این عبارت بعد از کوثری انتخاب مشابه زیر)

WHERE name = ‘Jesus’

این موضوع به این دلیل رخ می دهد که چیزی که در داخل کلید جدول هش عنوان شده است برای دسترسی به آن داده است و علاوه بر آن هیچ گونه نظم و ترتیب خاصی در چینش آنان در داخل فهرست جدول هش وجود ندارد. بنابراین اگر بخواهید بر اساس ترتیب فیلد سن-کارمند در جدول هش به دنبال کارمندان زیر ۴۰ سال باشید، به دلیل آن که ترتیبی در فهرست جدول هش وجود ندارد. نمی توانید بگوید این افراد را به صورت سریع برایتان بیابد؛ همین دلیل است که باعث شده که جدول هش به عنوان یک داده ساختار پیش فرض برای فهرست گذاری در سامانه های مدیریت پایگاه داده ها استفاده نشود. به طور کلی جداول هش کارایی و انعطافی که داده ساختار درخت بی ارائه می کند را نداشته و به خوبی آن نیست. یکی از مشکلات جدول هش تصادم است. در مثال قبل اگر کلید ویژه دو نام متفاوت از ده هزار نام اولیه مان یکی می شد باید کدام را در خانه مربوطه آرایه ذخیره می کردیم؟ به این اتفاق تصادم گفته می شود.

❸ فهرست چگونه کارایی را افزایش می دهد؟

به این دلیل که فهرست اساساً یک ساختمان داده ها برای ذخیره مقادیر ستون های جدول است. جست و جو در این مقادیر بسیار سریع تر خواهد بود. همچنین اگر فهرست از ساختار داده درخت بی که یکی از داده ساختار معروف است استفاده کند، اطلاعات و مقادیر داخل جدول نیز به صورت مرتب شده خواهند بود که مزایای خاص خود را دارد. وجود اطلاعات در یک جدول به صورت مرتب شده مزایایی را به همراه دارد که همان طور که در قسمت معایب جدول هش ذکر شد. برای بهبود کارایی و عملکرد پایگاه داده ها در جست و جو ... تاثیر زیادی دارد.

حال بیایید تصور کنیم که در پایگاه داده ها خود یک فهرست با ساختمان داده ها درخت بی برای ستون نام-کارمند (Employee_Name) ساخته ایم. این به این معنی خواهد بود که اگر به دنبال کارمندی با نام عیسی «Jesus» گشتیم؛ این کار را توسط کوثری اس کیوال ساده ای که در اوایل مقاله نوشتیم انجام خواهیم داد. در این زمان برای جست و جو نیازی به پویش و جست و جوی کل مقادیر برای یافتن نام کارمند مورد نظر «Jesus» نیست و در عوض پایگاه داده ها از فهرست برای یافتن کارمندی با نام عیسی «Jesus» استفاده خواهد کرد. به دلیل این که فهرست بر اساس حروف الفبا مرتب خواهد شد؛ جست و جو به دنبال فردی با نام عیسی بسیار سریع خواهد بود. فهرست فوق بر اساس حروف الفبا مرتب است و اگر به دنبال نام فردی مثلاً عیسی «Jesus» بگردیم؛ این نام با حرف جی لاتین «J» آغاز شده است. در این زمان است که فهرست مرتب شده به کار می آید و به جای این که پایگاه داده ها در تمامی جدول به دنبال آن فرد باشد؛ تنها در آن افرادی جست و جو را انجام خواهد داد که حرف اول آنان جی «J» است. همچنین در داخل جداول اشاره گری را نیز در سطرها ی جدول ذخیره می کند که باعث می شود بتوان مقادیر دیگر ستون ها را نیز بازیابی کرد.

به این صورت ذخیره کلیه مقادیر یک جدول به شکل مرتب شده در فهرست نیز کار عقل مدارانه ای نیست و همانند کپی و گرفتن پشتیبان از یک جدول است که باعث هدر

رفتن فضای ذخیره‌سازی و بزرگی بدون دلیلی پایگاه‌داده‌ها خواهد بود که برای مقادیر خیلی بزرگ بسیار غیربهرینه و غیرمنطقی است. بنابراین همواره در فهرست ستونی که بیشتر به آن نیاز داریم را قرار می‌دهیم. به عنوان نمونه برای سامانه اینترنتی یک بانک و پایگاه داده حساب‌های کارتهای اعتباری در آن سامانه، شماره کارت را در فهرست نگاه می‌داریم.

دقیقا چه چیزی در فهرست یک پایگاه‌داده‌ها وجود دارد؟

حال شما می‌دانید که فهرست پایگاه‌داده‌ها بر یک ستون از یک جدول ساخته می‌شود و در این فهرست مقادیر آن ستون خاص قرار گرفته‌است. اما این موضوع نیز مهم است که درک کنیم که فهرست مذکور به هیچ وجه اطلاعاتی که در دیگر ستون‌ها در آن جدول وجود دارد را نیز در خود نگاه نمی‌دارد. برای مثال اگر ما یک فهرست از نام-کارمندان (Employee_Name) ایجاد کنیم به این معنی است که سن-کارمند (Employee_Age) و آدرس-کارمند (Employee_Address) هم ستون‌های دیگر جدول کارمند هستند؛ در داخل فهرست پایگاه‌داده‌ها مذکور قرار ندارند.

یک اشاره‌گر نیز توسط فهرست در سطور جدول ساخته می‌شود. بنابراین سوال این است که مقداری که در جست‌وجوی آن هستیم در یک فهرست (مانند عیسی «Jesus») پیدا شود؛ مابقی فیلدهایی را که مرتبط با این فیلد یافت شده‌است را چگونه باید پیدا کنیم یا به آنان دست‌یابیم (فیلدهای دیگر مانند اطلاعات اطلاعات عیسی یا سن او)؟ خوب، موضوع کاملا ساده است؛ فهرست پایگاه‌های داده علاوه بر اطلاعات فیلد مورد نظر، از اشاره‌گرهایی را نیز در داخل فیلدهای مرتبط با این سطر نیز ذخیره کرده است. یک اشاره‌گر در واقع یک ارجاع است به مکانی از حافظه است که اطلاعات در دیسک ذخیره شده‌اند. پس علاوه بر مقادیر ستون که در فهرست ذخیره شده است؛ اشاره‌گری به آن سطر از جدول که آن ستون قرار دارد نیز وجود دارد که هرگاه مقداری را در فهرست یافتیم؛ به سطر مورد نظر نیز دسترسی خواهیم داشت. به عنوان نمونه در مثال مذکور و در کنار مقادیر گوناگون هر ستون مثلا کنار نام-کارمند در فهرست اشاره‌گری نیز به سطر مورد نظر قرار دارد. «Jesus», 0x82829» که در این مثال عبارت «0x82829» به آن سطر و مکانی از حافظه که کارمند فوق در آن ذخیره شده است اشاره دارد. بدون این اشاره‌گرها شما فقط یک مشخصه را خواهید داشت که اگر بخواهید به مشخصات مرتبط با آن مشخصه دسترسی داشته باشید؛ با مشکل مواجه خواهید شد. مخصوصا این که اطلاعات فوق بر اساس ترتیب دیگری از جدول اصلی مرتب شده‌اند.

چگونه می‌فهمید پایگاه‌داده‌ها چه زمانی از یک فهرست استفاده می‌کند؟

زمانی که شما کوئری زیر را اجرا کنید؛ پایگاه‌داده‌ها بررسی می‌کند که آیا برای ستون مورد نیاز فهرستی پرس‌وجو شده‌است یا خیر. با فرض این که برای ستون نام-کارمند (Employee_Name) دارای یک فهرست باشد؛ پایگاه‌داده‌ها بررسی خواهد کرد که آیا استفاده از فهرست برای این پرس‌وجو مناسب باشد. در صورتی که پایگاه‌داده‌ها تصمیم گرفت که باید از فهرست استفاده شود، آنگاه فهرست مورد استفاده قرار خواهد گرفت. به‌خاطر این که در برخی حالت ممکن است که استفاده از فهرست معنی خاصی نداشته باشد و بهتر باشد از فهرست استفاده نکند.

آیا می‌توان پایگاه‌داده‌ها را مجبور کرد در هر پرس‌وجو از فهرست استفاده کند؟

به طور کلی، شما نمی‌توانید به پایگاه داده بگویید که در چه زمان‌هایی از فهرست استفاده کند. این تصمیمات توسط خود پایگاه‌داده‌ها اتخاذ می‌شوند. اگر چه شایان ذکر است که در بسیاری از پایگاه‌داده‌ها (مانند اوراکل و MySQL) شما قادر خواهید بود که دقیقا مشخص کنید که پایگاه‌داده‌ها از فهرست استفاده کند.

روش ساخت فهرست توسط اس‌کیوال چگونه است؟

ساخت یک فهرست با استفاده از یک ستون؛ در زیر پرس‌وجوی «کوئری» اس‌کیوال را می‌بینید که برای ساخت یک فهرست برای ستونی با نام-کارمند (Employee_Name) در جدول کارمند است.

```
CREATE INDEX name_index ON Employee (Employee_Name)
```

ساخت یک فهرست با استفاده از چند ستون: اگر بخواهید دو یا چند ستون را در یک فهرست داشته‌باشید؛ باید پرس‌وجو «کوئری» زیر را اجرا کنید تا فهرست را اجرا نمایید. در مثال زیر ما از دو ستون استفاده کرده‌ایم؛ یکی ستون نام-کارمند (Employee_Name) و ستون دیگر سن-کارمند (Employee_Age) هستند.

```
CREATE INDEX name_index ON Employee (Employee_Name, Employee_Age)
```

مثالی خوب برای فهم بیشتر مفهوم فهرست پایگاه‌داده‌ها؛ اگر بخواهیم قیاس خوبی برای تعریف فهرست در یک بانک اطلاعاتی عنوان کنیم، باید از فهرست در یک کتاب نام برد. فهرست‌ها در کتاب‌ها معمولا در اول کتاب آمده و از عناوینی که در کتاب وجود دارد به همراه شماره صفحه تشکیل شده‌است. در برخی کتب انگلیسی و حتی برخی کتب فارسی، فهرست مطالب در آخر کتاب قرار دارد. مثلا اگر شما کتابی را در مورد حیوانات مثلا سگ خریداری کنید؛ در این کتاب اطلاعات مختلف صفحات زیادی وجود دارد که به بخش و فصل‌های مختلف تقسیم شده‌اند. هر فصل و بخش نیز خود دارای عناوین مختلفی هستند که برای پیدا کردن آنان به فهرست مطالب رجوع می‌کنید. مثلا اگر به دنبال مطلبی درباره انواع نژاد سگ‌های شکاری باشید باید به فهرست مراجعه کرده و با مشاهده عنوان مرتبط با موضوع، به صفحه نوشته شده در برابر عنوان مراجعه و آن بخش را مطالعه کنید؛ این کار باعث صرفه‌جویی در وقت و افزایش سرعت دسترسی به مطلب خواهد بود. اگر کتاب دارای فهرست نبود؛ شما مجبور بودید تا تمامی مطالب کتاب را از اول پویش کنید تا به عنوان دلخواه دست‌یابید. در این زمان بهترین حالت برای پیدا کردن عنوان این بود که عنوان در اوایل کتاب باشد و بدترین حالت نیز زمانی است که عنوان در اواخر کتاب باشد.

فهرست در پایگاه‌داده‌ها نیز همین‌طور است و با استفاده از فهرست‌ها شما به مقادیر و ستون دلخواه خواهید رفت؛ بدون آن که وقت زیادی را صرف پویش کلیه اطلاعات نمایید تا اطلاعات خود را پیدا کنید. فهرست در پایگاه‌داده‌ها تقریبا چنین کاری را انجام می‌دهد که فهرست کتاب انجام می‌دهد.

استفاده از فهرست چه مقدار هزینه دارد؟

خوب، همانند هر روش دیگری در دنیای واقعی، فهرست در پایگاه‌داده‌ها نیز معایب و مزایای خاص خود را دارد. یکی از معایب استفاده از فهرست همان‌طور که شاید تا الان پی برده باشید، لزوم استفاده از فضای بیشتر برای نگهداری مقادیر در فهرست است که ممکن است فضای بیشتری برای پایگاه‌داده‌ها نیاز باشد. نکته دیگری هم که برای فهرست در پایگاه‌داده‌ها می‌تواند باعث کاهش کارایی شود؛ این است که در هر زمان که شما نیاز به درج، حذف و ویرایش اطلاعات در پایگاه اینترنتی یا جدول خاصی که دارای فهرست است را داشته باشید تغییراتی که در اطلاعات اعمال می‌شود باید در فهرست نیز اعمال شود؛ فراموش نکنید که اطلاعاتی که در فهرست ذخیره می‌شود با اطلاعات واقع در ستون مورد نظر برای فهرست باید مطابقت داشته باشند و در هر بار تغییر، تغییرات باید همگام شوند. بنا به قاعده‌ای کلی، هر فهرست می‌تواند فقط در زمانی که نیاز به پرس‌وجو داشتیم ایجاد شود. ■



MySQL®

موتورهای پایگاه داده در مای اس کیوال

می‌توانید از نسخه موجود در پایگاه اینترنتی اوراکل استفاده کرده و یا آن را خودتان از منبع کامپایل و نصب کنید.

کامپایل نرم‌افزار از پایه شاید برای توسعه‌دهندگان سیستم‌عامل ویندوز کمی ناآشنا باشد، اما این موضوع برای توسعه‌دهندگانی که از سیستم‌عامل‌های شبه یونیکس استفاده می‌کنند، کاملاً موضوعی عادی به شمار می‌آید.

مای اس کیوال از سه موتور پایگاه داده استفاده می‌کند که آی‌زم «ISAM» مای‌آی‌زم «MyISAM» و هیپ «HEAP» نام دارند؛ دو نوع موتور پایگاه داده‌ها دیگر برای پایگاه‌های داده نیز با نام‌های اینودی‌بی «InnoDB» و برکلی «Berkley BDB» نیز از دیگر موتورهای پایگاه داده در مای اس کیوال به شمار می‌آیند.

آی‌زم «ISAM»

موتور آی‌زم یک موتور پایگاه داده‌ها است که روش مناسب و سریع برای مدیریت جداول در پایگاه داده‌ها است. این موتور با این ایده طراحی شده است که معمولاً در اکثر پایگاه‌های داده در بیشتر مواقع به جای آن که نیاز به به‌روزر کردن ویرایش اطلاعات باشد؛ نیاز به مشاهده و اجرای پرس‌وجو احساس می‌شود. بنابراین این موتور برای استفاده در مای اس کیوال از سرعت بالایی در خواندن اطلاعات جداول دارد و به راحتی می‌تواند بدون درگیر کردن منابع سیستم در حافظ اصلی و جانبی، اطلاعات را به راحتی نمایش دهد. با وجود این، دو نکته منفی درباره این موتور را می‌توان به عدم پشتیبانی از تراکنش (transactions) اشاره داشت یا همچنین نکته منفی مهم دیگر به عدم تحمل خطا در این موتور اشاره داشت. برای مثال اگر دیسک سخت شما از کار بیفتد و یا با مشکل مواجه شود، موتور دیگر نمی‌تواند که اطلاعات را بازیابی کرده و آنان را مجدداً احیا کند. اگر از این موتور در کاربردی تجاری با امکان خطرات بالا استفاده می‌کنید؛ بهتر است از پایگاه داده خود پشتیبانی زنده و ۲۴ ساعته داشته باشید که انجام این کار با استفاده از برخی ویژگی‌های موجود در مای اس کیوال به راحتی قابل انجام است.

مای‌آی‌زم «My ISAM»

مای‌آی‌زم را می‌توان نسخه سفارشی مای اس کیوال از موتور آی‌زم دانست

تصور کنید که چه اتفاقی می‌افتاد اگر به عنوان یک راننده خودرو قادر بودید که براساس نیازهای مختلف که در سفر و یا استفاده از خودرو داشتید به یک گاراژ مراجعه کرده و موتور شما توسط یک مکانیک موتور ماشین خود با موتور دیگری تعویض می‌گردید. در این حالت شما برای هر شرایطی قادر بودید که از موتور دل‌خواه استفاده کنید؛ موتوری برای مسافرت، موتوری برای مسابقه و ... در مای اس کیوال دقیقاً می‌توانید به چنین اقدام کرده و تعویض موتور در هر شرایط قابل انجام خواهد بود. به طور کلی تعویض موتور برای موتورهای پایگاه داده‌ها مورد استفاده توسط نرم‌افزار مدیریت پایگاه داده‌ها رابطه‌ای «RDBMS» مای اس کیوال یکی از مزایای مهم آن به شمار می‌رود. به این صورت که هر وقت نیاز داشتید، فراخور نیاز خود بتوانید به تغییر و تعویض موتور پایگاه داده‌ها به مورد دلخواه اقدام کنید.

قطعاً ممکن است همان موتور پیش‌فرضی که در مای اس کیوال مورد استفاده است برای شما کافی باشد؛ با این حال در برخی شرایط استفاده از موتورهایی که برای آن شرایط خاص عملکرد بهتری را به نمایش می‌گذارند، به بازدهی بهتری منجر خواهد شد. در برخی از مواقع حتی ممکن است نیاز داشته باشید با استفاده از دانش برنامه‌نویسی و رابط برنامه‌نویسی نرم‌افزار «API» مای اس کیوال خود دست به ایجاد یک موتور سفارشی بزنید. یکی از ویژگی‌های نرم‌افزارهای آزاد / متن‌باز نیز همین امکان سفارشی‌سازی بالای آنان با نگاه به کدهای دیگر است که باعث بهبود کارایی نرم‌افزار به دلیل قابلیت سفارشی‌سازی و انطباق بالا خواهد بود. در این مورد نیز به راحتی می‌توانید موتور خود را با انشعاب و سفارشی کردن یکی از موتورهای فعلی ایجاد کنید.

در نرم‌افزار مدیریت پایگاه داده‌های رابطه‌ای مای اس کیوال تقریباً اکثر موتورهای پایگاه داده‌های موجود برای مای اس کیوال را می‌توان استفاده کرد؛ به شرط آن که در زمان کامپایل نرم‌افزار آن‌ها فعال بوده و غیرفعال نشده باشند؛ معمولاً در اکثر توزیع‌های گنولینوکس و یا نسخه موجود در داخل پایگاه اینترنتی اوراکل از نسخه‌ای استفاده شده است که از اکثر موتورهای مرسوم اس کیوال استفاده می‌کند. با این حال اگر در توزیع شما از مای اس کیوال استفاده شده است که دارای موتور مورد نیاز شما نیست

که موتور پیش‌فرض در نرم‌افزار مدیریت پایگاه داده رابطه‌ای مای‌اس‌کیوال است. به علاوه فراهم کردن امکاناتی برای فهرست‌گذاری و مدیریت فیلد که در آیزم وجود نداشته‌اند. مای‌آیزم از یک شیوه قفل کردن جدول «table locking» بهره می‌برد که باعث بهبود کارایی بیشتر در زمان خواندن و نوشتن‌های همزمان در پایگاه‌داده خواهد بود. نکته منفی برای این موتور این است که در اکثر مواقع نیاز دارید که دستور بهینه‌سازی جدول را در اکثر مواقع اجرا کنید؛ تا فضاهای از دست رفته توسط الگوریتم به‌روزرسانی را از بازیابی کند. مای‌آیزم همچنین از چند یک ابزار افزونه «Extensions» مفید نیز برخوردار است؛ مانند افزونه «MyISAMchk» که وظیفه تعمیر فایل پایگاه‌داده‌ها را به عهده داشته و یا «MyISAMPack» که ابزاری برای فضاهای از دست‌رفته است.

موتور بر روی سرعت بالا تاکید داشته و عملکرد خوبی را در این زمینه بر جا گذاشته است و عملیات خواندن در آن بسیار سریع است. این به همان دلیل است که نرم‌افزار مدیریت پایگاه‌داده‌ها رابطه‌ای مای‌اس‌کیوال در اکثر پایگاه‌های اینترنتی مورد استفاده است و به یکی از سریع‌ترین مدیران پایگاه‌داده‌ها در بستر اینترنت تبدیل شده است. سرعت خواندن اطلاعات و نمایش اطلاعات در پایگاه‌های اینترنتی از اهمیت بالایی برخوردار است زیرا که اکثر عملیاتی که برای استفاده از پایگاه‌داده‌ها در اینترنت استفاده می‌شود؛ معطوف به خواندن اطلاعات از جداول پایگاه‌داده‌ها است و به همین دلیل هم اکثر IPPها کاربران را مجبور به استفاده از این موتور کرده‌اند.

3 هیپ «HEAP»

موتور هیپ اجازه می‌دهد جداول موقتی در داخل حافظه اصلی اقامت داشته باشند. اقامت در حافظه اصلی باعث بهبود و افزایش سرعت عمل موتور خواهد بود و سرعت این موتور را از موتورهای آیزم و مای‌آیزم بیشتر خواهد کرد. با این حال این نوع ذخیره داده در حافظه جانبی بسیار فرار است و اگر اطلاعات قبل از این که سیستم خاموش شوند، هنوز در حافظه اصلی باشند؛ اطلاعات فوق به راحتی از دست خواهند رفت. هیپ علاوه بر این از هدر رفتن فضای خالی در زمانی که برخی سطرها پاک می‌شوند؛ جلوگیری می‌کند. همچنین جداول هیپ در مواقعی که می‌خواهید انتخاب‌هایی تودرتو برای انتخاب و دستکاری داده‌ها داشته باشید؛ گزینه مناسبی به شمار می‌آید. لازم است یادآوری کنم که بعد از این که کارتان با جدول تمام شد آن را «destroy» کنید؛ بگذارید مجددا تاکید کنم که هرگز فراموش نکنید؛ هرگاه کار شما با جدولی تمام شد؛ آن را «destroy» کنید.

3 اینودی بی «InnoDB» بزرگلی دی بی «Berkley»

موتورهای اینودی بی و بزرگلی دی بی را می‌توان موتورهایی دانست که محصولات مستقیم فناوری هستند که باعث انعطاف مای‌اس‌کیوال و رابط برنامه‌نویسی نرم‌افزار مای‌اس‌کیوال ++ شده‌اند. تقریباً تمامی چالش‌هایی را که در استفاده از مای‌اس‌کیوال داشته‌اید؛ مستقیماً از این حقیقت ناشی می‌شود که موتور پایگاه‌داده‌ها آیزم و مای‌آیزم در تراکنش و کلیدهای خارجی مشکلاتی را داراست. در هر حال پشتیبانی از این دو موضوع در این دو موتور بسیار بهتر از سه موتور معرفی شده است و اگر این دو قابلیت برای شما مهم هستند، باید از این دو موتور پایگاه‌داده‌ها استفاده کنید. در اکثر مواقع از موتور اینودی بی بیشتر از موتور پایگاه‌داده‌ها برکلی استفاده

می‌شود. با وجود این سرعت در هنگام خواندن اطلاعات در دو موتور مذکور اینودی بی «InnoDB» بزرگلی دی بی «Berkley» بسیار پایین‌تر از سه موتور معرفی شده دیگر است.

با این حال اگر توانایی بیشتری را در خود احساس می‌کنید؛ می‌توانید موتور پایگاه‌داده‌های مختص به خود را نیز با استفاده از رابط برنامه‌نویسی نرم‌افزار مای‌اس‌کیوال ++ نوشته و از آن استفاده کنید. رابط برنامه‌نویسی نرم‌افزار مذکور تمامی مواردی را که برای ساخت یک موتور پایگاه‌داده‌ها، دسترسی به سطرها، فیلدها، جداول، پایگاه‌های داده‌ها، ارتباطات، حساب‌های امنیتی و دیگر موارد مورد نیاز و توابع پایگاه‌داده‌ها را که نیاز دارید را در اختیار شما قرار خواهد داد. آموزش استفاده از این رابط برنامه‌نویسی نرم‌افزار در این نوشته امکان‌پذیر نیست؛ با این حال در همین حد اشاره خواهیم کرد که رابط برنامه‌نویسی نرم‌افزار مای‌اس‌کیوال ++ برای توسعه‌دهندگان وجود دارد و همین رابط برنامه‌نویسی نرم‌افزار است که پشت قابلیت تعویض موتور در مای‌اس‌کیوال است و این قابلیت را مدیون مای‌اس‌کیوال ++ هستیم. با این حال این سبک از موتورهای قابل نصب در مای‌اس‌کیوال می‌تواند به موارد دیگری مانند ساخت یک فراهم کننده اکس‌ام‌ال بومی در مای‌اس‌کیوال نیز استفاده شود.

تعویض موتور، تمامی انعطاف موجود را توسط یک افزونه مای‌اس‌کیوال که برای انسی اس‌کیوال نوشته شده است؛ فراهم آورده است. این کار توسط پارامتر نوع «TYPE» انجام می‌شود. همچنین مای‌اس‌کیوال به شما این اجازه را نیز خواهد داد که برای هر یک از جداول مختلف، موتور پایگاه داده خاصی را اختصاص دهید. در برخی از مواقع به آنان قالب جدول «table format» نیز اطلاق می‌شود. مثالی که در زیر نوشته شده‌است؛ یک کوئری ساده اس‌کیوال برای ساخت چند جدول است که برای هر یک از آنان نوع جدول به انواع مختلف تغییر یافته است. به ترتیب انواع به کار رفته در مثال زیر موتورهای مای‌آیزم، آیزم و هیپ هستند. گفتنی است تمامی جداول ساخته شده از فیلدهای یکسانی برخوردارند؛ با این تفاوت که نام آن‌ها به همراه نوع آن‌ها با هم تفاوت دارد.

```
CREATE TABLE tblMyISAM (
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id),
    value_a TINYINT
) TYPE=MyISAM
CREATE TABLE tblISAM (
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id),
    value_a TINYINT
) TYPE=ISAM
CREATE TABLE tblHeap (
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id),
    value_a TINYINT
) TYPE=Heap
```




نکند، از موتور پیش فرض مای‌آی‌زم استفاده خواهد کرد که برای کسانی که حتماً به موتور خاص نیاز دارند اصلاً خوب نیست. این در حالی است که ممکن است در آینده چنین قابلیت‌هایی در مای‌اس‌کیوال افزوده شود و برنامه‌ریزیهایی برای درج پیغام هشدار در هنگام نیافتن موتور مورد نظر در مای‌اس‌کیوال در ذهن توسعه‌دهنده‌های مای‌اس‌کیوال وجود دارد تا این مشکل را در آینده حل کنند. با این وجود و در حال حاضر تنها کاری که از شما بر می‌آید آن است که با استفاده از دستور بالا مشاهده کنید که جدول فوق از چه موتوری استفاده می‌کند. اگر از یک میزبان اینترنتی استفاده می‌کنید؛ از آن شرکتی که خدمت فوق را دریافت کرده‌اید و از قسمت پشتیبان حتماً نصب بودن موتور دلخواه خود را جویا شوید.

انتخاب بیشتر به معنی بازدهی بیشتر خواهد بود؛ با وجود این که در مای‌اس‌کیوال از یک موتور خوب به صورت پیش فرض استفاده شده است که از سرعت و بازدهی تقریباً مناسب برخوردار است؛ چه نیازی به استفاده و تعویض موتور پایگاه داده به دیگر موارد وجود خواهد داشت؟ همچنین به وجود این که موارد مختلف جهت استفاده به عنوان موتور در پایگاه داده عرضه شده‌است؛ چرا باید موتور اختصاصی ساخته و استفاده کنیم؟ جواب سوالات فوق بسیار ساده است. نکته مهم در این است که هر گاه بتوانید انتخاب بیشتری داشته باشید و دستتان در ساخت ابزاری سفارشی مورد نظر خودتان باز باشد؛ آنگاه موتور فوق بهترین تطابق را با نیازهای شما خواهد داشت.

قطعاً مای‌آی‌زم موتوری سریع برای پایگاه داده به حساب می‌آید. با این حال اگر شما از قابلیت انتقال «transactional» در پایگاه داده استفاده کرده و به آن نیاز دارید؛ حتماً باید از دو موتور پایگاه داده‌ای که از این قابلیت پشتیبانی دارند؛ استفاده کنید. با این حال با استفاده از قابلیت در مای‌اس‌کیوال که به شما اجازه استفاده از موتوری خاص را تنها بر روی یک یا چند جدول خاص می‌دهد؛ شما به راحتی خواهید توانست از یکی از موتورهای پایگاه داده برکلی یا اینودی‌بی برای آن جدول که به قابلیت مذکور نیاز دارد استفاده کرده و برای دیگر جداول از موتور پیش فرض مای‌آی‌زم استفاده کنید تا سرعت بیشتری را برای نمایش اطلاعات به همراه پشتیبانی از انتقال «transactional» را در پایگاه داده را نیز داشته باشید. به صورت کلی نکته مهم و کلیدی در نرم‌افزار مدیریت پایگاه داده رابطه‌ای «RDBMS» مای‌اس‌کیوال را می‌توان انعطاف بالای آن دانست. ■

همچنین می‌توانید از «ALTER TABLE» برای انتقال یک جدول که از موتور خاصی استفاده می‌کند به جدول دیگر با موتور پایگاه داده‌های دیگر استفاده کنید. برای مثال در کد نوشته شده اس‌کیوال زیر از دستور «ALTER TABLE» برای انتقال جدول به جدول دیگر استفاده شده است. اولین جدول از موتور مای‌آی‌زم استفاده می‌کند که با دستور زیر نوع «TYPE» آن به اینودی‌بی تغییر خواهد یافت که به معنای تغییر موتور در جدول مقصد است.

```
ALTER TABLE tb1MyISAM CHANGE TYPE=InnoDB
```

مای‌اس‌کیوال این کار را در سه مرحله انجام خواهد داد. در اولین مرحله یک رونوشت از جدول مبدأ تهیه خواهد کرد. در مرحله بعدی تمامی تغییرات وارد شده در یک صف قرار می‌گیرند تا این که عملیات انتقال از یک موتور به موتور دیگر به طور کامل انجام شود. سپس در مرحله بعدی تغییراتی که در صف قرار داشتند یک به یک بر روی جدول مقصد اعمال شده و عملیات انتقال با موفقیت به پایان خواهد رسید.

نکته، راهی میانبر برای «ALTER TABLE»؛ اگر می‌خواهید به شکلی ساده جدول خود را از یک جدول بهره‌مند از مای‌آی‌زم به جدولی بهره‌مند از مای‌آی‌زم منتقل کنید؛ به راحتی قادر خواهید بود تا از دستوری کوتاه‌تر که روشی میانبر است استفاده کنید.

```
mysql_convert_table_format
```

برای این که مشاهده کنید که یک جدول خاص از چه موتور پایگاه داده‌ای استفاده می‌کند؛ می‌توانید از دستور «SHOW TABLE» استفاده کنید. با استفاده از این دستور در داخل کد انسی اس‌کیوال به راحتی قادر به تعیین نوع موتور استفاده شده در هر جدول خواهید بود. خروجی دستور فوق یک جدول است که اطلاعات مختلفی را از یک جدول به نمایش در می‌آورد؛ با این حال آن قسمت از اطلاعاتی که موتور مورد استفاده در جدول اشاره دارد؛ اطلاعاتی است که در بخش نوع «TYPE» قرار دارد.

```
SHOW TABLE STATUS FROM tb1InnoDB
```

نکته، راهی جایگزین برای نمایش اطلاعات جدول؛ به جای استفاده از دستور بالا و نوشتن دستور «SHOW TABLE»، می‌توانید از دستور زیر استفاده کنید؛ که اطلاعاتی مشابه با دستور قبلی را نمایش خواهد داد.

```
SHOW CREATE TABLE [TableName]
```

در انتها؛ چیزی بدتر از این نیست که موتور مورد نظر شما در مای‌اس‌کیوال وجود نداشته باشد و در همراه آن نباشد؛ همانطور که ذکر شد برای استفاده از یک موتور باید همراه با مای‌اس‌کیوال در زمان کامپایل، کامپایل شود تا بتوان از آن استفاده کرد. با وجود این اگر موتور فوق به همراه مای‌اس‌کیوال مورد استفاده شما نباشد و بخواهید از آن موتور در یک جدول استفاده کنید؛ پیغام خطایی برای شما نمایش داده نخواهد شد تا متوجه عدم وجود آن موتور در مای‌اس‌کیوال شود. اگر مای‌اس‌کیوال موتور مورد نظر را پیدا

ارتباط پایگاه داده‌ها با دنیای خارج



نویسنده
وحید سپهرزادلو

حاصله نوشتن با زبان C. اگر شما هم مثل من هیچ علاقه‌ای به نوشتن با C ندارید می‌توانید از یک اکستنشن با نام multicorn استفاده کنید. این اکستنشن به شما امکان می‌دهد که Warpper را با زبان Python پیاده‌سازی کنید. نوشتن Warpper کار سختی نیست پیشنهاد می‌کنم به صفحه <http://multicorn.org> مراجعه کنید

FDW FUN

```
DROP EXTENSION IF EXISTS www_fdw CASCADE;
CREATE EXTENSION www_fdw;
CREATE SERVER www_fdw_server_google_search FOREIGN DATA WRAPPER
www_fdw
  OPTIONS (uri 'https://ajax.googleapis.com/ajax/services/search/web?v=1.0');
CREATE USER MAPPING FOR current_user SERVER www_fdw_server_google_search;
CREATE FOREIGN TABLE www_fdw_google_search (
  /* parameters used in request */
  q text,

  /* fields in response */
  GsearchResultClass text,
  unescapedUrl text,
  url text,
  visibleUrl text,
  cacheUrl text,
  title text,
  titleNoFormatting text,
  content text
) SERVER www_fdw_server_google_search;
```

```
postgres=# select url,substring(title,1,25)||'...'::substring(content,1,25)||'...'
          url          | ?column? | ?column?
```

```
http://www.postgresql.org/ | <b>PostgreSQL</b>: The wo... | Sophis-
ticated open-source...
http://www.postgresql.org/download/ | <b>PostgreSQL</b>: Downlo... |
The core of the <b>Postgre...
http://www.postgresql.org/docs/ | <b>PostgreSQL</b>: Docume... |
There is a wealth of <b>P...
http://en.wikipedia.org/wiki/PostgreSQL | <b>PostgreSQL</b> - Wikip... |
<b>PostgreSQL</b>, often ...
```

در ۲۰۰۳ مفهومی به SQL اضافه شد بام نام SQL/MED که هدف آن ارتباط پایگاه داده‌ها با خارج از دنیای خود بود. این مفهوم در PostgreSQL با نام FDW مخفف Foreign data wrappers است معرفی شد. FDW تقریباً امکان ارتباط با هر Wrapper خارجی را می‌دهد از DBMS های مختلف گرفته تا NoSQL ها و حتی فایل های CSV و git و ... یک مثال ساده

– load extension first time after install

```
CREATE EXTENSION mongo_fdw;
```

– create server object

```
CREATE SERVER mongo_server FOREIGN DATA WRAPPER mongo_fdw
```

```
OPTIONS (address '127.0.0.1', port '27017');
```

– create foreign table

```
CREATE FOREIGN TABLE customer_reviews
```

```
(
  _id NAME,
  customer_id TEXT,
  review_date TIMESTAMP,
  review_rating INTEGER,
  product_id CHAR(10),
  product_title TEXT,
  product_group TEXT,
  product_category TEXT,
  similar_product_ids CHAR(10)[]
)
```

```
SERVER mongo_server
```

```
OPTIONS (database 'test', collection 'customer_reviews');
```

– collect data distribution statistics

```
ANALYZE customer_reviews;
```

در این مثال یک جدول با نام customer_reviews ایجاد می‌شود که اطلاعات خود را از سرور mongo دریافت می‌کند.

این جدول قابلیت انجام تراکنش بین سرورها، Sort، Join و حتی Aggregate را داراست. البته توقع کارایی مناسب در همه وارپرها را نداشته باشید. بسته به وارپری که استفاده می‌شود و روش‌های ارتباطی با مقصد می‌تواند کارایی متفاوتی را داشته باشد.

Wrapper های بسیار زیادی برای PostgreSQL نوشته شده است. از وبسرویس‌های SOAP گرفته تا git و توپتر و ... اما اگر شما نیاز به نوشتن یک Wrapper جدید داشته باشید باید دانش نوشتن اکستنشن به زبان C را داشته باشید و صد البته



۱۹۶۰:

اولین مدل‌های کامپیوتری
شده پایگاه داده

۱۹۷۰:

مدل رابطه‌ای بعنوان برنده‌ی اصلی در بین
مدل‌های دیگر اعلام می‌شود
عصر طلوع پایگاه داده
- مدل رابطه‌ای و زبان برنامه نویسی آن یعنی
SQL ادغام میشوند
- مدل مخرب باعث نابودی مدل‌های دیگر
می‌شود

۱۹۸۰:

موفقیت‌های تجاری پایگاه داده‌ی رابطه‌ای
یک توسعه‌ی تجاری
- SQL به یک استاندارد بالفعل بدل می‌شود
- پیشنهاد‌های تجاری از طرف IBM، رشد بازار اوراکل
- ورود مدل‌های دیگر به صحنه‌ی بازار، البته با جلب
توجه نه چندان زیاد

۱۹۹۰:

پایگاه داده‌ی رابطه‌ای غرق می‌شود
تغییرات در تکنولوژی
- انفجار داده‌ها با آمدن عصر اینترنت
- پایگاه‌های داده‌ی SQL که با یک سرور کار می‌کنند، با
مشکل منابع مواجه می‌شوند
- حوزه‌های هوش تجاری و تجزیه و تحلیل داده، از پایگاه
داده‌های معاملاتی خارج می‌شوند

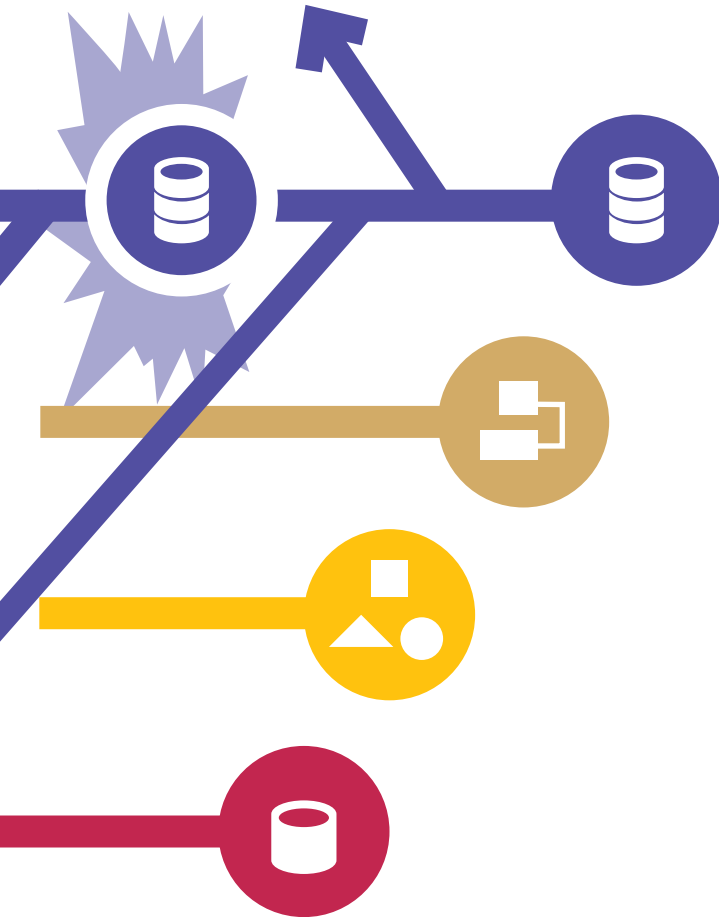


مدل سلسله‌مراتبی (IMS)



مدل شبکه (CODASYL)

۱۹۷۰: EF CODD مقاله‌ای را درباره‌ی
مدل رابطه‌ای منتشر می‌کند



۱۹۹۰:

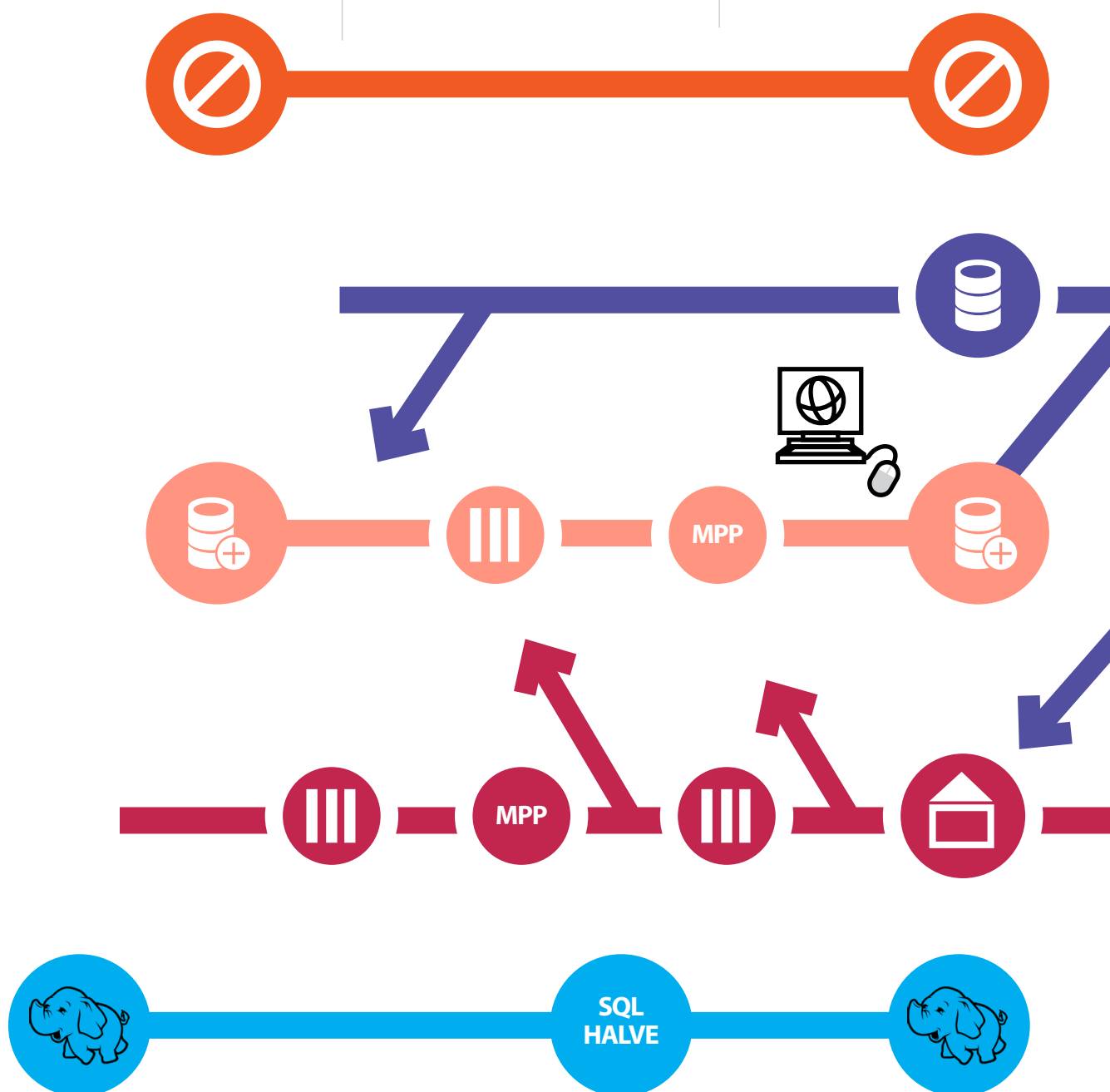
SQL توزیع شده و NOSQL ادغام می شوند
بازیگران جدید به هم ملحق می شوند
- پایگاه های داده ی جدید معاملاتی SQL معرفی شده اند
- پایگاه داده های NOSQL خلاء موجود در پردازش داده های سازمان نیافته را پر می کنند
- Hadoop با تجزیه و تحلیل پتابایت های(Petabyte: معادل یک میلیون گیگا بایت) داده توجه ها را جلب می کند

امروز:

پایگاه های داده ی Scale-out کمی جلب توجه می کنند
پایگاه های داده با علم روز منطبق میشوند و رشد می کنند
- کسب و کارها خواهان تجزیه و تحلیل های لحظه ای بر روی داده های عملیاتی هستند
- Scale-up SQL ثابت می کند که پر هزینه است، اما Scale-out SQL مشکلات محدودیت منابع را رفع میکند
- Scale-out تجزیه و تحلیل های آنی را همراه با مقادیر بالای نقل و انتقالات فراهم می کند
- Clustrix و Google در این فضا پیشگام هستند

آینده:

میراث Scale-up جایگزین شده است
کسب و کارها با نوآوری ها در حوزه ی پایگاه داده پیشرفت می کنند
- Single Node SQL جای خود را به Scale-out SQL میدهد
- تجزیه و تحلیل های انواع داده های انبار شده در پایگاه داده های آنی قابل دسترس می شوند
- کسب و کارها تحرک و چابکی فراوانی میگیرند





```

    read/2,
    remove/2,
    query_fetch/1,
    query_filtermap/2,
    query_foreach/2,
    query_raw/3
  ]).

include("tnesia.hrl").
include("types.hrl").

%% Main API

%% write

-spec write(tnesia_timeline(), tnesia_record()) ->
    {tnesia_timeline(), tnesia_timepoint()}.

write(Timeline, Record) ->
    Timepoint = tnesia_lib:write(
        #tnesia_input{
            timeline = Timeline,
            timepoint = now(),
            record = Record}),
    {Timeline, Timepoint}.

tnesia_record().

```

```

%% insert
%%
insert({{timeline,
    {keys, Keys},
    {values, Values}},
    pre_insert(Timeline,
        Timeline,
        Keys,
        {{list_values, Values},
        State) ->
    pre_insert(Timeline,
        Timeline,
        Keys,
        {{list_values, Values},
        State) ->
    do_insert(Timeline,
        Result = ?API:write(
            Timeline,
            Lists:zip(Keys, Values),
            do_insert(Timeline,
                d_insert(Timeline,
                    Lists:reverse(Results))
            )
        )
    )
%% delete
%%
delete({{timeline,
    {record_timepoint, Timepoint}},
    ?API:remove(Timeline,
        Timepoint)
    )
%% select

```

سرگذشت یک پروژه متن باز

چرایی، چیستی و چگونگی بانک اطلاعاتی Tnesia

در ثانیه برسد. با این فرض نیاز به یک بانک اطلاعاتی مناسب کاملاً احساس شد. اولین گزینه‌هایی که روی میز گذاشته شد بودند که همه آن‌ها خوشبختانه متن باز بودند.

بانک اطلاعاتی PostgreSQL با توجه به رابطه ای بودنش برای نگهداری داده های زماندار بسیار مناسب بود، اما برای توزیع پذیر کردن آن با هدف داشتن یک سیستم Highly Avail-able چالش های زیادی پیش رو داشتیم که باعث شد در همان ابتدای امر از آن صرف نظر کنیم.

بانک اطلاعاتی InfluxDB هم از این جهت که

بیسفون بودند محتوا تولید کنند و با آن‌ها در تماس باشند.

در مراحل طراحی اولیه همه چیز خوب پیش رفت تا آنجایی که نوبت به انتخاب بانک اطلاعاتی برای نگهداری محتوای کانال‌ها شد. ویژگی بارز این داده‌ها زماندار بودن آن‌ها

بود، به صورتی که کاربرانی که آن‌ها را دنبال می‌کنند هر بار که آنلاین می‌شوند بتوانند از آخرین زمانی که آنلاین بوده اند داده‌های جدید را بازیابی کنند. در برآوردهای اولیه فهمیدیم که با توجه به تعداد کاربران جاری و تعداد کانال‌های پیش‌بنی شده، در ساعات پیک شاید تعداد درخواست‌های بازیابی محتوای کانال‌ها به چند میلیون درخواست

در این مقاله به بررسی بانک اطلاعاتی Tnesia می‌پردازیم و به این سوالات پاسخ می‌دهیم: Tnesia چیست، چرا به وجود آمده و چگونه میتوان از آن استفاده کرد.

چرایی

پاییز سال ۹۳ وقتی قرار شد ویژگی جدیدی به نام کانال به نرم افزار بیسفون اضافه کنیم، من به عنوان توسعه دهنده سمت سرور مسئول طراحی و پیاده سازی موضوع شدم. کانال ویژگی ای بود که موسسات، نشریات و در کل تولید کنندگان محتوا میتوانند از طریق آن برای مخاطبان‌شان که کاربر



حمیدرضا عیسی

ویژگی بارز این

داده‌ها زماندار

بودن آن‌ها بود،

به صورتی که

کاربرانی که

آن‌ها را دنبال

می‌کنند هر

بار که آنلاین

می‌شوند



open source

Dependency در نرم افزاری که با Erlang نوشته شده است. در این راه شما میتوانید با استفاده از API فراهم شده به زبان Er-lang داده های خود را با سرعت بالایی ذخیره و بازیابی کنید. راه دوم نصب Tnesia به صورت Stand-Alone است که در آن نیازی به استفاده از زبان Erlang بر ذخیره و بازیابی اطلاعات ندارید و میتوانید با استفاده از زبان جستجوگر TQL بر روی HTTP به بانک اطلاعاتی خود دسترسی داشته باشید. البته راه دوم سرعت کمتری نسبت به راه اول دارد به خاطر استفاده از یک زبان جستجوگر و همچنین هزینه های پروتکل HTTP. توضیحات بیشتر راجع به چگونگی استفاده از Tnesia را می توانید در صفحه گیت هاب آن به آدرس <https://github.com/bisphone/Tnesia> مطالعه کنید

جمع بندی

بررسی سرگذشت پروژه های متن باز بیان گر این موضوع است که حتی پروژه های متن باز بزرگی که ما همه آن ها را می شناسیم و از آن ها استفاده می کنیم از پروژه های کوچکی شروع شده اند که هیچوقت تصور آن را هم نمی کردند که روزی به این شهرت و کاربرد برسند. از این رو شاید پروژه های کوچک من و شما روزی به چنین مرحله برسند، تنها اگر برای توسعه آن وقت بگذرایم و به فکر انتشار آن ها به صورت متن باز بیفتیم. ■

مراحل طراحی و پیاده سازی Tnesia شروع شد. پس از آزمایش های اولیه متوجه شدم که نه تنها میتوان از آن برای نیاز خودمان استفاده کنیم، بلکه انتشار آن به صورت متن باز باعث بالا رفتن انگیزه توسعه آن خواهد شد زیرا میتوان از حمایت جامعه متن باز به صورت های مختلف استفاده کرد و همچنین تبلیغی موثر نیز برای مجموعه خواهد شد. در نهایت موافقت شرکت با انتشار آن به صورت متن باز باعث این مهم شد.

چیزی

در ابتدا Tnesia تنها یک الگوریتم Time-Series Indexing بر روی بانک اطلاعات Mnesia بود، اما پس از گذشت زمان و با اضافه کردن ویژگی های مختلف به آن دیگر میتوان به آن یک بانک اطلاعاتی گفت. مثل هر بانک اطلاعاتی دیگری، هدف Tnesia ذخیره اطلاعات است، اما با تمرکز بر روی داده های زماندار. داده های زماندار، داده هایی هستند که بر اساس زمان رخدادشان ذخیره و بازیابی میشوند. این داده ها میتوانند در فواصل زمانی معین باشند مانند داده هایی که سنسورهای هواشناسی ذخیره میکنند، و یا با فواصل زمانی متغییر ذخیره شوند مانند لاگ های یک وب سرور و یا حتی توییت های شما در توییتر.

با همین هدف، Tnesia با معرفی سه مفهوم Timestep، Timeline و Timepoint داده های زماندار را دسته بندی میکند. به این صورت که کاربر آن میتواند با به وجود آوردن Timeline های متفاوت هر داده خود را با یک Timepoint که همان زمان جاری است در Timeline مورد نظر ذخیره کند. این Timepoint ها در Timestep هایی با فواصل زمانی یکسان ذخیره می شوند. وجود Timestep کمک میکند که در هنگام بازیابی داده ها بین دو فاصله زمانی، به جای جستجو تمام Timeline تنها قسمت هایی که احتمال وجود داده مورد نظر وجود دارد بررسی شود. همچنین با پیش بینی میزان تراکم داده و تغییر Timestep در تنظیمات Tnesia میتوان سرعت بازیابی داده ها را حتی بهبود بخشید.

چگونگی

برای استفاده از Tnesia دو راه وجود دارد. راه اول استفاده آن به عنوان یک

به صورت کلی مخصوص نگهداری داده های زماندار بود گزینه مناسبی بود، از طرف دیگر با دارا بودن ویژگی توزیع پذیری به عنوان یکی از خصوصیات اصلی، نگرانی ما را نسبت به داشتن یک بانک اطلاعاتی Highly Available از بین میبرد. اما چیزی که باعث شد در نهایت از آن استفاده نکنیم API آن بود که بر روی HTTP پیاده سازی شده بود. از آنجایی که از طرفی HTTP یک Text Protocol است و در مقایسه با Binary Protocol ها هزینه زیادی صرف کردن آن میشود و همچنین حجم داده ها را هم بالا میبرد، و از طرف دیگر نیازمند برقراری یک کانکشن جدید به ازای هر درخواست است، تصمیم گرفتیم به سراغ گزینه بعدی برویم.

بانک اطلاعاتی Riak هم با دارا بودن ویژگی های مناسب توزیع پذیری مانند Highly Available، Partition Tolerancy و Eventual Consistency گزینه بسیار مناسبی بود اما از آنجایی که درخشش این ویژگی ها زمانی میسر بود که از آن به عنوان یک بانک اطلاعاتی Key/Value استفاده شود، مجبور به حذف این گزینه هم شدیم.

در نهایت بانک اطلاعاتی Cassandra را انتخاب کردیم که از لحاظ توزیع پذیری تمام ویژگی های Riak را دارا بود به علاوه ساختار Column Family آن که کاملاً مناسب بود برای مدل کردن داده های زماندار. از طرف دیگر با داشتن یک زبان جستجوگر به نام CQL که شباهت هایی به SQL داشت و همچنین به صورت Binary هم بود در رقابت با InfluxDB برنده میشد.

این انتخاب زمانی متزلزل شد که متوجه شدیم Client Driver های موجود Cassandra برای زبان Erlang که زبان اصلی ما برای توسعه نرم افزار بود قابل اطمینان نیستند، زیرا یا به روز نبودند و یا تمام ویژگی های لازم را نداشتند و از همه بدتر هر کدام تنها یک توسعه دهنده داشتند که اگر متأسفانه بر اثر تصادف رانندگی فوت میکرد، ما به مشکل جدی ای برمیخوردیم.

اینجا بود که تصمیم گرفتیم این مشکل را به نحو دیگری حل کنیم و پیشنهاد پیاده سازی یک بانک اطلاعاتی کوچک با حداقل امکاناتی که به آن نیاز داریم را به مدیریت مجموعه دادم. خوشبختانه با موضوع موافقت شد و



بررسی

سرگذشت پروژه

های متن باز بیان

گر این موضوع

است که حتی

پروژه های متن

باز بزرگی که

ما همه آن ها

رامیشناسیم

و از آن ها

استفاده میکنیم

از پروژه های

کوچکی شروع

شده اند که

هیچوقت تصور

آن را هم

نمیکردند که

روزی به این

شهرت و کاربرد

برسند



????? ???? ???? ????
 AES-128 ??????
 ?????? ?????? ???? ????
 ???? ???? ???? ?

نکاتی که در فارسی باید رعایت کنیم

ویژگی فارسی پایگاه داده‌ها، جداول و فیلدها
 گاهی هنگام ذخیره‌سازی، حروف در جداول به صورت ناخوانا ذخیره می‌شوند حتی با این که نکته اول را رعایت کردید. گاهی هم هنگام مرتب کردن، برخی رکوردها به انتهای جدول می‌رود. چرا؟

علت آن است که یکی از خصیصه‌های پایگاه داده‌ها یا جداول یا فیلدها را بر روی «utf8-persian-ci» تنظیم نکرده‌اید و عباراتی که با «ک»، «گ» و «ژ» شروع می‌شود را در انتها یا ابتدا می‌آورد. شماره یونیکد این حروف بعد از حروف عربی است.

ذخیره تاریخ

برای ذخیره‌ی تاریخ (به خصوص تاریخ خورشیدی) در پایگاه داده‌ها روش‌های مختلفی وجود دارد.

ذخیره‌ی تاریخ به صورت timestamp: در این روش یک تاریخ خاص (معمولاً unix time، یک ژانویه ۱۹۷۰) را به عنوان مبدا در نظر گرفته و تعداد ثانیه‌های سپری شده تا تاریخ مورد نظر را ذخیره می‌کنند. (در این روش عدد به دست آمده برای تاریخ شمسی و میلادی یکسان بوده) استفاده از نوع داده‌ی datetime: در این روش به سادگی می‌توان تاریخ مورد نظر را با این نوع داده ذخیره کرد. البته برای تاریخ شمسی ابتدا باید تاریخ را به میلادی تبدیل کرده و سپس آن را با این نوع داده ذخیره کنید. در نظر داشته باشید که گاهی نوع داده‌ی datetimeoffset همیشه یک منطقه زمانی را نیز با خود حمل می‌کند می‌تواند بسیار مفید واقع شود. ذخیره‌ی تاریخ به صورت رشته‌ای: در این روش تاریخ را به صورت یک رشته ذخیره می‌کنند. برای مثال '01.04.2015'. در این روش همیشه اعداد روز و ماه را به صورت ده‌تایی ذخیره کنید تا در سرعت مرتب‌سازی و جستجو بهینه‌تر عمل کنید. اما توصیه می‌شود هرگز تاریخ (شمسی) را به صورت متن ذخیره نکنید تا بتوانید از توابع بهترین استفاده‌ها را داشته باشید. استثنائاً انگلیسی را پاس بداریم در نهایت هم توصیه می‌شود که در نام‌گذاری پایگاه داده‌ها، جداول و فیلدها از فینگلیش استفاده نکنید، مثل پسر/دختر خوب معادل انگلیسی آن را بیاورید و تا حد امکان کوتاه و در عین حال با معنا باشد. باور کنید انتخاب نام درست خودش هنر/البته می‌توانید فارسی هم استفاده کنید اما با توجه به احتمال بروز مشکلات بعدی، بهتر است فراموشش کنید. اما اگر اصرار دارید، باید هنگام صدا زدن نام را در میان نویسه «» قرار دهید، همانند زمان استفاده از اسمی، دارای فاصله بین کلمات. ■

هنگام کار با پایگاه داده‌های مختلف، معمولاً در ابتدای آشنایی برای ذخیره و بازیابی داده‌های فارسی، با مشکلات عجیب و قریبی مواجه می‌شویم. در این مقاله چند نکته ساده را برای رفع چنین مشکلاتی عنوان می‌کنیم.

ارتباط اولیه

حتماً یادتان هست اولین روزهای برنامه‌نویسی را که قصد اتصال به پایگاه داده‌ها را داشتید. گاهی اطلاعات به صورت علامت سوال ذخیره می‌شود یا حتی اطلاعاتی که به درستی ذخیره شده، به صورت علامت سوال یا نویسه‌های ناخوانا در می‌آید. ظاهراً همه چیز خوب درست انجام شده، پس مشکل کجاست؟
 این مساله جزو اولین مواردی است که باید در نظر داشته باشید و به سادگی نیز برطرف می‌شود؛ تنها با فراخوانی دستور زیر بعد از برقراری اتصال:

```
set names 'utf8'
```

یکسان‌سازی

بارها اتفاق افتاده است، هنگامی که الکی مثلاً جستجویی در داده‌ها انجام می‌دهیم، نتیجه‌ای را بر نمی‌گردانند. به پایگاه داده‌ها می‌گوییم: «خودم فلان اطلاعات را وارد کردم، چرا پیداش نمی‌کنی؟» گیر کار کجاست؟
 البته خوشبختانه ما آزادکاران که از مزایای گنو/لینوکس بهره می‌جوییم، چنین گیرهای مسخره‌ای را نداریم و این نانی است که مایکروسافت و به طبع اون شرکت‌های توسعه‌دهنده محصولات مایکروسافتی در دامان ما (یا آنها) گذاشتند. در گنو/لینوکس صفحه کلید استاندارد فارسی فعال است و به همین خاطر معمولاً همه چیز درست کار می‌کند. اما در زمان‌های قدیم که مایکروسافت فارسی را پشتیبانی نمی‌کرد، مجبور بودیم که از صفحه کلید عربی استفاده کنیم و تقریباً این عادت سالیان سال است که همراه کاربران ایران شده و به خصوص بخش دولتی، هنوز از این عادت بدون این که خیلی آگاه باشند، رنج می‌برد.
 ما در فارسی حرف «ی» یا «ک» نداریم اما چون هنوز برخی صفحه کلیدها این حروف را به صورت عربی وارد می‌کنند، ممکن است به دنبال «یک» باشید و غافل از این که در پایگاه داده‌ها «یک» یا «ک» با «ی» عربی یا حتی «یک» ذخیره شده باشد. پس باید هنگام ذخیره یا واکشی داده‌ها، همواره یک منش را پی بگیریم که قطعاً نویسه‌های فارسی در اولویت است.



مهدی خوشروی
 نویسنده



سه محدودیت NoSQL



نوین کریمیان

❶ عدم پشتیبانی تجزیه و تحلیل «عمیق» تطبیقی

پایگاه داده‌های مبتنی بر هادوپ، برای دست یافتن به کارایی بالا، از سربر نرم‌افزاری معنانشناسی داده‌ها، به شکل پیچیده جلوگیری می‌کنند. این نوع معنانشناسی به کاربر نهایی این امکان را می‌دهد تا «what-if» را با استفاده از تجزیه و تحلیل کردن چند بعد از ویژگی‌های داده به وسیله روشی انعطاف‌پذیر انجام دهند. راه‌های فراوانی برای قطعه کردن داده وجود دارد و یا این که داده‌ها در نوع و معنا نسبتاً سریع در حال تغییر هستند. انجام دادن چنین «پرسشی (query)» در لحظه ممکن است ترجیح داده شود، اما وقتی دانشمندان داده‌ها می‌خواهند تا با بررسی عمقی، بیشتر در مورد چیزی که اتفاق می‌افتد بدانند، SQL و رابطه پیروز می‌شود.

❷ عدم داشتن ویژگی دسترسی‌پذیری بالا برای ابرهای خصوصی و ترکیبی

این موضوع تقریباً زمانی که NoSQL تازه ظهور کرده بود دیده می‌شد، اما محدودیت بزرگی است. در حالی که IBM، EMC، Oracle یا یک SAP ممکن است بسیاری از ویژگی‌های failover، remote-copy و roll-back/roll-forward را ارائه کنند، این ویژگی‌ها در NoSQL‌ها تقریباً نادر است. علاوه بر این پایگاه داده‌های SQL سال‌های بسیار متمادی زمان داشته‌اند تا بهتر و بهتر راهی را برای بدست آوردن «دسترسی‌پذیری بالا» در محیط‌های مقیاس بزرگ و مقیاس متوسط، مشخص کنند. اما راه کارهای NoSQL در ابرهای عمومی مقیاس بزرگ انبوه به دنیا آمده‌اند و اکنون باید خود را با ابرهای خصوصی تطبیق بدهد.

SQL یا NoSQL یا چیزی دیگر

یکی از سخنرانان نشست «In-Memory Computing Summit» که در سانفرانسیسکو برگزار شد و در آن هر دو نوع پایگاه داده‌های SQL و NoSQL به خوبی معرفی شدند، گفت: آن‌ها یک جایابی را در کلاینت‌هایشان از SQL به NoSQL دیده‌اند و اکنون در حال مشاهده بازگشت دوباره به سمت SQL هستند. معنی این اتفاق به گوشه رانده شدن NoSQL نیست. اما می‌توان حدس زد که NoSQL بیشتر و بیشتر بر روی پردازش «عملیاتی» داده تمرکز کرده است در حالی که پایگاه داده‌های SQL به نمایش قدرتش در تجزیه و تحلیل ادامه می‌دهند.

بنابراین محدودیت‌های NoSQL نشان‌دهنده یک تقسیم کار است که باعث ایجاد یک حس بزرگ در بسیاری از کاربردها با حتی یک معماری پایگاه داده‌های ابری وسیع یا سازمانی وسیع می‌شود.

محدودیت‌های NoSQL به معنای فقدان سودمندی به شکل کلی نیست و آن‌ها از پس وظایف بر می‌آیند. با این حال حدس زده می‌شود کاربران زرنگ هواخواهی خودشان را برای هم نوعان هادوپ محدود خواهند کرد. ■

❸ برای سال‌ها جایگزین‌های SQL مبتنی بر هادوپ مورد توجه کاربران بود، این جایگزین‌ها ارزان و متن‌باز بودند و به خوبی در محیط ابرهای عمومی تجربه شده بودند. در بسیاری از موارد این پایگاه داده‌ها قابلیت مقیاس‌پذیری برای حجم گسترده‌ای از «بزرگ داده» را در مقابل انواع رابطه‌ای عظیم الجثه داشتند. البته باید در نظر داشت که پردازش‌کننده پایگاه داده‌های NoSQL برای بدست آوردن این پیروزی، یک رویکرد متفاوت را نسبت به انواع رابطه‌ای/SQL در مدیریت داده‌ها، انتخاب کردند. رویکردی که به طور ویژه آن‌ها را در استفاده، نسبت به پایگاه داده‌های SQL بسیار محدودتر می‌کرد.

❹ سه مورد از این محدودیت‌ها عبارتند از:

❶ عدم انسجام داده مکرر

❷ عدم پشتیبانی تجزیه و تحلیل «عمیق» تطبیقی

❸ عدم داشتن ویژگی دسترسی‌پذیری بالا برای ابرهای خصوصی و ترکیبی

❹ عدم انسجام داده مکرر

هادوپ، مپ‌ردیوس، مانگو دی‌بی، هاپو و غیره طراحی شده‌اند تا فراتر از پایگاه‌های داده SQL در ابرهای عمومی با ورودی بسیار سریع از داده، قابل مقیاس‌پذیری باشند. برای این هدف به آن‌ها امکان «سازگاری شرطی» در هر عملیات «commit» اجازه داده شده است که باعث اطمینان یافتن از سازگاری در زمان انجام عملیات با تاخیر می‌شود. مزیت این روش جلوگیری از نگه داشتن تراکنش‌ها (transaction) است. البته در طول این تاخیر، برخی داده‌ها نادرست خواهند بود که کاربر این مساله را به عنوان هزینه پرداختی برای بینش اضافی که از داده‌ها پیدا می‌کند، می‌پذیرد.

اخیراً سیستم‌های برپایه هادوپ شروع به اداره کردن عملیات به‌روزرسانی سریع در ابرها کرده‌اند. در بیشتر مواقع سیستم‌های برپایه هادوپ، داده‌ها را قبل از آن که بر روی دیسک نوشته شوند، آماده می‌کنند. در نتیجه اگر سیستم پیش از نوشتن داده‌ها بر روی دیسک «کُش» کند، داده‌ها از بین می‌روند. پس از این اتفاق است که تجزیه و تحلیل‌ها با یک مجموعه از داده‌ها روبرو می‌شوند که با داده‌های خوانده شده اصلی متفاوت‌اند.

در این سال‌ها پایگاه داده‌های مبتنی بر هادوپ، مانند مانگو دی‌بی، درصدی از داده‌ها را که سازگارند، بهبود بخشیده‌اند. اما هنوز هم به خوبی پایگاه داده‌های رابطه‌ای نیستند.

علاوه بر این، پایگاه داده‌های مبتنی بر هادوپ شروع به ارائه کردن «دسترسی‌پذیری بالا» عملیات‌ها کرده‌اند که باعث اطمینان از نوشتن بر روی دیسک پیش از خواندن می‌شود. البته در بسیاری از موارد هنوز هم به خوانده شدن قبل از نوشته شدن، اجازه می‌دهند.



ما شاپرک و بانکداری اینترنتی بانک ملت و همراه بانک ملت است. جمعا با سرورهای پشتیبان ۳۵۰ تا سرور داریم.

❖ روی هر سرور دیتابیس جدا دارید؟

نه. ما این جا چند شغله ایم. ادمین لینوکس و محصولات آی بی ام هستیم این بخش عمده کار ماست و پایگاه داده سهم کمتری در شغل ما دارد. تقریباً ۴۰ - ۵۰ سرور دیتابیس داریم و بقیه اش وب اپلیکیشن و وب سرور یا لینوکس ادمین انجام می دهد که در لایه بیزنس لاجیک یا لایه اپلیکیشن است. یک اپلیکیشن جاوایی روی ساختار آی بی ام نصب می شود. این ها به پایگاه داده های اصلی گُر بانکینگ ما وصل می شوند که آی بی امی هستند. یکی از معاونت های شرکت DBA است.

در بخش عملیات یکسری پایگاه داده های لوکال است که ارزش نداشته این ها را برایش مین فریم هزینه کنند و دیتابیس های کوچکتر و میانی که DB2 تحت PC هستند. ارزش و اهمیتشان خیلی فاصله اش کم است با آن چیزی که روی مین فریم است. چون در داده کاوی و وب کاوی به آن ها احتیاج داریم و مراجع قانونی به آن نیاز دارند. ما چندتا مین فریم داریم مثل سامانه مکفا. پنج سال پیش یک مین فریم را کلا خالی کردیم و DBA هم شدید قبلش این قدر جدی نبود MySQL و SQL server داشتیم.

عمده سیستم های

غیر حضوری

پیش ما شاپرک

و بانکداری

اینترنتی بانک

ملت و همراه

بانک ملت

است. جمعا

با سرورهای

پشتیبان ۳۵۰ تا

سرور داریم.

به بهانه بررسی یکی از بزرگ ترین پایگاه داده ها، گلوگاهی کشف شد

درد طراحی

رضا فیوجی از سال ۸۳ در تیم مهاجرت داده پردازی بوده و بعد از آن به قول خودش با همان تفکر آی بی امی به بهسازان ملت آمد. اصلی ترین مشتری های بهسازان، بانک ملت و وزارت بازرگانی است که مدتی می شود در حال راه اندازی سامانه تدارکات برای الکترونیکی کردن خریدهای دولت است. در چارت سازمانی این شرکت، زیر مجموعه مدیر عامل چندین معاونت وجود دارد و بخش عملیات شامل زیر بخش هایی مانند پشتیبانی، بهینه سازی، ادمین حضوری و ادمین غیر حضوری و... است.

کار می کند. یک سایت ساده نیست کلی هزینه کردیم روی محصولات IBM سرورهای لینوکسی نصب کردیم. پروژه های تحت جاوا روی این ها سوار می شود.

❖ چقدر پیش می آید این تغییر نسخه اتفاق بیفتد؟

این یکی از تغییرات ماست تغییرات دیگری مثل به روز رسانی و تغییر گذرواژه و به روز رسانی سخت افزار هم هست. یکسری تغییرات هم ناخواسته و اجباری است مثل مشکلات دیتاسنتر و از کار افتادن شبکه بانک. کار که بزرگ می شود مشکلات هم تصاعدی زیاد می شود. عمده سیستم های غیر حضوری پیش

❖ با توجه به این که دیشب تا دیر وقت بیدار بودید و در شرکت کار می کردید معمولاً چه کارهایی را در نیمه شب انجام می دهید؟

کارهای پر ریسکی که سیستم را با اختلال مواجه می کند. مثلاً وقتی می خواهیم نسخه را عوض کنیم سیستم داون می شود و باید از آن یک پشتیبان داشته باشید و روال انتقال سرور پشتیبان را رعایت بکنید. یعنی به درستی کابل عوض کنید و روی آن همه چیز شبیه این طرف باشد که کمترین زمان عدم پاسخگویی را داشته باشیم بعد نسخه را نصب کنیم و تستش کنیم و دوباره در محیط عملیاتی تست می شود تا مطمئن شویم برای کاربر واقعی هم درست

3 این جا از چه نرم افزارها و ابزارهای آزادی استفاده می کنید؟

MySQL یک دانه بیشتر نداریم. ما نرم افزار متن باز کم داریم همان سوزه اینترنت پر از هم که استفاده می کنیم متن باز نیست چون مشتری ما را مجبور کرده است که از چه سیستمی استفاده کنیم. شرکت مکفا که به همراه بانک ما کمک می کند می گوید من یک MySQL آن طرف دارم تو هم باید یک MySQL داشته باشی و گزینه استاندارد این جا لینوکس برای سیستم عامل است، WebSDL آی بی ام برای وب سرور است و DB2 برای پایگاه داده است. تنها ابزار آزادی که من استفاده می کنم nagios است که یک مانیتورینگ جامع برای همه ابزارهای تحت لینوکس و شبکه است. تقریباً ۲۵ آجکت را من الان دارم مانیتور می کنم. رم، اکسپشن، هاردها، CPU و تعداد کاربرها را روی سرورها مانیتور می کنیم و واقعا عالی است. حتی تنظیم کردیم در چه مواقعی برای ما پیامک بیاید. خوبی اش این است که می توانیم برایش پلاگین هم بنویسیم.

4 مشکلاتی که برای پایگاه داده ها پیش می آید و کندی آن ها معمولا به چه دلیل است؟

طراحی یکی از مشکلات عمده است و چیزی است که اصلا دیده نمی شود به نظر من طراحی هکر هستند. چون یک چیز غلط طراحی می کنند و فردا کندی دیتابیس و شبکه و اپلیکیشن رو اول مشتری می بیند بعد ادمین و تولید و بعد می گویند آقای طراح تو هم یک نگاهی بکن درحالی که هکر آی تی طراح ها هستند که با طراحی درست می توانند کاری کنند که هیچ کس ناراحتی نداشته باشد.

3 به نظر شما نگاهی که به استفاده از پایگاه های داده در ایران هست با کشورهای دیگر متفاوت است؟

آن چیزی که من شنیدم به لحاظ فنی هیچ کس به اندازه ایران متخصص نیست. چون ما شغل مشخصی نداریم همه مان همه کاره هستیم و مجبوریم به خاطر این که همه درست کار نمی کنند وارد کار همه بشویم و سر از خیلی چیزها در بیاوریم. من از کنترل پروژه گرفته تا دیتاستر را بلدم. در کشورهای صنعتی چون هر کس کار خودش را انجام

می دهد این طور نیست. به لحاظ تکنیکال ما از خیلی ها قوی تریم برای همین راحت اپلای می کنیم. تمام نمره منفی هایی که من به همکارانم می دهم بابت سواد نیست بابت عدم دقت و تعامل و هماهنگی است. چیزهایی که ما نمی دونیم طراحی درست و تشخیص و اندازه یک بیزینس است. اندازه کار را تشخیص نمی دهیم و می خواهیم این چیزی که بلد هستیم را ارائه دهیم و پولش را بگیریم. شرکت ما چون IBM بیس بوده و همه ما در IBM سابق و داده پردازی کار کردیم داریم IBM ای زندگی می کنیم. البته به درد بیزینس بانک می خورد اما خیلی جاها این طوری نیست

3 وضع DBA ها در ایران و جهان را چطور ارزیابی می کنید؟

در ایران بهتر است کسی که می خواهد در این زمینه کار کند یک بعدی نباشد. مثلا می روند اوراکل یاد می گیرند. در حالی که بسیاری با یاد گرفتن یک ابزار کار خود را تمام شده فرض می کنند. مثل این که فقط بگوئی من این کار را با نبردست باز می کنم. تو دیتابیس کار نشدی خیلی هنر کنی اوراکل کار شدی. در این زمینه اگر می خواهیم کارهای بشویم باید طراح دیتابیس شویم و اصول پایگاه داده را بدانیم. ابزار اصلا مهم نیست. بیزینس ایران را نگاه می کنی و آن ها را یاد می گیری. یاد گرفتن که کاری ندارد نوع استفاده از ابزارها و دیدت نسبت به کار مهم است که یک بحث جداست. من در مصاحبه هایم یک کلمه فنی از کسی سؤال نمی کنم. رزومه رو ببینم و تشخیص بدهم به درد ما می خورد فقط با او صحبت می کنم. هیچ سؤال فنی نمی پرسیم. ما ابزار کار نباید باشیم. یعنی با سوادترین فرد را می آوریم این جا احساس بی سواد می کند. برای همین سامانه بانکی شب عید می افتد. چرا؟ مگر مین فریم زد آی بی ام یا اوراکل نداریم؟ خود ما مشکل مان طراحی است.

3 وضعیت آموزش دیتابیس در ایران چطور است؟

ما کلا در ایران آموزش نداریم همه باید خودشان سعی کنند وقتی جایی می روند کار بگیرند. ما حتی در شرکت خودمان هم آموزش کاملی نداریم. چون روند استخدام

طول می کشد می گویم لینوکس را خودت بخوان چون کم ترین کار من لینوکس است. تقریباً سه ماه هیچ کاری با نفر جدید نداریم جز این که شیفت بیاید پیش بقیه باشد و کارهای دیگران را ببیند و یاد بگیرد و اصلا برایم مهم نیست بعد آزمایشی بخواهد با رزومه سه برابر از این جا برود. چون هیچ سودی برای من ندارد که کار را ذره ذره به او بدهم. اگر طرز فکرش این است که برود هیچ جایی کاری نمی تواند یاد بگیرد. هیچ بخل و خستی نباید در آموزش انجام دهی. شما می دانید شغل server administrator اصلا در سازمان فناوری اطلاعات تعریف نشده. تنها شغل administrator که تعریف شده ادمین شبکه است.

3 به اعتقاد شما نباید این ها را قبلش در دانشگاه بیاموزند؟

اینکه قطعاً درست است. اما خیلی شغل ها را نمی توانی؛ چون خاص هستند و ابزارها در شرکت به نوع خاصی استفاده می شود. یعنی من لینوکس را طور خاصی در این جا استفاده می کنم و بیزینس این جا را باید یاد کار آموزش بدهم. هر کسی که این جا می آید باید یکسری ابزار بلد باشد و سازمان را بشناسد و بداند از ابزارها در سازمان چطور استفاده کند. درسته که بیرون از این جا هیچ آموزش درست و هدف مندی وجود ندارد باید به کار آموزش فرصت دهید که خودش یاد بگیرد. شما بعد از دانشگاه تازه باید میلیونی هزینه کنی و یاد بگیری. کسی که آموزش می دهد هم طرف را تعصبی بار می آورد. فکر می کند اگر اوراکل بداند در دیتابیس سرهنگ تمام است. در حالی که اصلا دیتابیس کار نیست. هیچ جا نمی گویند مشکل آموزش ما تعصبی بار آوردن افراد است. حالا متن باز تفکر دموکراسی مآبانه ای دارد و دوستانه تر است و من از این دفاع می کنم ولی وقتی وارد جزئیات می شوی خیلی بد است بگویم فقط سوزه یا فقط DB2. تو فقط می توانی بگویی تجربیات در یک زمینه بیشتر است.

3 اگر بخواهیم دیتابیس های موجود را برای کارهای مختلف دسته بندی کنید چطور این کار را می کنید؟

قطعاً در فعالیت های بانکی که با تراکنش های مالی سر و کار دارند باید از ابزار مهمی مثل DB2 آی بی ام در بستر مین فریم استفاده کنیم. چون از اسکیل بزرگ می تواند پشتیبانی کند و تمام



مثلا می روند اوراکل یاد می گیرند. در حالی که بسیاری با یاد گرفتن یک ابزار کار خود را تمام شده فرض می کنند. مثل این که فقط بگوئی من این کار را با نبردست باز می کنم. تو دیتابیس کار نشدی خیلی هنر کنی اوراکل کار شدی. در این زمینه اگر می خواهیم کارهای بشویم باید طراح دیتابیس شویم و اصول پایگاه داده را بدانیم



آن کلمات انگلیسی که همه بلدیم، Availability و Reliability و غیره روی این بهتر است. هرچند که پشتیبانی بدی دارد و همش می‌خواهد شما را تشنه کند. البته مایکروسافت از تمام این‌ها بدتر است. اولاً که ما پشتیبانی نداریم ولی به روزرسانی‌ها منوط به یک فیکس لیستی است که هزارتا مشکل را برطرف می‌کند و هزارتا مشکل دیگر را به وجود می‌آورد. هرچند که ما بی‌سوادیم. شما پایداری سیستم فیس‌بوک را نگاه کنید که با چند میلیون مشتری آنلاین تبادل داده نامحدود و باورنکردنی می‌کند و شما کندی احساس نمی‌کنید و در بانکداری الکترونیک در پیک کار با ۲۰ هزار نفر که فقط سشن دارند یعنی فقط در سیستم هستند و لزوماً کلیک یا فعالیتی در سایت نمی‌کنند شب عید از کار می‌افتد. پس این که من می‌گویم بی‌سوادیم باور کنید راست می‌گویم. ما ساده‌ترین چیزها را در این جانی نمی‌توانیم نگه داریم.

تراکنش را می‌توانید به یک گله تشبیه کنید اگر یک سیستم چابک دارید با تراکنش‌های کوچک زیاد مثل دسته مورچه MySQL از تمام این‌ها بهتر است. اگر تراکنش‌هایی مثل فیل دارید اوراکل و SQL server بهتر است. اگر تراکنش‌های مالی با اهمیت و حجم داشتید DB2 و بعدش oracle جوابگوتر هستند. شاپرک ما الان روی DB2 است. یک ابزاری دارد که ۶ میلیون رکورد مالی را دو دقیقه‌ای لود می‌کند ولی اگر آن را با جاوا انجام دهی دو ساعت لود فایل طول می‌کشد. امنیت و مانیتورینگ خوبی دارد.

ابزارها و مفاهیم جدید مثل NoSQL چگونه؟

در این زمینه خیلی متخصص نیستیم اسمش را شنیده‌ام اما حتی یک پاراگراف درباره‌اش ننوخته‌ام. خیلی راه داریم تا به آنجا برسیم. همین الان در تشخیص این‌که از چه پایگاه‌داده‌ای استفاده کنیم گیر هستیم. بدون این‌که بدانیم الان کجا هستیم و در آینده به کجا می‌خواهیم برویم. این که بدون فکر از یک ابزار جدید استفاده کنیم اشتباه است. لزومی ندارد همه به روز باشیم ولی لازم است که چیزی که داریم را خوب نگه داریم و همراه با وندور باشیم.

وضعیت لایسنس نرم‌افزارهایتان چگونه است؟

لایسنس را به سختی از طریق شرکتی خریدیم

که یک نمایندگی در سوئد داشت. پشتیبانی نمی‌توانیم داشته باشیم. می‌توانیم از طریق سیستمی که آی‌پی‌اش دیده نمی‌شود یک پیج رایگان را دانلود کنیم. مشکل تحریم است. مین‌فریم بیست میلیاردی ممکن است ۴۰ میلیارد تمام شود اما برای بانک ارزش دارد.

مشکل شرکت‌های ایرانی در کارهای مرتبط با پایگاه داده چیست؟

شما باید ببینید از «الف» تا «ی» یک بیزینس را بررسی کنید ببینید کجایش مشکل دارد. تخصص که به اندازه کافی وجود دارد پس چرا سیستم درست کار نمی‌کند؟ اول به خاطر تعامل. دوم گفتن جمله «من مشکلی ندارم». سال ۸۶ من دنبال نیرو می‌گشتم و آقای باغومیان یکی از فعالان سایت تکنواکس را معرفی کرد. وقتی آمد و صحبت کردیم من حتی نگذاشتم فرم پر کند. وقتی به او گفتم اگر روزی بخواهی یک برنامه برای غیرلینوکس بنویسی چی. گفت من نمی‌توانم. این تعصب نباید وجود داشته باشد.

چرا بانک‌ها و ای‌تی‌ها هنوز روی ویندوز مانده‌اند؟

شاید علتش کم کار کردن آن‌هایی است که در بحث متن‌باز کار می‌کنند. این را رسانه‌ای مثل شما باید جا بیندازد. بانک‌ها باید به این سمت بروند. حداقل این که آزاد است و می‌توانی شخصی‌سازی‌اش کنی. نه دیگر پول آنتی‌ویروس می‌دهی نه از این بابت نگران هستی. من به عنوان سر دسته این قضیه حاضرم سه نصف شب با یک مدیر جلسه بگذارم و یک پرزنتیشن ارائه بدهم که نه با این دید که خواهش می‌کنم به سمت لینوکس بروید بلکه با این دید که احماق هستید اگر به این سمت نروید. یک نفر باید با قدرت جلو بیفتد و خسته نشود. من سال ۸۳ به همراه آقای تشکری و دوستان دیگر در تیم مهاجرت از ویندوز به لینوکس داده‌پردازی بودم. من آن زمان نماینده اداره کل اینترنت بودم. هدف این بود که تمام ۶۰ منشی، کارهای اداری را با نرم‌افزارهای متن‌باز انجام دهند. قرار شد این آفیس را آموزش دهیم. موفق نشدیم به خاطر دید مدیریت، چون حمایت نداشتیم. اولین هاستینگ لینوکس داده‌پردازی را من راه‌اندازی کردم و اولین مشتری‌ام هم داده‌پردازی اصفهان بود. مدیریت درست نیست. همان‌طور که بانک در فاکتور توسعه‌اش، دیتاستر نمی‌داند چیست

و جا کم می‌آورد. از اصول است که وقتی دیتاستر طراحی کردید اگر هیچ رشدی هم ندارید ۲۰ درصد بیشتر از چیزی که نیاز داری را در نظر بگیرید. ما سال ۸۵ از یک سرور شروع کردیم الان به ۳۵۰ سرور رسیده است. سالی دو نفر افزایش داشتیم ماهی دو سرور. ۵ سال پیش محاسبه کردم دیدم ما ۷ ترابایت نیاز داریم. درخواست ۱۰ ترابایتی دادم. این خریدش دو سال طول کشید. جایش یک ۴۰ ترابایتی دادند. هفته پیش ۳۰ ترابایت کم آمد. طرف صد برابر برای ویندوز و محصولات احماقانه‌اش می‌دهد که تحریم‌ها برداشته شود باید هزار برابر هزینه کند.

در بانک‌ها چیزی به نام تراکنش مالی داریم این آیا معادل همان تراکنش دیتابیس است؟

نه شما شاید در ۲ دیتابیس مختلف بابت یک کار کاربر سه رکورد ثبت کنید. برای مسائلی مانند وب‌کاوی، کشف تقلب، داده کاوی لاگ اتفاقات را هم ثبت می‌کنند. یعنی از چه مرورگری با چه آی‌پی‌ی چه کاری انجام داده است.

یعنی وقتی پیام می‌آید که تراکنش با موفقیت انجام شد این تراکنش مربوط به دیتابیس است یا چیزی غیر از دیتابیس هم این وسط وجود دارد؟

ممکن است به خاطر این که کاربر معطل ثبت در دیتابیس نشود یک روال آفلاین در بیاوریم که با یک تأخیر کوچکی در یک فایل ذخیره شود و از فایل به دیتابیس برود. در نهایت مرجع دیتابیس است. ولی لاگ مشتری‌ها در یک فایل ثبت می‌شود و با یک اپلیکیشن دیگری هر چند دقیقه یکبار به روزرسانی می‌شود.

وقتی که یک تراکنش ناموفقی داریم رول بک چرا این قدر طول می‌کشد؟

این سؤال اصلاً به جا نیست جایی که تراکنش مالی‌اش ۲۴ ساعت طول می‌کشد می‌خواهید رول‌بک‌اش ۴۸ ساعت طول نکشد؟ شاپرک در روز در ۷ سیکل پول را واریز می‌کند.

حجم و تعداد تراکنش‌های شما چقدر است؟

شاید در طول روز بانک ملت ۲۰ میلیون تراکنش مالی داشته باشد که از طریق شاپرک و پذیرنده‌ها انجام می‌شود. ■

شما پایداری سیستم فیس‌بوک را نگاه کنید که با چند میلیون مشتری آنلاین تبادل داده نامحدود و باورنکردنی می‌کند و شما کندی احساس نمی‌کنید و در بانکداری الکترونیک در پیک کار با ۲۰ هزار نفر که فقط سشن دارند یعنی فقط در سیستم هستند و لزوماً کلیک یا فعالیتی در سایت نمی‌کنند شب عید از کار می‌افتد



| تخصیصی |

| ۸۰ | نحوه احیای سرور

| ۸۲ | ایمکس جادوی گنو

| ۸۸ | ماجراهای سیس ادמיینی

| ۹۰ | واکنش به حملات سایبری مهم‌تر از جلوگیری است



#rm -rf /

در بدترین شرایط هم می شود از نو ساخت

نحوه احیای سرور

دیگر ls نداریم، ولی echo و file globs ها هنوز هستند. هش یا الیاس شده؟
با این ها چیکار می تونیم بکنیم؟

```
root@rmrf:/# type ls
```

```
ls is aliased to `ls -color=auto`
```

اوه دستور در واقع این می شود

```
ls-color=auto () { printf "%s\n" ${1:+${1%/}/*}; }
```

خوب می توانیم آن را از الیاس خارج کنیم.

```
root@rmrf:/# unalias ls
```

```
root@rmrf:/# ls() { printf "%s\n" ${1:+${1%/}/*}; }
```

```
root@rmrf:/# ls
```

```
/dev
```

```
/proc
```

```
/run
```

```
/sys
```

```
root@rmrf:/# ls /dev
```

```
/dev/pts
```

خوب کارمون را تا این جا می توانیم ذخیره کنیم.

```
root@rmrf:/# echo 'ls() { printf "%s\n" ${1:+${1%/}/*}; }
```

```
root@rmrf:/# echo *
```

```
dev proc run sys
```

```
# echo /dev/pts/*
```

```
/dev/pts/0 /dev/pts/3 /dev/pts/ptmx
```

آهان! ما /dev و /run و /proc و /sys را داریم. و حالا
که ls داریم، می تونیم کاری کنیم که خواناتر باشه.

```
root@rmrf:/# for file in /dev/pts/*; do echo $file;
```

```
done
```

```
/dev/pts/0
```

```
/dev/pts/3
```

```
/dev/pts/ptmx
```

و البته از آنجایی که printf هنوز هست و در بش
می شود تابع تعریف کرد، ما ابزاری مثلی ls می سازیم
اگر چه خیلی محدودتر است.

```
root@rmrf:/# ls() { printf "%s\n" ${1:+${1%/}/*}; }
```

```
-bash: syntax error near unexpected token `{'
```

چی؟ دستور کاملا درسته و مشکلی نداره. ببینیم

فقط برای تفریح، تصمیم گرفتم که یک
سرور لینوکس راه بندازم و دستور rm -rf / را
با کاربر ریشه (root) اجرا کنم و ببینم چه چیزهایی
باقی می ماند. فهمیدم که rm به خرابکارهایی مثل
من فکر کرده و برای انجام همچین کاری باید
no-preserve-root را اعمال کنیم.

```
# rm -rf --no-preserve-root /
```

بعد از اجرای این دستور مسخره، ابزارهای خوبی
مثل

```
/bin/ls
```

```
/bin/cat
```

```
/bin/chmod
```

```
/usr/bin/file
```

همگی از بین می رود! البته همچنان ارتباط جاری
SSH و بش و در نتیجه تمام ابزارهای توکار بش مثل
echo را در اختیار داریم.

```
root@rmrf:/# ls
```

```
-bash: /bin/ls: No such file or directory
```

```
echo; }' /proc/1136/exe
$ alias upload='{ xxd -p | encode | nc -q0 -lp 5050; }' /proc/1149/exe
$ upload < bin/busybox /proc/1179/exe
/proc/1215/exe
و در سیستم rmrf شده: /proc/1217/exe
/proc/1220/exe
# cd / /proc/1221/exe
# alias decode='while read -ru9 line; do printf /proc/1223/exe
"$line"; done' /proc/1248/exe
# alias download='( exec 9<>/dev/tcp/{IP OF NON /proc/1277/exe
HOSED BOX}/5050; decode )' /proc/1468/exe
# download > busybox /proc/1478/exe
/proc/1625/exe
الان می توانیم یک Object مشترک بسازیم که روی busybox مجوزها را تغییر می دهد: /proc/1644/exe
/proc/1/exe
$ cat > setx.c <<EOF /proc/374/exe
extern int chmod(const char *pathname, unsigned /proc/378/exe
int mode); /proc/471/exe
/proc/616/exe
/proc/657/exe
int entry(void) { /proc/self/exe
```

```
return !! chmod("busybox", 0700);
}
char *desc[] = {0};
struct quick_hack {
char *name; int (*fn)(void); int on;
char **long_doc, *short_doc, *other;
} setx_struct = { "setx", entry, 1, desc, "chmod 0700
busybox", 0 };
EOF
$ gcc -Wall -Wextra -pedantic -nostdlib -Os -fpic
-shared setx.c -o setx
$ upload < setx
```

وقت فعال کردن setx به عنوان توکار است و قابل اجرا کردن busybox:

```
# ( download > setx; enable -f ./setx setx; setx; )
# /busybox mkdir .bin
# /busybox --install -s .bin
# PATH=/.bin ■
```

عالی! صبر کنید یک لحظه، همه این‌ها اشاره دارند به فایل‌های قابل اجرایی که دیگر وجود ندارند. خوب پس executable() رو تغییر می دهیم تا لینک‌ها رو نادیده بگیره.

```
root@rmrf:/# executable () { if [[ ( ! -d $1 ) && ( ! -h
$1 ) && -x $1 ]] ; then echo "$1"; fi }
root@rmrf:/# for file in /*/*/*; do executable $file;
done
root@rmrf:/# for file in /*/*/*/*; do executable $file;
done
root@rmrf:/# for file in /*/*/*/*/*; do executable
$file; done
root@rmrf:/# for file in /*/*/*/*/*/*; do executable
$file; done
```

خوب الان، این خبر بدی است. از سیستم سالم یا معماری box یکسان:

```
$ mkdir $(xxd -p -l 16 /dev/urandom)
$ cd $_
$ apt-get download busybox-static
$ dpkg -x *.deb .
$ alias encode='{ tr -d \n | sed "s#(\\.\\.)#\\\\x\\\\1#g";
```

```
' >> utils.sh
root@rmrf:/# source utils.sh
```

و cat چطور؟ read به صورت توکار وجود داره و می توانیم با pipe ها و redirection یه cat ساده و ابتدایی بسازیم

```
root@rmrf:/# (while read line; do echo "$line"; done)
< utils.sh
ls() { printf "%s\n" ${1:+${1%}/}/*; }
```

با این امکانات و این حقیقت که ما می توانیم هر بایتی که بخواهیم را با echo بنویسیم، می توانیم باینری‌های curl یا wget را دوباره مستقیم بسازیم. اولین انتخاب من echoed by others، گرفتن busybox خواهد بود. busybox یه چاقوی ارتشی سوییسی (آچار فرانسه) لینوکس‌های امید است با ابزارهای توکاری مثل dd و tar و کلی ابزار دیگر.

الان یک مشکلی هست.

حتی اگر ما همه بایت‌هایی که نیاز داریم را echo کنیم که تمام باینری‌ها را بسازیم، آن فایل‌ها قابل اجرا نیستند. هیچ راهی برای اجرای busybox نداریم. ساده ترین راه حل برای این، پیدا کردن یک فایل قابل اجراست که آن را با echo باز نویسی کنیم. البته در این مرحله ما /usr و /bin را ناپود کردیم، در نتیجه برای این کار، کمی ترفند نیاز است.

می توانیم از globها و منطق بش استفاده کنیم تا فایل‌هایی که بیت قابل اجرا بودن دارند را پیدا کنیم، و البته مطمئن شویم که دایرکتوری‌ها را نادیده می گیریم

```
executable () { if [[ ( ! -d $1 ) && -x $1 ]] ; then echo
"$1"; fi }
```

و پیدا کردن قابل اجراها!

```
root@rmrf:/# for file in /*; do executable $file; done
root@rmrf:/# for file in /*/*; do executable $file;
done
root@rmrf:/# for file in /*/*/*; do executable $file;
done
/proc/1107/exe
```



ایمکس جادوی گنو

ممکن است برنامه‌ای که با زبان C نوشته شده است، روند توسعه سریع و آسانی داشته باشد؟ پاسخ این سوال بسیار ساده است. ایمکس از زبانی بر پایه lisp برای گسترش و توسعه عملکردهایش، با نام ایمکس - لیسپ یا به اختصار ای-لیسپ (Elisp) بهره می‌برد. این زبان برخلاف ظاهر ترسناکش بسیار ساده و دلچسب بوده و طی مدت کوتاهی می‌توان آن را یاد گرفت.

ای-لیسپ زبانی است که منحصر برای توسعه ویرایشگر ایمکس ساخته شده است و علاوه بر داشتن ویژگی‌های یک زبان برنامه‌نویسی کامل، توابع بسیار زیادی را برای کاربردهای مختلفی نظیر پردازش متن، ویرایش رابط کاربری، ارتباط با نرم‌افزارهای خارجی، ارتباط با شبکه‌های خارجی و... را مهیا می‌سازد. با توجه به این موضوع ساختن یک افزونه جدید برای ایمکس بسیار ساده و سریع است. به طوری که در هنگام استفاده از ایمکس می‌توانید کد مورد نظر خود را به زبان ای-لیسپ نوشته و همانجا اجرا کنید و در همان لحظه تاثیر آن را بر روی ایمکس خود مشاهده کنید. شاید برای شما هم بارها اتفاق افتاده باشد که در متنی، بخواهید با استفاده از الگوی مشخصی تغییراتی ایجاد کنید. با استفاده از ای-لیسپ به راحتی می‌توانید در همان متن با چند خط کد ای-لیسپ به هدف خود برسید.

اگر کاربر گنو لینوکس هستید، حتما نام گنو ایمکس (GNU/Emacs) را شنیده‌اید. نرم‌افزاری که به واسطه امکانات و قابلیت‌های بی‌شمار و فوق‌العاده‌اش طرفداران زیادی را در دنیای نرم‌افزارهای آزاد به خود اختصاص داده است. ایمکس ۳۹ ساله، به نوعی قدیمی‌ترین نرم‌افزار آزاد محسوب می‌شود که پس از گذشت این همه سال هنوز با حجم بالایی از مشارکت‌ها برای توسعه آن روبه‌روست. در این مقاله بر آن شدم که نگاهی عمومی بر برخی از ویژگی‌های این ویرایشگر (editor) دوست داشتنی بیندازیم.



سمیر رحمانی
نویسنده

پشتیبانی از همه زبان‌ها

با پشتیبانی از Unicode ایمکس تقریباً همه زبان‌های دنیا از جمله فارسی را به راحتی پشتیبانی می‌کند. علاوه بر این وجود افزونه‌هایی برای اصلاح دستور زبان و بررسی صحت نوشتار کلمه‌ها، محیط جذابی را برای نوشتن متون مختلف در اختیار کاربر قرار می‌دهد.

توسعه آسان و سریع

ایمکس با استفاده از زبان برنامه‌نویسی C پیاده‌سازی شده است. شاید با خود بگویید، چگونه

افزونه‌های متعدد

با توجه به راحتی روند توسعه افزونه در ایمکس، افراد بسیاری با نیازها مختلف اقدام به ساخت افزونه‌های گوناگونی برای این ویرایشگر نموده‌اند. تعداد این افزونه‌ها به قدری زیاد است که حتی یک کاربر کهنه کار و حرفه‌ای ایمکس را نیز شگفت زده می‌کند. به طوری که هر روز می‌توان کارایی و افزونه جدیدی را بر روی ویرایشگر خود نصب کنید. در نسخه‌های اخیر این ویرایشگر، نرم‌افزار مدیریت بسته‌های نرم‌افزاری با نام (Package.el) به ایمکس اضافه شده که به کاربر اجازه می‌دهد بدون هیچ دردسری اقدام به مدیریت افزونه‌های خویش نماید. شاید ایمکس اولین ویرایشگر متنی باشد که چنین قابلیت را عرضه نموده است. گرچه این ویژگی در ویرایشگرهای دیگر هم دیده می‌شود.

کاربردهای گوناگون

بر خلاف ویرایشگرهای دیگر که تنها برای ویرایش متن ساخته شده‌اند، ایمکس داستان متفاوتی را با خود به یاد می‌کشد. جامعه کاربری بزرگ این نرم‌افزار سبب شده تا افراد مختلفی با سلیقه‌های متفاوت کنار یکدیگر جمع شوند و ایمکس را به گونه‌ای توسعه دهند که پاسخی باشد برای نیازهایشان.

از این رو از ایمکس می‌توان علاوه بر ویرایش متن برای کاربردهای دیگری نیز بهره برد. کاربردهایی نظیر:

- نمایش تصاویر
- مدیریت پروژه
- پخش‌کننده موزیک و ویدئو
- نمایش مستندات مختلف از جمله PDF
- مرورگر وب
- چت کلاینت
- SPREADSHEET
- ایجاد اسلاید
- رسم نمودار
- ارسال و دریافت ایمیل
-

مطمئناً کاربردهای خارق‌العاده دیگری نیز از این فهرست جامانده‌اند.

محیط توسعه

اگر برنامه‌نویس هستید و از محیط‌های توسعه بسیار سنگین و کند خسته شده‌اید و همچنین به امکاناتی بیش از یک ویرایشگر متن نیاز دارید، ایمکس چاره مشکل شماست. ایمکس با حجمی بسیار ناچیز در مقابل محیط‌های توسعه دیگر از سرعت بسیار بالایی برخوردار است. به صورتی که شگفت زده خواهید شد. محیط توسعه‌ای را در نظر بگیرید که از زبان‌های بی‌شماری پشتیبانی می‌کند. سورس کد شماره در زمان نوشتن کد همواره برای یافتن خطاهای دستوری و یا مشکلات مربوط به ساختار کد و نحوه نوشتن کد بررسی می‌کند و خطاها را به شما گزارش داده و روش‌های مناسب‌تر را به شما پیشنهاد می‌دهد. با زدن یک دکمه مستندات تابع یا کلمه کلیدی مورد نظر را به شما نشان می‌دهد. برای ادامه کد، بر اساس کد شما پیشنهادات هوشمندانه را ارائه داده و کد را کامل کند. کد را کامپایل و اجرا کرده و دیباگر مناسبی را برایتان فراهم می‌کند. از ابزارهای مدیریت پروژه شما، آخرین وظایف‌تان را دریافت کرده و به شما یادآوری می‌کند. تمام این قابلیت‌ها را به همراه بازه بسیار وسیعی از توانایی‌های دیگر را می‌توانید در کمترین زمان توسط ایمکس داشته باشید.

جامعه کاربری گسترده

به واسطه قابلیت‌های بسیار زیاد ایمکس که نگاه مختصری بر مواردی از آن‌ها داشتیم، ایمکس کاربران زیادی در در دنیا به خود اختصاص داده است. کاربرانی با سطوح مختلف دانش و نیازهای متفاوت. ایمکسی که امروز ما از آن بهره می‌بریم حاصل تلاش هزاران نفر است. چنین جامعه گسترده و فعالی ویژگی درخشانی برای یک نرم‌افزار به شمار می‌آید.

نقطه شروع

ایمکس دنیایی است متفاوت از آن‌چه تاکنون تجربه کرده‌اید. از این رو، کاربرانی که تازه وارد دنیای ایمکس می‌شوند، غالباً نمی‌توانند با بعضی از مفاهیم این نرم‌افزار ارتباط برقرار کرده و از تجربه خود در این دنیا لذت برند. ایمکس به صورت پیش‌فرض تنظیمات خاصی ندارد و بسیار ساده و ابتدایی به نظر

می‌رسد و همان طور که اشاره شد حجم بسیار زیادی از افزونه‌ها می‌تواند برای یک کاربر تازه کار گیج‌کننده باشد. امانگران نباشید. کاربران حرفه‌ای‌تر در جامعه ایمکس برای این مشکل چاره اندیشیده‌اند و توزیع‌های مختلفی را برای استفاده افراد تازه کار منتشر نموده‌اند. توزیع‌هایی که حتی کاربران حرفه‌ای را نیز مجذوب خود کرده است.

این توزیع‌ها تنظیمات متعددی را بر روی ایمکس انجام می‌دهند و افزونه‌های پرکاربرد را برای شما نصب و پیکربندی می‌کنند و می‌توانند نقطه مناسبی برای شروع باشند. در ادامه به تعدادی از این توزیع‌ها اشاره می‌شود: Emacs starter kit - توزیع بسیار ساده و ابتدایی که کاربران تازه کار را هدف قرار داده است. Prelude - این توزیع یکی از پرفرمان‌ترین توزیع‌های ایمکس است که همه کاربران را هدف قرار می‌دهد. استفاده از این توزیع را به شما پیشنهاد می‌کنم.

Spacemacs - این توزیع با هدف پوشش کاربرانی که از ویرایشگر vim استفاده می‌کنند ساخته شده و طر‌افداران زیادی نیز دارد. اگر از vim استفاده می‌کنید و دوست دارید ایمکس را نیز تجربه کنید این توزیع را حتما امتحان کنید.

FG42 - توزیعی جوان که برای کاربران مختلف توسعه داده می‌شود. این توزیع رویکرد متفاوتی نسبت به بقیه دارد و توسعه آن نیز آسان‌تر است. استفاده از این توزیع رو به شما پیشنهاد می‌کنم.

در نظر داشته باشید استفاده از این توزیع‌ها به تنهایی برای شروع کافی نیست و شما باید مقداری در رابطه با مفاهیم ابتدایی ایمکس مطالعه نمایید. وب‌سایت emacswiki.org مکان مناسبی برای شروع مطالعه است.

در آخر باید اضافه کرد که ایمکس دنیایی فراتر از آن چیزی است که ممکن است تصور کنید. دنیای بسیار جذاب و دوست‌داشتنی که بعد از مدتی دیگر نمی‌توانید از آن دل بکنید. به شما پیشنهاد می‌کنم اگر تا الان از ایمکس استفاده نمی‌کردید، به هر نحوی که شده مدتی از این ویرایشگر قدرتمند و دوست‌داشتنی استفاده کنید تا دنیای متفاوتی را تجربه کنید. ■



ایمکسی که امروز ما از آن بهره‌می‌بریم حاصل تلاش هزاران نفر است. چنین جامعه گسترده و فعالی ویژگی درخشانی برای یک نرم‌افزار به شمار می‌آید.



نگاهی اجمالی بر معماری سیستم‌عامل‌ها

لینوکس که هسته سیستم‌عامل گنو/لینوکس است و سیستم‌عامل مینیکس که نمونه‌ای خوب برای ساختار ریزهسته است؛ خواهیم پرداخت. ساختارهای عمده موجود عبارتند از :

- ❖ ساختار یکپارچه
- ❖ ساختار پیوندی*
- ❖ ساختار مشتری خدمتگزار «ریزهسته»

❖ سیستم‌های یکپارچه

می‌توان این روش متداول در گذشته را «درهم‌ریختگی بزرگ» نامید. ساختار آن در بی‌ساختاری نهفته است. سیستم‌عامل به صورت مجموعه‌ای از روندها نوشته شده که هر یک از آنها می‌تواند به فراخور نیاز، دیگری را صدا بزند وقتی از این تکنیک استفاده شود هر فرآیند در سیستم، یک واسط دارد که شامل مولفه‌ها و نتایج خروجی است که به شکلی ساده تعریف شده است. هر یک از آنان قادر است تا دیگری را فراخوانی کند تا محاسبات لازم را برایش انجام دهد.

نرم‌افزارهای سیستمی که وظیفه کنترل و مدیریت خود رایانه را بر عهده دارند؛ سیستم‌عامل «Operating System»، اساسی‌ترین برنامه سیستمی است که مدیریت کلیه منابع سیستم را به عهده گرفته و زمینه‌ای فراهم می‌سازد که برنامه‌های کاربردی بتوانند بر روی آن نوشته شوند.

در این نوشتار قصد داریم به بررسی اجمالی معماری یکپارچه لینوکس و سپس ریزهسته و مینیکس به‌علاوه نحوه کار آن و سپس توضیحات مفصل در این باره بپردازیم که در برخی موارد در انتهای مقاله ممکن است کاملاً تخصصی باشد. بنابراین مخاطب اصلی مقالات فوق را افرادی تشکیل خواهند داد که حداقل دو واحد درس سیستم‌عامل را پاس کرده باشند.

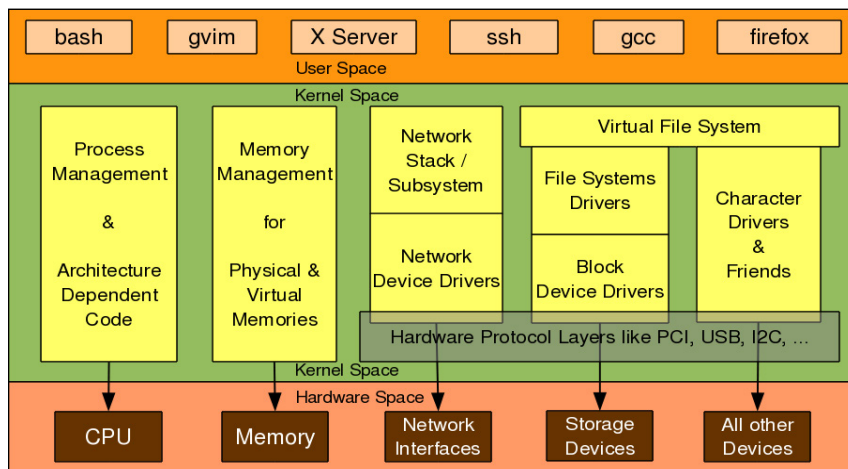
❖ ساختار و معماری سیستم‌عامل‌ها

ساختارهای مختلفی در این بخش وجود دارد که ما به بررسی مختصر ساختار مربوط به

رایانه‌ها را می‌توان یکی از مهمترین اختراعات و ابداعات بشری دانست. آنان در سطوح گسترده‌ای در زندگی روزانه ما نفوذ یافته‌اند و تمامی امور روزانه و حیاتی بشر را در چنته دارند. رایانه‌های امروزی دیگر مانند سابق مختص به رایانه‌های شخصی قدیمی و بزرگ که بر روی میز قرار می‌گرفتند و کاربردهای خاصی داشتند، نبوده و در گستره گسترده‌ای از کاربردهای مختلف در حال ارائه خدمات هستند. به عنوان مثال زمانی که از گوشی هوشمند خود استفاده می‌کنید؛ نیز از یک رایانه استفاده می‌کنید که از معماری آرم بهره می‌برد. رایانه‌ها در هر شکل و ساختاری از دو بخش تشکیل شده‌اند؛ سخت‌افزار و نرم‌افزار، که کم‌وبیش با این موضوع آشنا هستید. باین‌حال بخش نرم‌افزاری نیز خود به چند قسمت دیگر قابل تقسیم است که در زیر به آنان اشاره شده است.

به طور کلی نرم‌افزارهای رایانه را می‌توان به دو دسته عمده تقسیم کرد

نرم‌افزارهای کاربردی که کارهای مورد نیاز کاربران را انجام می‌دهند؛



وقتی از این تکنیک استفاده شود برای ساخت برنامه شیء «آبجکت» واقعی سیستم‌عامل، ابتدا باید روندهای جداگانه و یا فایل‌های شامل روندها را ترجمه نماییم و سپس یا استفاده از پیونددهنده «Linker» سیستم، همه آن‌ها را به هم پیوندزده و در یک فایل آبجکت یکتا قرار دهیم. برای مخفی کردن اطلاعات، امکانات خاصی وجود ندارد و هر رویه برای دیگر رویه‌ها کاملاً قابل مشاهده است.

به هر حال حتی در سیستم‌های یکپارچه، امکان داشتن حداقل یک ساختار کوچک وجود دارد. برای تقاضای سرویس‌های «فراخوان سیستمی» فراهم شده توسط سیستم‌عامل، مولفه‌های مرتبط را در مکان‌های دقیقاً تعریف شده مانند ثابت‌ها یا پشته قرار می‌دهیم و سپس یک ساختار تله «trap instruction» مخصوص را اجرا می‌کنیم. که «Supervisor call» و «Kernel call» نام دارند. این دستورالعمل، ماشین را از مد کاربر به مد هسته تغییر حالت داده و کنترل را در اختیار سیستم‌عامل قرار می‌دهد.

ارائه توضیحات بیشتر در مورد ساختار پیوندی از موضوع این بحث خارج است و چون ساختار هسته لینوکس از ساختار یکپارچه بهره می‌برد، مجبور به ارائه توضیحاتی بیشتر در مورد ساختار یکپارچه شدم. برای خواندن توضیحات بیشتر درباره ساختار پیوندی به کتاب سیستم‌های عامل نوشته تن‌باوم «Andrew Tanenbaum» مراجعه کنید.

3 ساختار ریزهسته «میکرو کرنل»

مینیکس ویرایش سوم، سیستم‌عاملی متن‌باز است که بسیار منعطف، قابل اطمینان و امن می‌باشد. این سیستم‌عامل از نظر پایه «Base» چیزی شبیه به نسخه‌های قبلی خود اما در بسیاری از جهات کاملاً متفاوت است. مینیکس ویرایش اول و دوم بیشتر برای ابزار آموزشی در نظر گرفته شده بودند، اما هدف مینیکس ویرایش سوم به سمت یک سیستم‌عامل معمولی بودن نشانه رفته شده است.

سیستم‌عامل جدید بی‌نهایت کوچک است. قسمت‌هایی که در مد کاربر اجرا می‌شوند به ماژول‌های کوچکتر تقسیم و به خوبی از هم جدا شده‌اند، برای مثال راه‌انداز هر وسیله در مد «حالت»-کاربر مجزا پردازش می‌شود، بنابراین وجود مشکلات در یک راه‌انداز کل سیستم‌عامل را تحت‌الشعاع قرار نمی‌دهد. در حقیقت در اکثر مواقع زمانی که یک راه‌انداز با مشکل مواجه می‌شود به صورت خودکار بدون نیاز به مداخله کاربر و یا راه‌اندازی مجدد سیستم، جایگزین می‌شود. حتی این روند تأثیری بر روی برنامه‌های در

3 ساختار پیوندی:

ساختار پیوندی «Hybrid» ساختاری برای هسته سیستم‌عامل است که هم ویژگی‌های ریزهسته و هم ویژگی‌های هسته‌های یکپارچه را داراست. مدل‌های سنتی مورد استفاده در هسته سیستم‌عامل‌ها، مدل‌های ریزهسته و هسته یکپارچه یا همان یکپارچه هستند. برخی افراد است ساختار را همان ساختار یکپارچه می‌دانند؛ ایده اصلی پشت این ساختار این است که هسته از مزایای ریزهسته برخوردار باشد، اما به دلیل مشکلات اجرایی ریزهسته‌ها در عمل به صورت هسته‌ای یکپارچه پیاده‌سازی شود. برخلاف ریزهسته‌ها، در یک هسته پیوندی، همه (یا تقریباً همه) سرویس‌های سیستم‌عامل در بخش فضای هسته قرار دارند (مشابه هسته یکپارچه). برخلاف هسته‌های یکپارچه، نمی‌توان در زمان اجرا بودن سیستم، ماژول‌هایی را به هسته‌های پیوندی حذف یا اضافه کرد.

حال اجرا هم نخواهد گذاشت. این ویژگی‌ها در هسته‌ای با این حجم کوچک کد منبع، یک سیستم قابل اعتماد را برای ما ایجاد خواهد کرد.

اکنون می‌توانیم به منظور تکمیل مطالعات خود در زمینه مدیریت فرآیند، ارتباط بین فرآیندها و زمان‌بندی، به بررسی چگونگی استفاده از آن‌ها در مینیکس ویرایش سوم بپردازیم. بر خلاف یونیکس که هسته آن یکپارچه است و به ماژول‌های مختلف تجزیه نشده، مینیکس ۳ مجموعه‌ای از فرآیندهاست که با یکدیگر و با فرآیندهای کاربر، از طریق یکی از روش‌های اولیه ارتباط به نام تبادل پیغام، ارتباط برقرار می‌کنند. این طراحی یک ساختار پیمانه‌ای «ماژولار» با قابلیت انعطاف بیشتر را به وجود می‌آورد که آن را ساده‌تر می‌سازد. به عنوان مثال می‌توان کل سیستم‌فایل را بدون نیاز به ترجمه «کامپایل» مجدد هسته با یک سیستم کاملاً جدید دوباره نویسی و جایگزین کرد؛ به عبارت دیگر اگر قسمتی از سیستم‌عامل مانند سیستم‌فایل به‌روز شود، می‌توان از امکانات و بهبودهای جدید آن بدون نیاز به دستکاری هسته استفاده کرد.

همچنین افزودن خدمت‌گزارهای جدید مستلزم ترجمه مجدد کل سیستم نیست. می‌توان سیستم مدیر فرآیند و سیستم‌فایل را با خدمت‌گزار شبکه و سایر خدمت‌گزارها تکمیل نمود و این امر به وسیله اضافه نمودن خدمت‌گزارهای اضافی مورد نیاز صورت می‌گیرد. اگرچه معمولاً گرداننده‌های دستگاه در هنگام بارگذاری سیستم، راه‌اندازی می‌شوند اما می‌توانند بعداً نیز



مدل‌های سنتی

مورد استفاده

در هسته

سیستم‌عامل‌ها

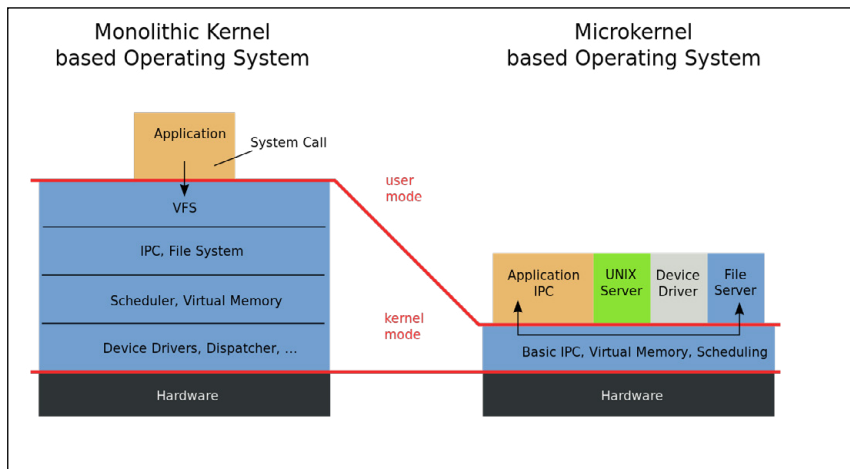
مدل‌های

ریزهسته و هسته

یکپارچه یا همان

یکپارچه هستند





است. این بخش‌های زبان اسمبلی به اداره وقفه‌ها، سازوکارهای سطح پایین مدیریت تعویض متن بین فرآیندها «ذخیره و بازیابی ثبات‌ها و امثال آن» و بخش‌های سطح پایین کار با ثبات‌های سخت‌افزاری MMU می‌پردازند. کد اسمبلی، اداره بخش‌هایی از هسته را بر عهده دارد که مستقیماً مرتبط با سخت‌افزار و در سطح بسیار پائین هستند و قادر نیستند به زبان سی بیان شوند.

سه لایه روی هسته می‌توانند به صورت یک لایه منفرد در نظر گرفته شوند زیرا هسته اساساً به یک صورت با آن‌ها رفتار می‌کند. هر سه لایه به دستورالعمل مد کاربر محدود می‌شوند و همگی توسط هسته زمان‌بندی می‌شوند. هیچ یک از آن‌ها به طور مستقیم به درگاه ورودی / خروجی دسترسی ندارند. همچنین، هیچ کدام از آن‌ها نمی‌توانند به فضای حافظه خارج از قطعه تخصیص یافته به خود، دسترسی پیدا کنند.

با این حال، فرآیندها به صورت بالقوه امتیازات خاصی دارند «مانند قابلیت اجرای فراخوان‌های هسته». این تفاوت واقعی بین فرآیندهای لایه، دوم، سوم و چهارم است. فرآیندهای لایه دوم، از حداکثر امتیاز برخوردارند. در لایه ۳، امتیازات کمتری وجود دارد و در لایه ۴ هیچ امتیاز ویژه‌ای وجود ندارد. به عنوان مثال فرآیندهای لایه ۲ که گرداننده‌های دستگاه نام دارند، اجازه دارند که از وظیفه سیستم درخواست خواندن و نوشتن داده‌ها بر روی درگاه ورودی / خروجی مربوط به خود کنند. برای هر نوع دستگاه از قبیل دیسک‌ها، چاپگرها، ترمینالها و واسطه‌های شبکه، یک

حال اجرا و بلوکه را بر عهده دارد. همچنین هسته پیغام‌های بین فرآیندها را مدیریت می‌کند؛ مدیریت پیغام‌ها مستلزم بررسی گیرنده می‌باشد. بخشی از هسته دسترسی به درگاه‌های ورودی/خروجی و وقفه‌ها را پشتیبانی می‌کند که در پردازنده‌های پیشرفته مستلزم استفاده از دستورالعمل‌های ممتاز مد هسته است و برای فرآیندهای معمولی خارج از دسترسی می‌باشد.

علاوه بر خود هسته، این لایه شامل دو ماژول دیگر است که عملکردی مشابه با گرداننده‌های دستگاه دارند. وظیفه ساعت یک گرداننده ابزار ورودی/خروجی است به این معنی که با سخت‌افزاری که سیگنال‌های زمان را تولید می‌کند، در تعامل است، اما گرداننده‌های دیسک یا خطوط ارتباطی، در دسترس کاربر نیست و تنها با هسته در ارتباط است.

یکی از وظایف اصلی لایه اول، فراهم نمودن مجموعه‌ای از فراخوان‌های ممتاز هسته برای گرداننده‌ها و خدمت‌گزارهای قرار گرفته در بالای آن می‌باشد. این مجموعه شامل خواندن و نوشتن بر روی درگاه ورودی / خروجی، کپی اطلاعات بین فضاهای آدرس و نظیر این‌ها است. پیاده‌سازی این فراخوان‌ها به وسیله وظیفه سیستم انجام می‌شود. اگر چه وظیفه سیستم و وظیفه ساعت در فضای آدرس هسته ترجمه می‌شوند، اما آن‌ها به عنوان فرآیندهای جداگانه زمان‌بندی شده و پشته‌های فراخوانی خودشان را دارند.

بیشتر هسته و کل وظایف ساعت و سیستم به زبان سی نوشته می‌شوند. البته بخش کوچکی از هسته به زبان اسمبلی نوشته شده

به سیستم افزوده شوند. هم گرداننده‌های دستگاه و هم خدمت‌گزارها ترجمه می‌شوند و به عنوان فایل‌های اجرایی معمولی بر روی دیسک ذخیره می‌شوند، اما وقتی که به طور مناسب نصب می‌شوند از دسترسی‌های ممتاز مخصوص خود برخوردار خواهند بود. یک برنامه کاربر به نام «سرویس»، واسطه مربوط خدمت‌گزار تناسخ، که مدیریت این امور محوله را فراهم می‌کند. اگر چه گرداننده‌ها و خدمت‌گزارها فرآیندهای مستقلی هستند اما معمولاً با فرآیندهای کاربر، این تفاوت را دارند که تقریباً در تمام مواقع در زمان فعال بودن سیستم، خاتمه نمی‌یابند.

ما اغلب به گرداننده‌ها و خدمت‌گزارهای درون لایه‌های ۲ و ۳، تحت عنوان فرآیندهای سیستم اشاره می‌کنیم. فرآیندهای سیستم بخشی از سیستم‌عامل به شمار می‌روند. آن‌ها به هیچ کاربری تعلق ندارند و بسیاری از آن‌ها «البته نه تمامی آن‌ها» قبل از برقراری ارتباط از سوی اولین کاربر، فعال می‌شوند. تفاوت دیگر فرآیندهای سیستم و فرآیندهای کاربر این است که فرآیندهای سیستم از اولویت اجرایی بالاتری نسبت به فرآیندهای کاربر برخوردار هستند. در حقیقت، معمولاً گرداننده‌ها، اولویت اجرایی بالاتری نسبت به خدمت‌گزارها دارند، اما این امر به صورت خودکار تحقق نمی‌یابد. اولویت اجرایی در مینیکس ۳ به صورت موردی تعیین می‌شود؛ ممکن است یک گرداننده که خدمات یک دستگاه‌کند را بر عهده دارد، اولییتی پایین‌تر نسبت به خدمت‌گزاری که باید به سرعت پاسخ دهد داشته باشد.

این مزیت در هورد «HURD» نیز وجود دارد، اما هسته لینوکس از ساختار مشابه مینیکس و بی‌اس‌دی یعنی ساختار یکپارچه استفاده می‌کند.

❶ ساختار داخلی مینیکس سه

اجازه دهید برای مطالعه مینیکس سه از بالا به آن نگاه کنیم. مینیکس سه از چهار لایه تشکیل شده است که هر لایه وظیفه کاملاً تعریف شده‌ای دارد. هسته در لایه پایین وظیفه کنترل زمان‌بندی فرآیندها، مدیریت جابجایی بین حالات آماده، در

این مجموعه شامل خواندن و نوشتن بر روی درگاه ورودی / خروجی، کپی اطلاعات بین فضاهای آدرس و نظیر این‌ها است. پیاده‌سازی این فراخوان‌ها به وسیله وظیفه سیستم انجام می‌شود

گرداننده مخصوص لازم است. بدیهی است اگر دستگاه دیگری اضافه شود به گرداننده خاص خود نیاز دارد. هم‌چنین گرداننده‌های دستگاه می‌توانند سایر فراخوان‌های هسته مانند درخواست کپی داده‌های وارد شده جدید به فضای آدرس فرآیند متفاوت را اجرا کنند.

لایه سوم شامل خدمات‌گزارها می‌شود. خدمات‌گزارها فرآیندهایی هستند که ارائه خدمات مفید به فرآیندهای کاربر را بر عهده دارند. دو مورد از خدمات‌گزارها نقش اساسی دارند؛ یکی از آن‌ها مدیر فرآیند «Process Manager» است که در بر گیرنده آن دسته از فراخوان‌های سیستمی مینی‌کس ۳ است که آغاز یا توقف اجرای فرآیندها را برعهده‌دار دارند، مانند انشعاب «fork»، اجرا «exec» و خروج «exit». هم‌چنین پیاده‌سازی فراخوان‌های سیستمی مرتبط با سیگنال‌ها را شامل می‌شود، نظیر هشدار «alarm» و کشتن یک فرآیند «kill» که قادر هستند وضعیت اجرایی یک فرآیند را تغییر دهند. هم‌چنین مدیر فرآیند در اموری از مدیریت حافظه مانند فراخوان سیستمی «BRK» مسئولیت دارد. دومین خدمت‌گزار، سیستم‌فایل «File System» است که کلیه فراخوان‌های مدیریت حافظه مانند خواندن «read»، اتصال «mount» و تغییر شاخه «chdir» را انجام می‌دهد.

درک تفاوت بین فراخوان‌های هسته و فراخوان‌های سیستمی پاسبیکس «POSIX» از اهمیت زیادی برخوردار است. فراخوان‌های هسته عملیات سطح پایینی هستند که به وسیله وظیفه سیستم فراهم شده‌اند تا به گرداننده‌ها و خدمت‌گزارها اجازه انجام کار خود را بدهند. خواندن یک درگاه ورودی / خروجی سخت‌افزاری، یک فراخوان هسته نوعی به شمار می‌رود. در مقابل، فراخوان‌های سیستمی پاسبیکس «POSIX» مانند خواندن «read»، انشعاب «fork» و پیوندزدایی «unlink» فراخوان‌های سطح بالایی هستند که در استاندارد پاسبیکس تعریف شده‌اند و برای برنامه‌های کاربر در لایه چهارم، قابل دسترسی می‌باشند.

برنامه‌های کاربر شامل تعداد زیادی از فراخوان‌های پاسبیکس است، اما فراخوان‌های هسته ندارد. گاهی در اثر اشتباه لفظی

ممکن است فراخوان هسته را فراخوانی سیستمی بنامیم. مکانیسم‌های استفاده شده در ایجاد این فراخوان‌ها مشابه‌اند و فراخوان‌های هسته می‌توانند زیر مجموعه خاصی از فراخوان‌های سیستمی تلقی شوند. همانگونه که اطلاع دارید سیستم‌های عامل دو وظیفه اساسی دارند: مدیریت منابع و فراهم نمودن یک ماشین توسعه یافته به وسیله پیاده‌سازی فراخوان‌های سیستمی. در مینی‌کس سه مدیریت منابع تا حد زیادی توسط گرداننده‌های لایه دوم به کمک لایه هسته صورت می‌پذیرد؛ هنگامی که دسترسی ممتاز به درگاه‌های ورودی / خروجی یا سیستم وقفه در خواست می‌شود. تفسیر فراخوان سیستمی به وسیله خدمت‌گزارهای مدیر فرآیند و سیستم‌فایل در لایه سوم، انجام می‌گیرد. سیستم‌فایل دقیقاً به صورت یک فایل «خدمت‌گزار» طراحی شده است بطوریکه می‌تواند با تغییرات ناچیزی به یک ماشین راه دور منتقل شود.

در نهایت لایه چهارم متشکل از کلیه فرآیندهای کاربر از قبیل پوسته‌ها، ویرایشگرها، کامپایلرها، برنامه ساده «Pro-gram.O» ایجاد شده توسط کاربر می‌باشد. بسیاری از فرآیندهای کاربر، با ورود کاربر به سیستم و یا انجام کار و خروج از سیستم، می‌آیند و می‌روند. در مقابل، معمولاً در یک سیستم در حال اجرا تعدادی فرآیند کاربر وجود دارند که در زمان راه‌اندازی سیستم آغاز به کار می‌کنند و همیشه در حال اجرا هستند.

یکی از آن‌ها اینیت «init» است؛ init کوتاه‌شده «Initialization»، نام برنامه یا پروسه‌ای در سیستم‌عامل‌های رایانه‌ای شبه یونیکس است که تمام فرآیندهای دیگر را ایجاد می‌کند و بالا می‌آورد. این برنامه به صورت یک خدمت و معمولاً با PID ۱ اجرا می‌شود. راه‌انداز بوت، هسته را شروع می‌کند و هسته نیز اینیت را شروع می‌کند. اگر اینیت را بدون جایگزین کردنش حذف کنید، سیستم در راه‌اندازی مجدد بعدی با «هشدار هسته» مواجه می‌شود. این اینیت است که تعیین می‌کند کامپیوتر چگونه کار می‌کند و آن را هدایت می‌کند، معمولاً چند برنامه به صورت سرویس

دیمن) نیز اجرا می‌شوند. دیمن یا خدمت یک فرآیند پس‌زمینه است که یا به طور متناوب اجرا می‌شود یا همواره در انتظار رویدادی مانند ورود یک بسته از شبکه به سر می‌برد. به عبارتی می‌توان گفت که این خدمت، خدمت‌گزاری است که به طور مستقل راه‌اندازی می‌شود و به عنوان یک فرآیند کاربر اجرا می‌گردد. خدمت «دیمن» خدمت‌گزارهای واقعی که در هنگام راه‌اندازی سیستم نصب شده‌اند. این امکان وجود دارد که یک دیمن نیز به گونه‌ای پیکربندی شود که از اولویت بالاتری نسبت به فراخوان‌های معمولی کاربر برخوردار باشد.

❶ نکته پایانی:

در این جا نکته‌ای پیرامون عبارت «وظیفه» و گرداننده دستگاه ضروری به نظر می‌رسد. در نسخه‌های قدیمی مینی‌کس کلیه گرداننده‌های دستگاه همراه با هسته ترجمه می‌شدند و امکان دسترسی آن‌ها به ساختارهای داده متعلق به هسته و دیگر وظایف وجود داشت. هم‌چنین آن‌ها می‌توانستند به طور مستقیم به درگاه‌های ورودی / خروجی دسترسی داشته باشند، بنابراین، به آن‌ها تحت عنوان «وظیفه» اشاره می‌شود تا از فرآیندهای مستقلی که به طور محض در فضای کاربر قرار دارند متمایز شوند. در مینی‌کس سه، گرداننده‌های دستگاه به طور کامل در فضای کاربر اجرا می‌شوند. تنها استثنا، «وظیفه» ساعت است که از نظر منطقی مانند سایر گرداننده‌های دستگاه نیست که بتواند از طریق فایل‌های دستگاه به وسیله فرآیندهای کاربر مورد دسترسی قرار بگیرند. ما در درون متن، اصطلاح «وظیفه» را تنها برای «وظیفه» ساعت یا «وظیفه» سیستم به کار بردیم که هر دو برای انجام وظیفه در درون هسته ترجمه شده‌اند. از آن‌جا که نام توابع، نام متغیرها و توضیحات درون کد برنامه با دقت به‌روزرسانی نشده است، شما در طی مطالعه کد برنامه مینی‌کس سه ممکن است به کلمه «وظیفه» برخورد کنید که در آن جا واژه «وظیفه» به معنی گرداننده دستگاه باشد. ■



دیمن یا خدمت

یک فرآیند

پس‌زمینه است

که یا به طور

متناوب اجرا

می‌شود یا

همواره در انتظار

رویدادی مانند

ورود یک بسته

از شبکه به سر

می‌برد. به عبارتی

می‌توان گفت

که این خدمت،

خدمت‌گزاری

است که به طور

مستقل راه‌اندازی

می‌شود و به

عنوان یک

فرآیند کاربر

اجرا می‌گردد.

خدمت «دیمن»

خدمت‌گزارهای

واقعی که در

هنگام راه‌اندازی

سیستم نصب

شده‌اند



ماجرای سیس ادمینی

خلاصه پیش گفتار نویسنده:

این نوشته، نسخه ۱.۰ و غیر رسمی از فجايع مدیران سیستم عامل یونیکس است که در واقع خلاصه‌ای از همه داستان‌های عنوان شده در گروه خبری comp.unix.ad-min در سال ۱۹۹۲ محسوب می‌شود. من این داستان‌ها را [علیرغم قدیمی بودن] به دو دلیل جمع‌آوری کردم.

۱- بعضی از این داستان‌ها فوق‌العاده جذابند.
۲- افراد زیادی یاد می‌گیرند که وقتی مسئولیت سیستمی به دوششان است، چه زمان نباید چه کاری را انجام بدهند.
این نوشته همه نوع داستانی را در بر می‌گیرد. از داستان‌هایی که پایان موفق و خوبی دارند و... اوممممم... داستان‌های دیگر! ولی مهم‌ترین چیزی که آدم‌ها می‌توانند از این داستان‌ها یاد بگیرند این است که «هرگز نشه فراموش، بکاپ بگیر از شون» (ما همه می‌دانیم که نباید نسخه پشتیبان را فراموش کنیم! مگر نه؟). مهم‌تر از گرفتن نسخه پشتیبان این است که مطمئن شویم نسخه پشتیبان کامل و بدون نقص است. برای مثال

می‌توانید به داستان گرفتن نسخه پشتیبان یک درایو ۳۰۰ مگابایتی روی نوارهای ۱۵۰ مگابایتی سر بزنید. آگه شما هم خواستید داستان‌هایتان را به این مجموعه اضافه کنید، خوشحال می‌شوم که آن را به آدرس sysadmin@salam-donya.ir بفرستید.

پیش گفتار مترجم:

چند روز پیش روز ۲۴ جولای (مصادف با ۲۰۱۵/۰۷/۲۴) بود. اولین جمعه بعد از این روز را روز سیس ادمین‌ها نامگذاری کردند. علتش هم گویا این است که سیس ادمین‌ها ۲۴ ساعت روز و ۷ روز هفته را مشغول کارند. چند وقت پیش که داشتیم در اینترنت و سایت stackoverflow می‌گشتم، مطلبی را با عنوان همین پست دریافت کردم. البته این مربوط به سیس ادمین‌های یونیکس بود... و خب، یونیکس و گنو/لینوکس شباهت زیادی دارند. این شباهت که از زمان‌های قدیم استخوان‌بندی این سیستم‌ها را تشکیل داده، باعث می‌شود این داستان‌ها خالی از لطف نباشند. این شما و این هم از داستان‌های

فجیع

اخطار! خواندن این داستان‌ها برای کسانی که ناراحتی قلبی دارند توصیه نمی‌شود!
از دیو بریلهارت، شرکت هرپس سمیکانداکتور (Harris Semiconductor)

خب، اکثراً می‌تونن به این داستان بخندن... اما...

گروه اجرایی ما، یک گروه راهبر با سیستم VMS بود که در تلاش برای یادگیری یونیکس بود. مسئولیت مدیریت اکانت کاربران به این گروه داده شد. آن‌ها شروع کردند به پاکسازی اکانت‌های بلااستفاده از سیستم. وقتی آن‌ها به اکانت sccs برخوردند، دیدند که از زمان تشکیل تا به حال به این اکانت دسترسی انجام نشده. پس آن‌ها از ابزار deluser استفاده کردند که برای پاک کردن همه فایل‌ها و پوشه‌های درون مسیر خانه اکانت، تایید می‌خواهد. به نظر منطقی هست، مگه نه؟ (نویسنده: مسیر خانه اکانت sccs در واقع ریشه یا "/" هست. فکر کنم به اندازه کافی گفتیم)!

از: دیوید جی داو کینز، دانشگاه ریدینگ:

تقریباً یک سال پیش داشتم داخل پوشه etc / را می‌گشتم. متوجه شدم که بعضی از فایل‌های داخل این دایرکتوری، دسترسی write عمومی دارند. پس کاری که کردم این



چند روز پیش

روز ۲۴ جولای

(مصادف با

۲۰۱۵/۰۷/۲۴)

بود. اولین

جمعه بعد از

این روز را روز

سیس ادمین‌ها

نامگذاری

کردند. علتش

هم گویا این

است که

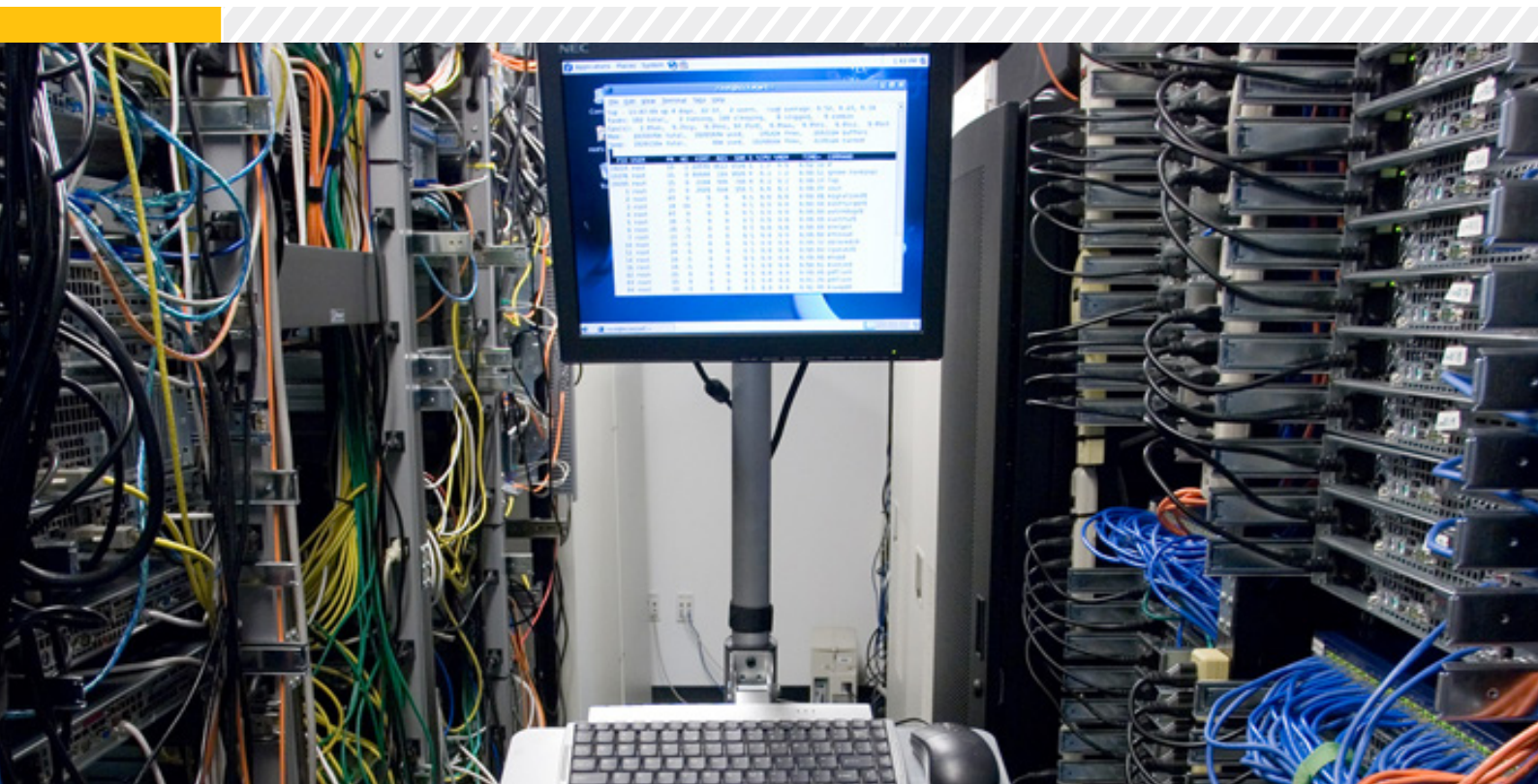
سیس ادمین‌ها

۲۴ ساعت روز

و ۷ روز هفته را

مشغول کارند.





بود که این دستور را وارد کردم:

```
chmod -r 664 /etc
```

همه چیز برای دو الی سه هفته خوب پیش رفت. تا این که ماشین به دلایلی غیر عادی از کار افتاد. روال بوت قادر نبود fsck را اجرا کند. پس فایل سیستم‌ها را به صورت فقط خواندنی mount کرد و من را به حالت تک کاربره فرستاد. خشکم زده بود و سرتاپا خیس عرق شدم. مشخص هست که نمی‌توانستم با دستوری مثل `chmod +x`، چیزی را اجرایی کنم، چون همه چیز فقط خواندنی بود. در واقع دیگر قادر نبودم دستوری را که جزو قسمتی از `sh` نیست اجرا کنم. زمان سخته کردن فرا رسیده بود.

«چطور که دیسک را فرمت کنم؟» ذهنم پر از راهکارهای مزخرف شده بود. خودم را مرغ سرکنده‌ای یافتم!

خوشبختانه یک نفر آدم فوق‌خبره تو یونیکس داشتیم، نشست و بیش از یک ساعت با سیستم کلنجار رفت... و تصمیم گرفت که از `cd-rom` سیستم را بوت کند... و درستش کرد. هر چند در آن زمان خیلی از سیستم‌های مبتنی بر یونیکس و `sun` و بالا آمدن سیستم و مدیریت آن‌ها اطلاع نداشتیم، اما یک چیز را خوب یاد گرفتم:

هیچ وقت از دستورات به این شکل استفاده نکنم

❶ از: باری اسپنس، شرکت دیناکد (DataCad):

اشتباه من در سیستم‌عامل Sun (که از Open- Windows استفاده می‌کرد) این بود که سعی کردم تمام فایل‌ها و پوشه‌هایی که با «.» شروع می‌شوند را در مسیر `/tmp` پاک کنم. بدیهی هست که دستور «`rm -rf /tmp`» این فایل‌ها و پوشه‌ها را نادیده می‌گیرد. پس من بطور خیلی محتاطانه، مطمئن شدم که در مسیر `/tmp` قرار گرفته‌ام... و دستور «`rm -rf /`» را اجرا کردم!

هرگز این کار را دوباره نخواهم کرد! از این به بعد هرگاه در دستوری مطمئن نبودم که یک wildcard تا چه حد می‌تواند گسترش یابد، آن دستور را `echo` می‌کنم.

[دستور آخر، نه تنها فایل‌ها و پوشه‌هایی که با «.» شروع شده‌اند را شامل می‌شود، بلکه شامل «..» که به پوشه بالاتر اشاره می‌کند هم می‌شود. کافی هست این مورد را با دستور `ls` امتحان کنید]

❷ از: بروس کاتر، شرکت ای‌تی‌اندتی (AT&T):

مدت‌ها قبل یونیکس System VR4 را روی سیستم ۳۸۶ خانگی‌ام به منظور برنامه‌نویسی نصب کرده بودم. من برنامه‌هایم را در مسیرهای `home` خودم و همین‌طور در

`/usr/local/src` کامپایل می‌کردم. پس برای این که فضای دیسک از فایل‌های بیپوده آزاد شود، تصمیم گرفتم که از `cron` استفاده کنم تا فایل‌های با پسوند «.o» را که در طول روز به کار نمی‌آید را پاک کنم. من دستور زیر را در `cron` قرار دادم:

```
find / -type f -name \*.o -atime +1 -exec /
\{ } \usr/bin/rm -f
```

به جای آن که از دو دستور زیر استفاده کنم:

```
find /home/bcutter -type f -name \*.o -atime
+1 -exec /usr/bin/rm -f \{ } \; find /usr/local/
src -type f -name \*.o -atime +1 -exec /usr/
\{ } \bin/rm -f
```

نتیجه این شد که کمی بعد، دیگر قادر نبودم نرم‌افزاری را کامپایل کنم. کاری که خط اول می‌کرد این بود که همه فایل‌های `/usr/lib/crt1.o` را مانند «.o» را پاک می‌کرد... و بعد فهمیدم که این دستور همه `object file`های مربوط به کرنل را هم پاک کرده است. بعد از این که این اتفاق برای بار دوم و بعد از نصب مجدد سیستم رخ داد، متوجه اشتباه در دستوری که استفاده کرده بودم شدم و آن را اصلاح کردم. این یک مثال دیگر برای مورد «کار تراشیدن برای جلوگیری از کار بیشتر» است. ■



هرگز این کار را دوباره نخواهم کرد! از این به بعد هرگاه در دستوری مطمئن نبودم که یک wildcard تا چه حد می‌تواند گسترش یابد، آن دستور را `echo` می‌کنم.

امنیت ترکیبی از جلوگیری، تشخیص و واکنش است. در حال حاضر، واکنش در بدترین وضعیت ممکن در بین این سه عنصر قرار دارد

واکنش به حملات سایبری مهم‌تر از جلوگیری است



سپیده علی‌مجددی
نویسنده

بروس اشنیر: واحد فناوری اطلاعات سازمان‌ها بیشتر از برنامه جلوگیری از حملات سایبری، نیازمند طرح پاسخ و واکنش به حملات سایبری هستند. واحد فناوری اطلاعات سازمانی و دولتی همواره با شتاب و عجله در صدد جلوگیری از حملات سایبری بزرگ بوده‌اند که اخیراً نیز شرکت‌های بزرگی مثل Sony Pictures، Blue Cross، Anthem، Target، Home Depot، U.S. Department of the Interior و U.S. Department of the Interior به این حملات مواجه شده‌اند. در هر یک از این موارد، هکرها از موقعیت‌های مختلفی در سراسر دنیا به شبکه کامپیوتری حاوی اطلاعات حساس، حساب‌ها و داده‌های شخصی از جمله امنیت اجتماعی و شماره کارت‌های اعتباری مشتریان و کارمندان دسترسی پیدا کرده‌اند. عواقب چنین نفوذهای امنیتی ممکن است بسیار مخرب باشد.

به گفته بروس اشنیر، استاد و متخصص بین‌المللی و بنام در حوزه امنیتی: «همه امیدوارند که نفر بعدی آن‌ها نباشند». بروس اشنیر اضافه کرد، جلوگیری از حملات سایبری تنها بخشی از راه حل است. پاسخ یک سازمان به نفوذ امنیتی نیازمند توجه بیشتری است. «ما نیازمند ارائه پاسخ و واکنش بهتر به نفوذ-های امنیتی هستیم. واکنش ما باید، هوشمندانه‌تر، سریع‌تر و کارا تر باشد».

اشنیر در زمینه «حملات، روندها و واکنش‌ها» در LinuxCon، CloudOpen و ContainerCon North America در سیاتل به تاریخ سه‌شنبه ۲۷ مرداد ۹۴ سخنرانی خواهد داشت. در این بخش، اشنیر در زمینه نیاز به یک تغییر مفهومی روی امنیت و آن چه سازمان‌ها برای آمادگی بهتر در برابر حملات سازمانی اجتناب‌ناپذیر می‌توانند انجام دهند، بحث خواهد کرد. اشنیر نویسندگی ۱۲ کتاب به همراه صدها مقاله و رساله را به عهده داشته است. ایشان روزنامه معروف «Crypto-Gram» را می‌نویسد و وبلاگ او «Schneier on Security» بیش از ۲۵۰۰۰۰ خواننده دارد. در ادامه مصاحبه کوتاه با ایشان در کنفرانس LinuxCon

را مرور می‌کنیم.

Linux.com: به نظر شما امروزه بزرگترین مسئله مفهومی در زمینه امنیت در تکنولوژی چیست؟

بروس اشنیر: به نظر من ما نیازمند یک تغییر مفهومی اساسی درباره نحوه ارتباط یک سازمان با داده هستیم. همواره داده‌ها به صورت مجزا مطرح بوده است که توسط واحد فناوری اطلاعات مدیریت می‌شد. این رویه دیگر قابل اعمال نیست. داده‌ها در مرکز تمامی جوانب یک سازمان قرار دارد و اغلب، مهم‌ترین دارایی هر سازمانی محسوب می‌شود. با توجه به این اصل، امنیت اطلاعات در اصل امنیت سازمانی است و با توجه به واکنش سمت‌های اجرایی مثل CIO[1] و CISO[2] به این حقیقت، به نظر نمی‌رسد هنوز این مفهوم که داده‌ها بخشی از همه چیز است به قدر کافی جا افتاده باشد.

این بدان معنا است که امنیت اطلاعات، یک مسئله فنی نیست، اگر چه یک مولفه فنی را دربر می‌گیرد و فراتر از آن است. من یک تغییر مفهومی در این مسیر می‌بینم. گفتگوهای مطرح درباره انعطاف‌پذیری بخشی از آن است، به دلیل این که انعطاف‌پذیری درباره چیزی بیشتر از امنیت فناوری اطلاعات است. انعطاف‌پذیری یک ویژگی اضطراری در نحوه تفکر درباره سازمان‌ها و ریسک و امنیت محسوب می‌شود.

به نظر شما بزرگ‌ترین نکته در مورد حملات در مقیاس بزرگ مثل سونی چیست؟

اشنیر: مهم‌ترین نکته این است که ما همگی در برابر این نوع حملات آسیب‌پذیر هستیم. مهم نیست چه نوع هک‌هایی باشند: هک‌های دولتی-ملتی (سونی)، داخلی (hactivists (HB Gray Federal، Hacking Team یا NSA)، (US State Department) (عربستان سعودی)، همگی مستندات داخلی یک سازمان را در دیده و منتشر می‌کنند که خود این می‌تواند یک حمله مخرب باشد. ما باید در مورد این

تاکتیک بیشتر بی‌اندیشیم: کم‌تر «چگونه جلوگیری کنیم»-که هم اکنون در حال انجام است و موثر نیست- و بیشتر «چگونه واکنش نشان دهیم».

در حال حاضر صنعت چگونه با آن برخورد می‌کند؟

اشنیر: هر کسی امیدوار است که نفر بعدی آن‌ها نباشند

مهم‌ترین روشی که سازمان‌ها با آن می‌توانند فعالیت‌های امنیتی خود را بهبود بخشند چیست؟

اشنیر: امنیت ترکیبی از جلوگیری، تشخیص و واکنش است. در حال حاضر، واکنش در بدترین وضعیت ممکن در بین این سه عنصر قرار دارد و سازمان‌ها نیازمند بیشترین میزان بهبود در این زمینه هستند. به عبارت ساده‌تر، ما باید در زمینه پاسخ و واکنش به رخدادهای امنیتی ارتقا پیدا کنیم. ما باید هوشمندانه‌تر، سریع‌تر و کارا تر شویم. ما باید پاسخ به رخدادهای حوزه فناوری اطلاعات را با مدیریت بحران سازمان یکپارچه کنیم. ما باید قادر به تشخیص این که چه اتفاقی در سازمان ما در حال حادث شدن است و نحوه برخورد با آن باشیم و باید به نحوی این کار را انجام دهیم که به عنوان یک سازمان، انعطاف‌پذیر باشیم.

چگونه می‌توانیم مسائل امنیتی را در مقیاس جهانی برطرف کنیم؟

اشنیر: اگر من جواب این سوال را میدانستم الان در حال پیاده‌سازی آن بودم. مسائل بین‌المللی بسیار پیچیده هستند و مسائل سایبری مستثنی نیستند. جاسوسی یک مسئله جهانی است. جرایم سایبری، جهانی هستند. نظارت حقوقی شرکت‌های بزرگ یک مسئله جهانی است. در سال‌های آتی این مسائل بسیار بزرگ‌تر و حادث‌تر خواهند شد. ■



| داستان علمی و تخیلی |

روزی که مدیر سیستم‌ها بر زمین حکم راندند | ۹۲ |



روزی که مدیر سیستم‌ها بر زمین حکم راندند

از خونه درست بشه». البته اشتباه می‌کرد. چیزها بارها از خانه درست شده بودند اما چون این کار بی‌سر و صدا و سریع اتفاق می‌افتاد، کسی آن را به یاد نمی‌آورد. البته شاید هم درست می‌گفت! لاگ‌ها نشان می‌دادند که مشکلاتی که بعد از یک نصفه شب اتفاق می‌افتادند همیشه نیازمند این بودند که فلیکس لباس بپوشد، از خانه بیرون برود، تا شرکت رانندگی کند و مساله را از نزدیک بررسی کند. اسمش را گذاشته بودند قانون انحراف بی‌نهایت جهان یا همان قانون فلیکس.

پنج دقیقه بعد فلیکس پشت فرمان بود. نتوانسته بود مساله را از خانه حل کند. شبکه مستقل روترها هم قطع بود. آخرین باری که این اتفاق افتاد ماجرا مربوط به احمقی بود یک بیل مکانیکی را به کابل اصلی برقی زده بود که دیتاسنتر اصلی سیستم‌های شرکت فلیکس بود. آن ماجرا باعث شده بود او و

رو راه بندازی. نمی‌شه تلفن رو جواب ندی؟ تلفن هنوز زنگ می‌زد. می‌دانست که حق با همسرش است. تلفن را پیدا کرد و جواب داد. صدایی مکانیکی اعلام می‌کرد «روتر اصلی جواب نمی‌دهد. بی‌جی‌پی جواب نمی‌دهد». صدای مکانیکی ای‌وی‌آر با این که او فحش بدهد تا غر بزند هیچ مشکلی نداشت پس حداقل نظرش را اعلام کرد و این کار حس بهتری به او داد.

– شاید هم بتونم از همین‌جا درستش کنم. می‌توانست از خانه به سیستم‌های برق اضطراری لاگین کند و روترها را ریپوت کند. یوپی‌اس‌ها در شبکه مستقلی بودند که روترهای خودشان را داشتند که به منابع برق بی‌وقفه وصل شده بودند. جودی حالا در تخت نشسته بود؛ مانند شبیهی تاریک در جلوی بخش بالایی تخت. «توی این پنج سال زندگی من حتی ندیدم یه بار هم چیزی

وقتی تلفن مخصوص فلیکس ساعت دو صبح زنگ زد، همسرش جودی غلطی زد و آرام با مشت به شانه او کوبید و گفت «چرا این لعنتی رو قبل این که بخوابیم خاموش نکردی؟» جواب داد «چون آنکال هستم».

روی لبه تخت نشسته بود که جودی با غر و لند گفت «اما تو که دکتر نیستی!» و لگدی هم به سمتش فرستاد. داشت شلوارش را که از روی زمین برداشته بود چپ و راست می‌کرد تا جیبی که موبایل در آن بود را پیدا کند. «آخه مگه تو مدیر سیستم نیستی؟»

– هستم. این هم بخشی از شغل منه.

– ببین دارن مثل خر از تو کار می‌کشن. می‌دونی که حق با منه! آخه چی بگم؟ الان پدری و نمی‌شه که هر وقت نصف شب یکی نتونست اون ویدئوهای لعنتی‌اش رو نگاه کنه، تو بیدار بشی بری استریمش



جادی امیر میرزایی
مترجم



کوروش دکترو
نویسنده

تقریباً بیست و پنج مدیر سیستم دیگر یک هفته دائماً به کارگاه سر زده بودند تا بالاخره شرکت سرویس‌دهنده و کارگرهای برون‌سپاری شده‌اش توانسته بودند ده هزار سیم قطع شده را دوباره به هم وصل کنند.

موبایلش دو بار دیگر در مسیر زنگ زد تا صدای مکانیکی به روی بلندگوهای بلوتوث منتقل شود و قطعی‌های بیشتری را در قطعات حساس‌تری اعلام کند. یک بار هم جودی زنگ زد. فلیکس گفت:

– سلام

– شاکا نباش. لحن شاکیه

خندیدن و دوباره گفت:

– لطفاً چک کن که توش شاکا بازی نباشه: سلام همسرش جواب داد:

– عاشقتم فلیکس.

– منم عاشقتم. حسای. حالا برو تو تخت و بخواب. – ورژن ۲ بیدار شده!

همیشه این شوخی را می‌کردند. وقتی فرزندانش هنوز در رحم مادر بود می‌گفتند که نسخه بتا است. وقتی کیسه آب پاره شده بود و به دفتر زنگ زده بودند تا خودش را به خانه و بیمارستان برساند همکارش داد کشیده بود «مستر آماده نصب!» و بعد از این‌که اولین گریه‌اش تمام شد تصمیم گرفتند که فرزندشان به نسخه ۲ رسیده.

تقریباً به دیتاسنتر رسیده بود. گفت «ببخشید که باعث شدم بیدار بشین». خوشبختانه ساعت ۲ صبح از ترافیک خبری نبود. خودرو را کند کرد و به خروجی پیچید و در ورودی پارکینگ ایستاد. نمی‌خواست تماس جودی با رفتن به پارکینگ زیرزمینی قطع شود. در گوشی شنید که:

– مساله بیدار شدن نیست. تو هفت ساله این‌جا کار کردی و سه نفر دارن به تو ریپورت می‌دن. گوشی رو بده به اونها. به اندازه کافی زحمت کشیده‌ای. جواب داد:

– من دوست ندارم از کارمندهام تسک‌هایی رو بخوام که خودم انجامشون نمی‌دم.

– تو اینکارها رو کرده‌ای! خواهش می‌کنم ازت.. من دوست ندارم شب‌ها تنها باشم.

– جودی..

– من اصلاً عصبانی نیستم. فقط جات خالیه. تو که هستی من خیلی خوب می‌خواهم!

– باشه.

– به همین سادگی؟

– بله. به همین سادگی. من دوست ندارم بد

بخوابی و به اندازه کافی هم شب‌ها کار کرده‌ام. از حالا فقط تعطیلات آخر هفته‌ها آنکال می‌شم.

جودی خندید و گفت که «سیستم‌ادمین‌ها که تعطیلات آخر هفته ندارن».

– من خواهم داشت. قول می‌دم.

– عالی. ممنونم.. اوه... ورژن دو روی حوله‌ام کثیف کاری کرد!

– پسر منه دیگه هههه

– بله واقعا هست

و تلفن را قطع کرد. فلیکس خودرو را به حرکت درآورد و پس از توقف کوتاهی جلوی باجه نگهبانی و خیره شدن به داخل اسکنر قرینه چشم به اندازه‌ای که مطمئن شود دستگاه توانسته چشم خوابالودش را اسکن کند، وارد تونل منتهی به پارکینگ شد.

کنار دستگاه فروش خودکار ایستاد و یک شکولات پاوربار و یک قهوه گرفت. قهوه در لیوان ضد ریزش و دارای نی تحویل داده شده؛ لیوانی قابل بردن به اتاق سرورها. پاوربار که شبیه بیسکویت ولی حاوی مواد مغذی بود را بلعید و جرعه‌ای هم از قهوه نوشید و اجازه داد در ورودی داخلی برای یک لحظه اثر انگشت و ابعاد بدنش را بخواند. در با صدایی خفه باز شد و مثل همیشه هوای خشک دور بدنش را فرا گرفت. بالاخره به داخل اتاق سرورها رسیده بود.

دیوانه خانه‌ای بود. رک‌ها طوری چیده شده بودند که حداکثر یکی دو مدیر سیستم بتوانند لابلاشان حرکت کنند و تمام فضای ارزشمند باقی مانده به رک‌های در حال وز وز اختصاص یافته بود. اما حالا در فضایی به این کوچکی حداقل بیست مدیر سیستم در تقلا بودند. در نگاه اول آدم فکر می‌کرد وارد گردهمایی یک گنگ سیاه پوش شده که علامت مشخصه‌شان تی‌شرت‌های نوشته‌دار و شکم‌های بزرگ و تلفن همراه و ابزارهای چندکاره است.

در حالت عادی همه باید در این سالن در حال یخ زدن باشند اما این همه آدم زنده این اتاق کوچک و بدون محفظه را گرم کرده است. پنج شش نفری با وارد شدن او نگاهی به وی انداختند. دو نفر به اسم سلام کردند و او با جواب دادن سعی کرد شکمش را جمع کند و از لابلائی آدم‌ها و رک‌ها رد شود و خودش را به بخش آردنت برساند که در انتهای سالن قرار داشت.

ون که حتی امشب آنکال هم نبود گفت «فلیکس!»

جواب داد: «تو این‌جا چکار می‌کنی؟ قرار نیست فردا هر دو خسته باشیم و در حال استراحت»

– من؟ اوه... ماشین شخصی خودم حوالی ساعت یک و نیم غیب شد. سیستم مانیتورینگ خبرم کرد. باید زنگ می‌زدم و می‌گفتم من میام این‌جا ولی فکر نمی‌کردم ماشین تو هم خراب شده باشه ماشین شخصی فلیکس که آن را با پنج مدیر سیستم دیگر شریک بود در طبقه پایین بود. با خودش فکر کرد نکند آن هم قطع شده باشد.

– ماجرا چی؟

– یک حمله فلش‌ورم گسترده! به عوضی با یه اکسیلویت روز صفر همه ماشین‌های ویندوز روی همه بلوک‌های آی‌پی رو نابود کرده؛ حتی ماشین‌های آی‌پی ورژن شش! همه سیسکوها بزرگمون روی ورژن شیش هستن و اگر بیشتر از ده تا پروب مونت کارلو بهشون برسه پایین می‌یان. در واقع الان همه پایین هستن. دی‌ان‌اس هم مشکل داره! انگار یکی دیشب زون ترنسفر رو آلوده کرده باشه. اوه و البته ایمیل و پیام شخصی‌ها هم هستن. یک بخش از حمله داره تقریباً به هر کسی که توی لیست دوستان هست پیام می‌فرسته که یک تروجان رو نصب و اجرا کنن، اونم با خوندن سابقه ایمیل و چت‌ها با اون آدم و درست کردن یک پیام قانع‌کننده الیزاتوری.

– خدایا!

وان یک مدیر سیستم تیپ ۲ بود. بیشتر از ۱۸۰ قد و موی بلند دم اسبی و سیب آدم برجسته. جواب داده «بله.. دقیقاً!»؛ روی تی‌شرتش نوشته بود «اسلحه‌ات رو انتخاب کن» و زیر آن عکس چندین تاس‌بازی‌های ایفای نقش پرینت شده بود. فلیکس تیپ ۱ بود؛ با اضافه وزن در ناحیه کمر و یک ریش کوتاه ولی منظم که غبغیش را می‌پوشاند. تی‌شرت فلیکس می‌گفت «سلام کاتولو» و مزین به یک کاتالوی مهربان و بدون دهن به سبک هلو کیتی بود. آن‌ها پانزده سالی بود که یکدیگر را می‌شناختند. در یوزنت آشنا شده بودند و بعد حضوری همدیگر را در یک جلسه آبجوخوری فرینت در تورنتو دیده بودند. بعدش هم یکی دو گردهمایی استراترک و در نهایت فلیکس به ون پیشنهاد داده بود که در آردنت با هم کار کنند. ون قابل اعتماد و منظم بود. درس مهندسی برق خوانده بود و یک دفتر سیمی داشت که تمام کارهایی که در مدیریت سیستم می‌کرد را قدم به قدم با تاریخ در آن می‌نوشت.



ون گفت «این بار حتی پیکاک هم نیست». منظورش Problem Exists Between Keyboard And Chair بود یا اشاره به این که مشکل از کاربر است - شکلی از مشکلات که اگر آدم‌ها عقل سلیم به خرج دهند پیش نمی‌آید - چیزهایی مثل تروجان‌ها. اما کرم‌هایی که مشغول خوردن روترهای سیسکو بودند مشکل مهندس‌های بی‌سواد بودند.

فلیکس گفت «نه پیکاک نیست. این مشکل مایکروسافت است! هر بار که من ساعت دو شب این‌جا بودم یا مشکل مایکروسافت خبیث بود یا مشکل پیکاک».

در نهایت روترهای مورددار را از اینترنت قطع کردند. البته این کار فلیکس نبود چون او دسترسی نداشت روتری را از اینترنت قطع، اینترفیس آی‌پی نسخه شش را خاموش و در نهایت کل دستگاه را ریست کند. دو اپراتور لعنت‌شده از جهنم بودند که باید به شکل همزمان کلیدشان را در قفل‌های قفسه سیسکوها فرو می‌کردند، می‌چرخاندند و به روترهایی که ۹۵٪ ترافیک راه دور کانادا از آن‌ها رد می‌شد دسترسی پیدا می‌کردند.

فلیکس و ون به راحتی ماشین‌های آدرنت را بالا آوردند. ماشین‌ها هنوز تحت حمله مداوم کرم‌ها بودند - روترها بخشی از جریان را بازگردانده بودند. در حال حاضر احتمالا هر ماشینی که در اینترنت بود زیر کرم‌ها غرق شده بود یا خودش مشغول ارسال کرم به جهان بود، شاید هم هر دو. فلیکس بعد از تحمل صد تایم‌اوت توانست به NIST و باگ‌ترک برسد و پچ‌های کرنل که انتظار می‌رسد فشار کرم‌ها را کمتر کنند دانلود کند و روی ماشین‌های تحت کنترل خودش نصب کند. ساعت ده صبح بود و آن قدر گرسنه بود که می‌توانست یک خرس را بر خورد ولی خوشحال بود که توانسته کرنل جدید را کمپایل کند و ماشین‌هایش را مقاوم کند. انگشت‌های بلند ون روی کیبورد مدیریتی می‌چرخید و زبانش وقتی داشت وضعیت لود سیستم‌ها را بررسی می‌کرد بیرون آمده بود.

ون ناراحت بود. گفت «من روی گریدو دوپیست روز آپتایم داشتم». گریدو قدیمی‌ترین سرور رک بود، تقریبا از همان زمانی که اسم رک را به افتخار کاراکتر جنگ ستارگان نامگذاری کرده بودند. حالا اسم همه بر اساس شخصیت‌های اسمورفا بود. البته در اسمورفا هم تقریبا به آخر اسم‌ها رسیده

بودند و مشغول رفتن به سراغ شخصیت‌های مک‌دونالدند بودند که از لپ‌تاپ ون شروع شده بود. اسم لپ‌تاپ ون میور مک‌چیز بود.

فلیکس گفت «گریدو دوباره بر خواهد خواست! اما باورت نمی‌شه. من طبقه پایین یک چهارصد و هشتاد و شیش داشتم که پنج سال آپ‌تایم‌شه! قلبم می‌شکند اگه ریپوت شده باشه»

- چهارصد و هشتاد و شش؟! باه‌اش چیکار می‌کردی آخه؟

- هیچچی. ولی خب کی یه کامپیوتر با پنج سال آپتایم رو شات دان می‌کنه؟ مثل اینه که مادر بزرگت رو بخشی چون دیگه کار مفیدی نمی‌کنه.

ون گفت که گرسنه است و باید غذا بخورند. - خب پس اول ماشین شخصی تو رو بالا میاریم و بعد مال من رو و بعدش یک پیتزا به عنوان صبحانه مهمون منی و بقیه روز رو هم نمیای سر کار. - قبوله! و رییس تو با ما خیلی خوبی!

ون گفت «تلفنت داره زنگ می‌زنه». فلیکس دستش را از کیس ۴۸۶ بیرون آورد. سیستم در مقابل روشن شدن مقاومت می‌کرد و حالا فلیکس سعی می‌کرد یک پاور جدید که از افراد طبقه بالا قرض گرفته بود را داخل کیس ببندد. ون گوشی را به دست فلیکس داد. گفت «سلام جودی!»، صداهای عجیبی در پشت خط شنیده می‌شد. شاید نویز استاتیک. شاید هم نسخه ۲ داشت در حمام دست و پا می‌زد. «جودی؟». خط قطع شد. سعی کرد دوباره زنگ بزند اما نه صدای زنگ شنیده شد نه صدای پیام‌گیر. بعد از چند لحظه تلفن تایم‌اوت داد و پیام «خطا در شبکه» روی صفحه نقش بست. گفت «لعنتی» ولی خشن نبود. تلفن را به کمر بندش وصل کرد. جودی می‌خواست بداند چه زمانی برمی‌گردد یا شاید هم لازم داشت در راه برگشت چیزی بخرد. احتمالا پیام می‌گذاشت.

هنوز مشغول سر و کله زدن با پاور جدید بود که تلفنش دوباره زنگ زد. جواب داد و گفت «هی جودی! وضعیت چیه؟». سعی می‌کرد در صدایش هیچ شکلی از غر زدن یا ناراضی نباشد. احساس گناه می‌کرد: از نظر فنی کار او بعد از بالا آمدن شبکه آدرنت تمام شده بود ولی هنوز هم آن‌جا بود. سه سال قبل مشغول کارهای شخصی بود

- البته به هر حال این ساعت‌ها هم در تایم‌شیت حساب می‌شدند و پولش را از کمپانی می‌گرفت. از تلفن صدای هق هق می‌آمد.

«جودی؟» حس می‌کرد خون بدنش تمام شده و انگشتانش یخ زده‌اند.

صدای جودی ناواضح بود «فلیکس... اون دیگه نفس نمی‌کشه. خدای من. اون مرده!»

- کی؟! کی مرده جودی؟

- ویل

ویل؟ ویل دیگر چه... روی زانوهایش خم شد. ویلیام اسمی بود که در سند تولد نوشته بودند و فراموش کرده بودند. حالا نسخه ۲ نامیده می‌شد. فلیکس ناله‌ای کرد. صدایی شبیه یک سگ بیمار.

- من حالم بده... حتی دیگه نمی‌تونم رو پام بایستم... فلیکس... من عاشقت هستم.

- جودی؟! چه اتفاقی افتاده؟

- همه، همه! فقط دو تا کانال مونده. خدایا! فلیکس. انگار از پنجره به مرگ خیره شدیم.

صدای استفراف از تلفن به مغزش هجوم برد. ارتباط داشت قطع می‌شد و حتی صدای استفراف را هم به سختی می‌شد تشخیص داد. صدا در تلفن می‌چرخید و حالا چیزی به جز صدای خودش نمی‌شنید.

به سمت خط قطع شده فریاد کشید «همونجا باش جودی!» و شماره ۹۱۱ را گرفت اما دوباره پیام «خطا در شبکه» روی گوشی نمایان شد. میور مک‌چیز را از ون گرفت و کابل شبکه ۴۸۶ را به آن زد و فایرفاکس را از کامندلاین اجرا کرد و به دنبال پلیس شهری گوگل کرد. دنبال تماس با ما بود. فلیکس هیچ‌وقت کنترلش را از دست نمی‌داد. کارش حل مساله بود و می‌دانست که وحشت کردن هیچ مساله‌ای را حل نمی‌کند.

یک فرم تماس با ما پیدا کرده بود و مکالمه‌اش با جودی را درست مثل وقتی که مشغول ریپورت کردن یک باگ است در آن وارد کرد. انگشتانش سریع بودند و توضیحاتش دقیق و دگمه *ارسال* را فشار داد.

ون از بالای سرش به صفحه نگاه کرد. گفت «فلیکس..»

فلیکس گفت «خدا!» روی زمین نشسته بود. سعی کرد بلند شود. ون لپ‌تاپ را گرفت و سعی کرد سایت‌های خبری را باز کند اما همه سایت‌ها تایم‌اوت می‌دادند. نمی‌شد گفت مساله به خاطر یک اتفاق عجیب است که همه به سایت‌های



خبری هجوم برده‌اند یا کار کرمی که اینترنت را فلج کرده.

فلیکس گفت «باید برسم به خونه».

و ون گفت که رانندگی می‌کند و او را می‌رساند تا بتواند دائما شماره همسرش را بگیرد.

به نزدیکی آسانسورها رسیده بودند. یکی از معدود پنجره‌های ساختمان کنار آسانسور بود. یک پنجره دایره‌ای با شیشه‌ای بسیار ضخیم. همان طور که منتظر آسانسور بودند از آن به بیرون نگاه کردند. به عنوان یک چهارشنبه، ترافیک چندانی به چشم نمی‌خورد. آیا پلیس‌ها بیشتر از معمول بودند؟

ون اشاره کرد و گفت «اووه آسمان‌های بالای سر!» برج سی‌ان از این پنجره دیده می‌شد. یک برج مخابراتی عظیم که در شرق آسمان را می‌شکافت. برج کج بود، مثل شاخه‌ای که در شن خیس فرو شده باشد. آیا واقعا برج داشت حرکت می‌کرد؟ بله! برج در حال پایین افتادن بود. آرام اما در حال سرعت گرفتن. برج داشت روی منطقه تجاری

سقوط می‌کرد. یک ثانیه بعد برج از نقطه بحرانی عبور کرد و با خارج شدن مرکز ثقلش از محدوده پایه برج، ناگهان فرو افتاد. شوک را حس کردند و بعد صدایش را هم شنیدند. کل ساختمان از این سقوط به خود لرزید. از خرابه‌ها گرد و خاک بلند شده بود و همان طور که بزرگ‌ترین سازه عمودی جهان با شکستن ساختمان‌ها داشت روی زمین آرام می‌گرفت، صداها هم بیشتر می‌شدند. ون گفت «مرکز مخابرات هم داره می‌ریزه!». نوبت برج‌های شبکه سی‌بی‌سی. بود که به آرامی منحرف شوند. مردم از هر طرف در حال دویدن بودند و آوار بر سر آن‌ها فرو می‌ریخت. اگر لحظه‌ای تخیلشان را به کار می‌انداختند می‌توانستند تصور کنند که از این پنجره دایره‌ای شکل در حال نگاه کردن به یک تصویر ساخته شده با کامپیوتر فوق العاده هستند که از تورنت دانلود شده. حالا مدیرسیستم‌ها دیگر هم پشت پنجره جمع شده بودند و به این سقوط مداوم نگاه می‌کردند.

یکی از آن‌ها پرسید «چی شده؟» فلیکس گفت «برج سی‌ان سقوط کرد». صدایش از دوردست‌ها به گوش می‌رسید.

- به این ویروس مربوطه؟

فلیکس گفت «به این کرم؟ چی میگی؟!» و به مرد که مدیر سیستم جوانی بود تازه چربی تیپ ۲ در اطراف شکمش در حال جمع شدن بود چشم دوخت.

- نه کرم نه! من یک ایمیل گرفتم که می‌گفت شهر به خاطر یک ویروس قرنطینه شده. احتمالا یک سلاح بیولوژیک.

مرد بلک‌بری اش را به فلیکس داد. فلیکس آن قدر غرق ایمیل فوروارده شده از مرکز سلامت کانادا بود که حتی متوجه خاموش شدن تمام چراغ‌های سالن هم نشد. بعد فهمید. بلک‌بری را به صاحبش پس داد و بغض کرد.

یک دقیقه بعد ژنراتورها روشن شده بودند. مدیرسیستم‌ها به طرف پله‌ها دویدند اما فلیکس



بازوی ون را گرفت و او را نگاه داشت.

- به نظرم بهتره ما همینجا منتظر بمونیم

- اما جودی چی می‌شه؟

فلیکس حس کرد دارد بالا می‌آورد.

خودش را جمع کرد و گفت «باید بریم تو اتاق

سرورها». اتاق سرورها سیستم تهویه‌ای داشت

که هوا از هر غبار و میکروارگانیزمی پاک می‌کرد.

بر خلاف اکثریت به سمت پله‌هایی رفتند که به

بالا می‌رفت تا خودشان را به بزرگترین اتاق سرور

برسانند. فلیکس در را باز کرد و اجازه داد پشت

سرش بسته شود.

- فلیکس باید بری خونه

- این یک سلاح بیولوژیکه. سوپرباگ! این‌جا سالم

می‌مونیم. حداقل تا وقتی فیلترها کار می‌کنن.

- چی؟

- برو ای آر سی.

همین کار را کردند. ون پشت میور مک‌چیز بود

و فلیکس از اسمورفت استفاده می‌کرد. به چند

کانال چت سر زدند تا بالاخره به جایی رسیدند که

چندین اسم آشنا مشغول صحبت بودند.

پنتاگون رفته / کاخ سفید هم همین طور

تو سن دیگو هستیم. همسایه‌هام رو بالکن دارن

خون استفراغ می‌کنن

توی گرکین یکی تو خیابون مرده، بانک‌دارها دارن

مثل موش فرار می‌کنن

یکی بهم گفته که گینزا آتیش گرفته

فلیکس نوشت: در تورنتو هستیم. خودمون دیدیم

که برج سی‌ان سقوط کرد. در مورد سلاح‌های

بیولوژیک حرف می‌زنن، یک چیز با اثر خیلی

سریع.

ون که نوشته را خوانده بود تذکر داد که نمی‌دانند

این مساله چقدر سریع عمل می‌کند و ممکن است

همگی مثلاً سه روز قبل در معرض آن قرار گرفته

باشند. فلیکس جواب داد «اگر این طور بود الان ما

هم به چیزهایی حس می‌کردیم».

به نظر می‌رسه یک پالس قوی الکترومغناطیسی

کل هنگ‌کنگ رو گرفته.. شایدم پاریس! هیچ

سیگنال شبکه از انجاها نداریم. کل نت بلوک اونها

کاملاً تاریکه!

تورنتو هستین؟

این شناسه را نمی‌شناختند.

بله. در خیابون فرانت

خواهر من دانشگاه تورنتو است. نمی‌تونم ازش

خبر بگیرم. می‌تونین به زنگ بهش زنین؟

هیچ تلفنی کار نمی‌کنه

فلیکس دوباره به گوشی‌اش نگاه کرد که عبارت

«خطا در شبکه» از رویش محو نمی‌شد.

ون با هیجان گفت «من یک سافتفون روی میور مک‌چیز دارم!» و برنامه ویپ را اجرا کرد و ادامه داد «اصلاً یادم نبود».

فلیکس لپ‌تاپ را قاپید و شماره خانه‌اش را در برنامه تایپ کرد و انتر را فشار داد. تلفن یکبار زنگ زد و بعد تنها چیزی که به گوش می‌رسید یک جیغ ممتد بود. چیزی مانند صدای آمبولانس در فیلم‌های ایتالیایی.

در آی‌آرسی تایپ کرد:

تلفن خط نمی‌ده

و در بالای سرش به ون نگاه کرد و دید که شانه‌های لاغرشی می‌لرزند. ون گفت «لعنت تو روح خبیث! دنیا داره تموم می‌شه».

یک ساعت بود که در آی‌آرسی بودند. آتلانتا کاملاً سوخته بود. منهن با حجمی از رادیواکتیو پر شده بود که وبکم‌های نصب شده در لینکلن پلازا دیگر هیچ چیزی مخابره نمی‌کردند. بعضی‌ها مسلمانان تندرو را متهم به عملیات تروریستی می‌کردند اما بعد که مشخص شد به عربستان و نقاط مقدس هم حمله شده، این فرضیه رد شد. دست‌های فلیکس می‌لرزید و ون به آرامی در گوشه اتاق اشک می‌ریخت. دوباره سعی کرد به خانه تلفن کند و بعد هم به پلیس. هیچ فایده‌ای نداشت. درست مانند ۲۰ بار آخری که امتحان کرده بود.

به ماشین طبقه پایین ssh کرد و ایمیل‌هایش را گرفت. اسپم، اسپم، اسپم و باز هم اسپم. ایمیل‌های اتوماتیک. آهان! یک ایمیل از طرف سیستم خودکار تشخیص نفوذ به شبکه آرذنت.

ایمیل را باز کرد و سریع خواند. یک نفر با خشونت و به شکل مستمر مشغول چک کردن پورت‌های روترهایش بود. این کار امضای هیچ کرمی را نداشت. آی‌پی را تریس کرد و کشف کرد که حمله از طرف ماشینی در همین ساختمان است - به شکل دقیق‌تر از ماشینی در طبقه پایین.

قدم‌های مواجهه با چنین حمله‌ای از قبل مشخص بود. آی‌پی دستگاه حمله‌کننده را پورت اسکن کرد و مشخص شد که پورت ۱۳۳۷ باز است. این عدد در کارکترهای زبانی/عددی هکرها leet یا elite خوانده می‌شود. این پورت شبیه پورت‌هایی است که کرم‌ها باز می‌گذارند تا بعداً در صورت نیاز از آن داخل شوند. در گوگل به دنبال اسپلویت‌هایی جستجو کرد که از پورت ۱۳۳۷ استفاده می‌کنند و با اضافه کردن اثرانگشت سیستم‌عاملی که این پورت روی آن باز

بود به نتیجه رسید. این یک کرم بسیار قدیمی بود. کرمی که احتمالاً هر سیستم‌عاملی سال‌ها قبل علیه آن واکنش داده است. مشکوک بود. کلاینت را دانلود کرد و به ماشین مورد نظر وصل شد و برای خودش یک اکانت روت ساخت. حالا وقتش بود به اطراف نگاهی بیندازد.

یک کاربر دیگر نیز در این سیستم لاگین بود. کاربری به اسم اسکاردی. با دستور تاپ متوجه شد که همین کاربر مسئولیت اجرا کردن صدها پروسه برای چک کردن پورت‌های دستگاه او و تعداد زیادی دستگاه دیگر را بر عهده دارد. یک چت باز کرد و نوشت:

ماشین منو ول کن

انتظار توهین، معذرت خواهی یا اصولاً نفی حمله را داشت اما با جواب سورپریز شد:

شما در دیتاسنتر فرانت استریت هستین؟

بله

خدای من. فکر می‌کردم تنها کسی هستم که زنده مونده. من توی طبقه چهارم. فکر کنم یک حمله بیولوژیک شده. نمی‌خوام از این اتاق تمیز بیرون برم. فلیکس نفس بلندی کشید و نوشت:

- به ماشینم حمله می‌کردی که من پیدات کنم؟

- آره

- هوشمندانه بوده.

ادمین باهوش خبیث!

ما در طبقه ششم هستیم. دو نفریم.

خبری دارین؟

فلیکس لاگ ای‌آرسی را برایش کپی پیست کرد و منتظر ماند تا طرف مقابل بتواند همه آن را بخواند. ون سر پا بلند شده بود و به اطراف سرک می‌کشید.

- ون؟ رفیق چی شده؟

- باید برم دستشویی.

- در رو باز نکن! یه بطری نوشابه خالی تو سطل اشغال دیدم.

- آره ...

ون مثل یک زامبی به سمت سطل اشغال رفت و بطری نوشابه خانواده را از آن بیرون کشید و پشتش را به فلیکس کرد.

اسم من فلیکس است.

ویل

معه فلیکس تیر کشید. باز داشت به نسخه ۲ فکر می‌کرد.

ون گفت «فلیکس فکر می‌کنم باید برم بیرون» و به سمت در محفظه هوا راه افتاد. فلیکس کیبورد

را به زمین گذاشت و روی پاهایش ایستاد و در تلاش برای حفظ تعادل، به سمت ون دوید و قبل از رسیدن به در او را متوقف کرد.

به چشمان دوستش زل زد. چشمان ون به جای خاصی نگاه نمی‌کردند. گفت «ون! ون! به من نگاه کن»

ون گفت «باید برم. باید برم خونه و به گربه‌هام غذا بدم»

- اون بیرون یک چیزی هست. یک چیز کشنده و خیلی هم سریع. شاید با باد پخش می‌شه. شایدم تا حالا رفته باشه. اما ما تا جایی که بتونیم این‌جا می‌مونیم و فقط وقتی خارج می‌شیم که دیگه هیچ جوری نتونیم این‌جا بمونیم. بشین ون! بشین.

- سردمه فلیکس

واقعاً هم سرد بود. دست‌های فلیکس از سرما مثل پوست مرغ دانه‌دانه شده بودند و پاهایش شبیه قطعات یخ بی‌حس بودند. کلک همیشگی مدیرسیستم‌ها را مطرح کرد: «بیا بریم کنار یک رک داغ. اگر کامپیوتر داخل رک کرم داشته باشه، سی‌پی‌یو داغه و حسایی گرم می‌شیم».

هنوز اونجایی؟

بله. اینجاییم و داریم سعی می‌کنیم یخ‌زننیم

چقدر باید صبر کنیم؟

هیچ ایده‌ای نداریم.

تا مدت‌ها کسی چیزی ننوشت.

فلیکس مجبور شده بود دوبار از شیشه نوشابه خانواده استفاده کند. بعد ون از آن استفاده کرد. فلیکس دوباره تلاش کرد به جودی زنگ بزند و سایت پلیس هم هنوز داون بود.

در نهایت کنار رک گرم نشست، با دست‌ها زانوهایش را بغل کرد و مثل بچه‌ها اشک ریخت.

یک دقیقه بعد ون کنارش نشست و دست‌هایش را دور شانه‌هایش حلقه کرد.

فلیکس گفت «ون. همه‌شون مرده‌اند. جودی و پسر. خانواده‌ام مردن».

- مطمئن نیستیم.

فلیکس غرید «به اندازه کافی مطمئن هستیم. همه چیز تموم شده. نشده؟»

- چند ساعت دیگه هم صبر می‌کنیم و بعد می‌زنیم بیرون. تا اون وقت همه چی درست شده. آشنشانی مشکلات رو حل می‌کنه. ارتش وارد عمل شده. درست می‌شه.

قفسه سینه فلیکس تیر کشید. از تولد نسخه ۲ به بعد گریه نکرده بود. محکم‌تر زانوهایش را بغل کرد.



احتمالا یک گاز. تنفس هوای آلوده افراد را آلوده می‌کند. از پنج صبح امروز هیچ کدام از درهای این ساختمان باز نشده و تا وقتی من تایید نکنم، هیچ کس هیچ دری را باز نخواهد کرد.

«حمله‌ها به شهرهای بزرگ در همه جهان سیستم‌ها و نیروهای اضطراری را فلج کرده. حمله‌ها الکترونیکی، بیولوژیکی، هسته و انفجارهای سنتی بوده‌اند و بسیار گسترده. من یک مهندس امنیت هستم و این الگوی حمله آشناست: گروه ب یک پل را منفجر می‌کنند چون همه سیستم‌های محافظتی در گیر حمله اتمی گروه آ هستند. هوشمندانه است. یک هسته اوم شین ریکو در سنول ساعت ۲ بامداد اولین حمله را انجام داده ولی ما مطمئن هستیم که چنین گروهی نمی‌تواند مسوول این سطح از حملات باشد. آن‌ها هرگز به این سلاح‌ها دسترسی نداشته‌اند و آن سطح از سازمان‌دهی را هم ندارند که بتوانند در تمام جهان وارد عمل شوند. به عبارت دیگر به اندازه کافی باهوش نیستند.

«ما تا جایی که بتوانیم در این‌جا می‌مانیم، حداقل تا وقتی که سلاح بیولوژیک کشف و خنثی شود. ما تلاش می‌کنیم رک‌ها و سرورها روشن و بالا باشند. این‌جا زیرساخت‌های مهمی داریم و وظیفه ما حفظ

بود. هیچکس به چشم دیگری نگاه نمی‌کرد. فلیکس به این فکر کرد که کدام یک ویل هستند و در صف ماشین خودکار تحویل خوراکی ایستاد. دو انرژی‌بار و یک لیوان عظیم قهوه از دستگاه گرفت و پول خردهایش تمام شد. ون روی یکی از میزها برایشان جا گرفته بود. فلیکس یک انرژی‌بار و لیوان قهوه را روی میز گذاشت و با گفتن «یک کمی هم برای من نگاه دار» به سمت صف دستشویی رفت. وقتی خودشان را خلاص کرده بودند و در حال خوردن بودند، تاک نردی و دوستش به کافه وارد شدند. آن‌ها خرت و پرت‌های روی میز پذیرش را به کناری زدند و تاک نردی به بالای میز رفت و به جمعیت چشم دوخت. صحبت‌ها کم کم قطع شد و سکوت کافه را فرا گرفت.

«من یوری پوپویچ هستم و این دیگو روزنباوم. متشکرم که به این طبقه آمدید. این چیزی است که ما از آن مطمئن هستیم: برق ساختمان سه ساعت است که روی ژنراتور است. شواهد بصری می‌گویند ما تنها ساختمان روشن در تورنتو هستیم و احتمالا تا سه روز دیگر هم برق خواهیم داشت. پشت در این‌جا یک ماده بیولوژیک با منبع ناشناخته وجود دارد که سریع می‌کشد؛ در عرض چند ساعت.

و بعد درها باز شدند.

دو مدیر سیستم داشتند به داخل سرک می‌کشیدند. یکی از آن‌ها تی‌شرتی داشت که رویش نوشته بود «TALK NERDY TO ME» و نفر دوم تی‌شرت «مدافعان دیجیتال کانادا» به تن داشت.

گیکی حرف بز ن گفت: «زود پاشین بیاین. همه تو طبقه آخر جمع شدن. با پله بیاین».

فلیکس فهمید که چند ثانیه‌ای است نفس نکشیده. تاک نردی گفت: «اگر بیو توی ساختمون باشه، همه آلوده‌ایم. بیاین. اونجا می‌بینمتون». فلیکس روی پاهایش ایستاد و گفت «یکی هم توی طبقه ششم هست» و جواب شنید که «آره. اونو دیدیم. رفته بالا».

تاک نردی یکی از مدیر سیستم‌های جهانی بود که روترهای بزرگ را خاموش کرده بودند. فلیکس و ون به آرامی از پله‌ها بالا می‌رفتند و صدای قدم‌هایشان در راه‌پله می‌پیچید. بعد از چندین ساعت در اتاق سرورها بودن، هوای راه‌پله مانند سونا بود.

در طبقه آخر یک کافه داشتند و دستشویی‌هایی که به خوبی کار می‌کردند. همین طور آب و قهوه و ماشین خودکار تحویل خوراکی. صفی ناآرام از مدیر سیستم‌ها جلوی هر یک از این‌ها تشکیل شده

آپتایم «پنج نه» است (اشاره به بالا بودن نود و نه ممیز نهصد و نود و نه درصد مواقع سرورها). در مواقع بحران وظیفه ما چند برابر است» یکی از مدیر سیستم‌ها دستش را بالا برد. جوان بود و تی‌شرت سبز رنگ هالک به تن داشت.

– کی فوت شده که تو شاه شدی؟

– من کنترل همه سیستم‌های امنیتی رو دارم. همین‌طور کلید همه اتاق‌های سرور رو و رمز درهای خروجی رو که الان قفل هستن. همچنین من کسی هستم که همه شمارو این‌جا جمع کردم و جلسه رو ترتیب دادم. برام مهم نیست اگر یکی دیگه بخواد این مسوولیت رو قبول کنه. کار چرتی است. اما به هر حال یکی باید این کار رو بکنه.

– درست می‌گی. کار بی‌خودیه ولی یکی می‌تونه بکنه. من هم می‌تونم. اسم من ویل ساریو است. پوپویچ گفت «باشه. اگر بذاری حرف‌هام رو تموم کنم شاید آخرش همه کارها رو تقدیم جنابعالی کردیم».

ساریو پشت به جمعیت کرد و به سمت پنجره رفت و گفت «پس تموم کن» و به بیرون خیره شد. فلیکس خط مسیر او را دنبال کرد. در بیرون از پنجره ستون‌های دود چرب به هوا بلند بود. کمر سخنرانی پوپویچ شکسته بود. گفت «همین‌ها».

پسرک برگشت و به جمعیت نگاه کرد و گفت «پس حالا نوبت منه؟»

اکثر مدیر سیستم‌ها بدون تمسخر لبخند زدند. – من این‌طوری فکر می‌کنم که دنیا داره به گند کشیده می‌شه. حملات هماهنگی به زیرساخت‌های مهم بوده. فقط با یک روش می‌شه این قدر دقیق این همه حمله رو در کل جهان هماهنگ کرد: اینترنت. حتی اگر فکر کنیم حمله‌ها فرصت طلبانه بودن، باید به این فکر کنیم که چطور می‌شه این همه حمله فرصت طلبانه کرد؟ بازم جواب معلومه: اینترنت.

پوپویچ با تمسخر گفت «یعنی می‌گی باید اینترنت رو قطع کنیم؟» و در مقابل جواب ندادن ساریو، خاموش شد.

– ما دیشب یک حمله داشتیم که عملاً اینترنت رو قطع کرد. یه داس روی روترهای حیاتی و یک دی‌ان‌اس فو که مثل پنیر همه چیز رو پایین آورد. پلیس و ارتش همیشه از تکنولوژی فراری بودن و این احق‌ها تقریباً هیچ وقت از اینترنت استفاده نمی‌کنن. اگر اینترنت رو قطع کنیم بیشتر از

همه به ضرر حمله‌کننده‌ها می‌شه و تقریباً روی مدافعین اثری نداره. هر وقت وضع طبیعی شد دوباره وصلش می‌کنیم.»

پوپویچ با دهان باز فریاد زد «مهمل می‌گی!» ساریو گفت: «منطقی حرف می‌زنم. البته خیلی‌ها وقتی منطق منجر به اقدامات جدی می‌شه، ازش فرار می‌کنن. این مشکل آدم‌ها است نه مشکل منطق».

همه شروع به صحبت با کنار دستی کردند و زمزمه‌ها به سرعت تبدیل به یک همه‌مه و بعد غرش درهم شد. پوپویچ فریاد زد «خفه شنین!» و بحث‌ها یک وات پایین آمد. پوپویچ دوباره فریاد زد و پایش را به میز زیرپایش کوبید و سر و صداها کمی آرام شد. «در هر لحظه یک نفر!». قرمز شده بود ولی دست‌هایش هنوز در جیب‌هایش بود.

یک مدیر سیستم طرفدار ماندن بود، یکی طرفدار رفتن. باید در اتاق سرور پناه می‌گرفتند. باید همه خوراکی‌ها را یکجا جمع می‌کردند و یک نفر مسوول تقسیم می‌شد. باید بیرون می‌رفتند و با پلیس تماس می‌گرفتند یا در بیمارستان‌ها به شکل داوطلب کار می‌کردند. باید کسانی را به حفاظت از درهای ورودی می‌گماشتند. فلیکس از این تعجب کرد که دستش را بالای سرش در حال اجازه گرفتن می‌دید. پوپویچ گفت حرف بزنند.

گفت «اسم من فلیکس ترمونت است» و روی میز رفت و PDA را بیرون آورد. «می‌خوام براتون یه چیزی بخونم».

«حکومت‌های جهان صنعتی. ای درماندگان گوشت و آهن، من از سایبراسپیس آمده‌ام، از خانه جدید تعقل. من از طرف آینده از شما گذشتگان می‌خواهم که ما را ترک کنید. شما این‌جا خوش نیامده‌اید. شما در جهان ما حکمرانی نمی‌کنید.

ما حکومت منتخب نداریم و نمی‌خواهیم هم داشته باشیم پس من با قدرتی بالاتر از آنچه لیبرالیسم حکم می‌کند با شما سخن می‌گویم. در فضای اجتماعی جهانی‌ای که من اعلام می‌کنم، قدرت تحمیلی شما جایگاهی ندارد. شما حق اخلاقی ندارید تا بر من حکم برانید و دیگر روشی نیز برای ترساندن واقعی ما در دستانتان نیست.

حکومت‌ها قدرت خود را از پذیرش کسانی می‌گیرند که بر آن‌ها حکم می‌رانند. شما هرگز نظر ما را نپرسیده و از طرف ما پذیرفته نشده‌اید. ما هرگز شما را دعوت نکرده‌ایم. شما حتی ما را نمی‌شناسید، و دنیای ما را نیز. سایبراسپیس خارج

از مرزهای شماست. تصور نکنید می‌توانید همانند بقیه پروژه‌های عمرانی‌تان، سایبراسپیس بسازید. نمی‌توانید. این فضا نتیجه طبیعت است و خودش را از طریق اعمال جمعی ما شکل می‌دهد».

سالن ساکت بود. فلیکس گفت «این بخشی از اعلامیه استقلال سایبراسپیس است. این متن ۱۲ سال قبل نوشته شده. من فکر می‌کردم این یکی از زیباترین متن‌هایی است که خواننده‌ها و دوست داشتیم کودکانم در دنیایی بزرگ شوند که سایبراسپیس آزاد باشد و آزادی آن بر دنیای فیزیکی هم تأثیر بگذارد و گوشت‌اسپیس^۱ آزادتر شود».

آب دهانش را قورت داد و با پشت دست چشم‌هایش را پاک کرد. ون به شکلی غریب روی کفش‌هایش زد تا از او تشکر کرده باشد.

«پسر و همسر زیبایی من امروز مردند. میلیون‌ها نفر دیگر هم، شهر به معنی واقعی کلمه توی آتش می‌سوزه. شهرها از نقشه محو شدن».

سرفه‌ای کرد. اشکش را پاک کرد و ادامه داد: «توی همه دنیا مردمی مثل ما توی ساختمان‌های مثل این جمع شدن. همه داشتن سعی می‌کردن سرورها رو به اینترنت برگردونن و سیستم‌هاشون رو از دست اون کرم نجات بدن. ما غذا و آب و برق داریم. ما شبکه‌ای داریم که آدم‌های بد ازش این قدر خوب استفاده کردن و آدم‌های خوب چندان ازش سر در نیاوردن. ما همه علاقه مشترکی به آزادی داریم که ریشه‌ای عشق ورزیدن به شبکه اینترنته. ما مسوول این مهم‌ترین سازمان و ابزار هستیم که دنیا تا به حال به خودش دیده. ما در حال حاضر نزدیک‌ترین چیز به مفهوم دولت هستیم، ژنو از بین رفته، رود شرقی تو آتش می‌سوزه و سازمان ملل هم که تخلیه شده.

«جمهوری توزیع شده سایبراسپیس این طوفان رو بدون صدمه رد کرده. ما متولی این ماشین بدون مرگ، عظیم و شگفت‌آوریم. ماشینی که می‌تونه دنیا رو دوباره شکل بده. من به جز این چیزی برای زنده موندن ندارم».

چشم‌های ون پر از اشک شده بود. او تنها نبود. کسی او را تشویق نکرد اما اتفاق بهتری افتاد. همه با احترام سکوت کردند. سکوت کامل همراه با فکری که تا یک دقیقه به طول کشید.

پوپویچ گفت: «چطوری باید این کار رو بکنیم؟». در صدایش هیچ رگه‌ای از استهزا نبود. ■

ادامه داستان در شماره بعدی



Meat Splice .۱
که منظور دنیای واقعی ما
انسان‌هاست



برخی از امکانات سیستم PCC

- توزیع خودکار تماس ها (ACD)
- امکان تعریف صف های پاسخگویی به هر تعداد
- امکان تعریف وزن برای صف
- امکان تعریف وزن برای اپراتورها
- امکان تعریف منوی صوتی تعاملی (IVR) به صورت چند لایه ای
- امکان تعریف شرایط زمانی
- امکان تعریف کنفرانس تلفنی و ویدئویی
- امکان انتقال مکالمات (Call Transfer)
- امکان تعریف تعداد نامحدود داخلی
- امکان تعریف اپراتور به همراه نام کاربری و کلمه عبور برای ورود به صف
- امکان ورود به همه صف های دارای دسترسی با گرفتن کد ویژه
- امکان تعریف داخلی های دارای دسترسی به صورت ریموت (پاسخگویی از راه دور)
- امکان تعریف لیست سیاه شماره ها
- امکان محدود کردن ضبط مکالمات برای داخلی ها و یا صف های مورد نظر
- امکان ضبط کلیه مکالمات و نگهداری آنها
- امکان تعریف نظرسنجی
- امکان تعریف صندوق صوتی
- امکان هدایت تماس ها بر اساس شماره تماس گیرنده
- امکان ایجاد درخواست تلفنی و ثبت آن در سیستم پشتیبانی



ParsPooyesh
Fanavar

www.parspooyesh.com

info@parspooyesh.com

+982148056000

+982148056789

متنک

می توانید هر خبر یا مقاله ای را خلاصه کنید



<http://matnak.com>

متنک آماده شده شما برای آدرس:

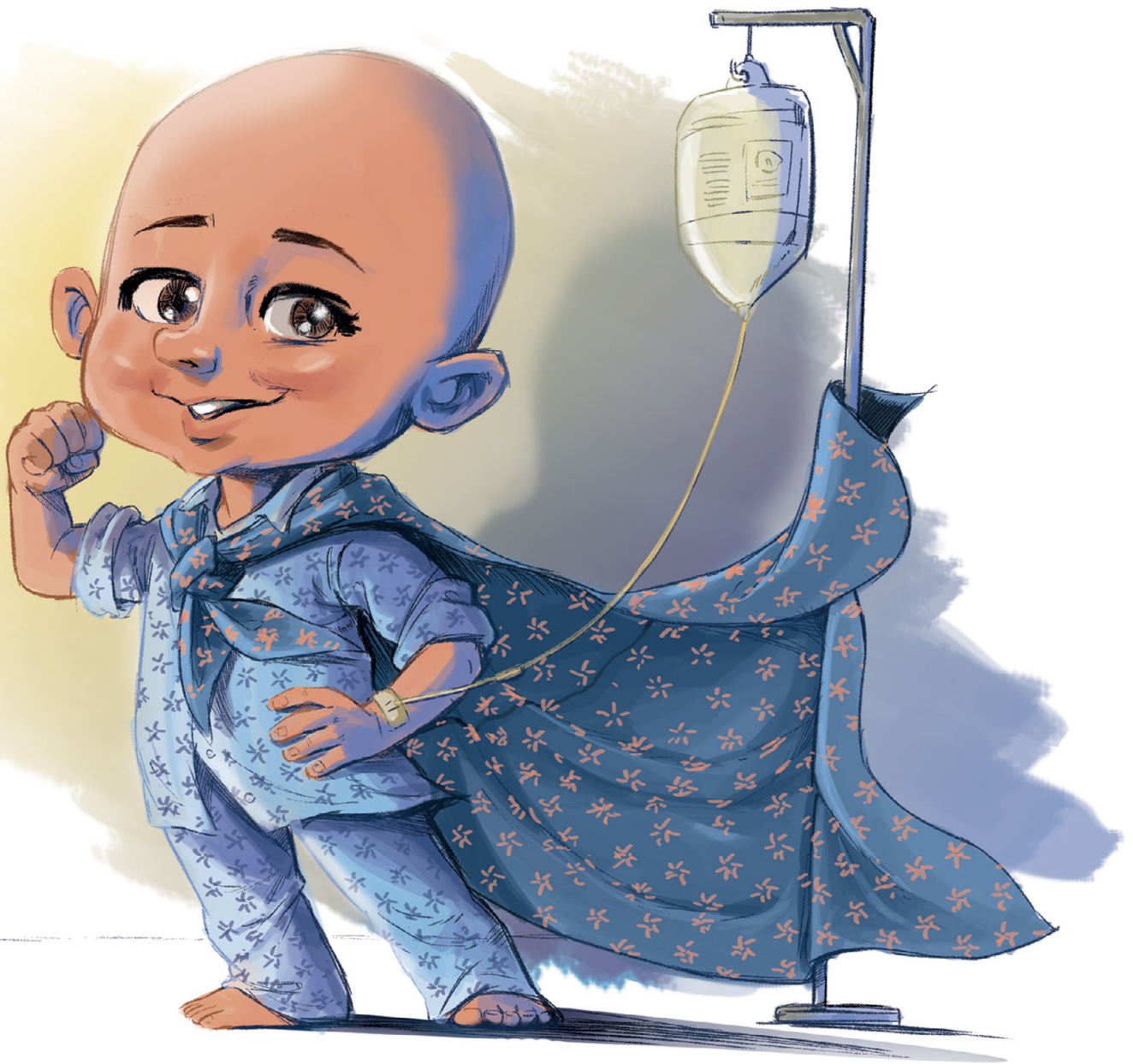
با توجه به حجم انبوه اطلاعات، دسترسی به داده های مطلوب گاهی بسیار پیچیده می شود. متنک خدمتی است که می توانید از آن برای سهولت دسترسی به اطلاعات مورد نظر استفاده کنید. از هر خبر منتشر شده، قسمت های مهم آن خبر را بیابید و طبقه بندی کنید و یا نتایج موجود در تحقیق و یا مقاله ای را جمع بندی نمایید. در نهایت شما قادر خواهید بود متنی را به عنوان ورودی به متنک ارائه دهید و آن را خلاصه کنید.



مسیر
راه کارهای بر پایه وب

روان ارتباط
ارتباط انسان و سیستم

ایران، تهران، میدان محسنی، بلوار میرداماد، جنب بانک آینده، ساختمان کامیار، طبقه سوم، واحد ۱۸
تلفن: (داخلی ۱۱۱) ۲۱ ۲۲۹۰۳۹۲۴ (+۹۸)
نمابر: ۲۱ ۲۲۲۹۷۵۶ (+۹۸)
www.ravanertebat.com
www.cvas.ir



سرطان را شکست دهیم

در این روزها که دل هایمان پر از امید است و هر روز خبر قهرمانی و پیروزی ایرانیان را می شنویم، کمک کنیم تا کودکان محک نیز سرطان را شکست دهند و قهرمانی خود را جشن بگیرند. برای پیوستن به خانواده بزرگ محک و آگاهی از روش های کمک به کودکان مبتلا به سرطان با ما تماس بگیرید.



۰۲۱ - ۲۳۵۴۰



۲۳۵۴۰ * ۷۲۰ *

شماره حساب بانک پارسیان: ۸۱۰۴۴۴۴۹

از اینکه به پیام ما توجه کردید، سپاسگزاریم.



محک

مؤسسه خیریه حمایت از
کودکان مبتلا به سرطان

mahak-charity.org