بایدها و نبایدهایی برای انتخاب راهکارهای ابری

اگر عضوی از یک تیم فنی و یا توسعه‌دهنده نرم‌افزار باشید قطعا به گوشتان رسیده است که در صورت استفاده از راهکارهای ابری، سرعت توسعه وانعطاف‌پذیری نرم‌افزار افزایش می‌یابد.

انتخاب بهترین راهکار های ابری و آگاهی نسبت به مزیت‌های هریک از آنها برای پوشش بیشترین نیازهای فنی پروژه‌ها، نیازمند اطلاعات و بررسی‌های جامع است. در این مقاله تلاش می‌کنیم تا مجموعه‌ای از بایدها و نبایدهای انتخاب این راهکارها را معرفی و بررسی کنیم:

انتخاب بین انواع ابرها

وقتی از سرویس‌های ابری صحبت می‌شود، عموما می‌توان به 3 سرویس اصلی زیر اشاره کرد:

infrastructure as a service)IaaS)

platform as a service)PaaS)

(software as a service)SaaS

(Infrastructure as a service) IaaS

در این نوع سرویس، فقط زیرساخت سخت‌افزاری برایتان تهیه می‌شود و در آن، تعدادی ماشین مجازی توسط ارائه‌دهنده سرویس، راه‌اندازی می‌شوند. از این نقطه به بعد نگهداری سرویس، نصب نرم‌افزار، دیتابیس‌ها، لایه وب و... برعهده تیم فنی خودتان است و ارائه دهنده‌ی اولیه، فقط مسئول نگهداری ماشین‌های مجازی و فعال نگهداشتن سیستم عامل‌هاست و مسئولیت بروزسانی سیستم‌عامل و ایجاد قوانین فایروال برعهده تیم فنی شماست.

مزایا

تقریبا همه جنبه‌های سیستم در اختیارتان است.

معایب

نیاز به دانش فنی یا تیم فنی برای نگهداری زیر سیستم‌ها خواهید داشت.

(Platform as a service (PaaS

در این مدل سرویس، توسعه‌دهندگان نرم‌افزار، آن را با این نگاه که باید روی زیرساخت ابری IaaS پیاده‌سازی شود، تولید می‌کنند. تقریبا توسعه‌دهندگان به هر چیزی که لازم دارند بجز زیرساخت اجرای نرم‌افزار و فایل‌های ذخیره شده روی سرورها دسترسی دارند. معمولا در این مدل، نگهداری از دیتابیس‌ها و مولفه‌های سیستم عاملی به عهده‌ی

ارائه دهنده‌ی سرویس است و برنامه‌نویسان می‌توانند تمرکز بیشتری روی کدنویسی نرم‌افزار داشته باشند.
در این مدل، شما به لایه‌ی سیستم عامل دسترسی ندارید و با محدودیت‌هایی برای تغییرات در این لایه و تغییر Runtimeها و پورت‌های فایروال مواجه خواهید بود.
افزایش مقایس‌پذیری نرم‌افزار از دیگر مزایای استفاده از این مدل سرویس است.

مزایا

به شما یک سیستم کاملا ایزولو می‌دهد که توسط تیمی حرفه‌ای مدیریت و نگهداری می‌شود و اجازه می‌دهد که با تمرکز بیشتری به توسعه نرم‌افزار بپردازید.

معایب

معمولا فرآیند انتشار نسخه‌های جدید، محدود به نسخه‌های Major/Minor است؛ به طوریکه شرکت ارائه‌دهنده زیرساخت بتواند یکپارچگی نرم‌افزار را روی همه‌ی سرورها حفظ کند.

(Software as a service (SaaS

در این مدل، تقریبا همه چیز مانند سرور، سیستم عامل، نرم‌افزار و ... توسط خود ارائه‌دهنده، تامین می‌شود. نگهداری همه‌ی راهکار به صورت یکپارچه توسط ارائه‌دهنده نرم‌افزار، انجام می‌شود. در این مدل به هر آن چیزی که در نرم‌افزار اصلی طراحی شده، دسترسی دارید؛ اما به لایه سیستم عامل و دیتابیس، دسترسی نخواهید داشت. احتمالا

همه نیازهای کسب و کارتان باید از طریق واسط **نرم‌افزارهای تحت وب** برطرف شوند که در صورت نیاز باید توسط APIهایی به نرم‌افرار متصل شوند. معمولا این مدل سرویس برای استقرار نرم‌افزارهایی که به صورت مستقل تهیه کرده‌اید، مناسب نیستند.

مزایا

همه راهکار به صورت یکپارچه توسط ارائه‌دهنده نرم‌افرار تهیه و نگهداری می‌شود و تقریبا هر آنچه که لازم دارید در دسترس است.

معایب

کنترل کمتری روی زیرساخت نرم‌افزاری که روی آن کار می‌کند، خواهید داشت و در موقعیت‌های خاص، امکان ترکیب آن با فرآیندهای خارجی کمتر است.

از کدام مدل استفاده کنم؟

اگر توسعه دهنده نرم‌افزار هستید احتمالا تمایل بیشتری به استفاده از مدل PaaS دارید چون اجازه می‌دهد که تمرکز بیشتری روی فرآیند اصلی کدنویسی داشته باشید و درگیر مسائل زیر ساختی و شبکه‌ای نشوید.

اگر یک کاربر نهایی هستید و نرم‌افزارهای موجود در بازار جوابگوی نیاز کسب و کارتان هستند احتمالا سراغ مدل SaaS می‌روید؛ اما اگر نرم‌افزارهای موجود در بازار جوابگوی نیازتان نیستند و به هر دلیلی امکان تامین زیرساخت و نگهداری آن را ندارید و یا از نظر مالی به صرفه نیست که هزینه زیادی کنید، احتمالا از مدل IaaS استفاده

خواهید کرد.

مقیاس‌پذیر کردن نرم‌افزار

همانطور که قبلا هم اشاره شد معمولا PaaS به صورت پیش‌فرض ذاتا و خود به خود امکانات لازم برای مقیاس‌پذیر کردن نرم‌افزارها را ارائه می‌دهد؛ اما از دید یک توسعه‌دهنده نرم‌افزار باید انواع مقیاس‌ها را بشناسید و بدانید چه زمانی مقیاس افقی، مورد نظر شماست و چه زمانی، مقیاس عمودی بهترین انتخاب برای پروژه‌اتان خواهد بود.

توسعه مقیاس به صورت عمودی

ازاین روش، همواره به عنوان سخت‌ترین تصمیم برای افزایش ظرفیت نرم‌افزار یاد می‌شده است. براحتی می‌شود گفت که این نوع توسعه معمولا برای افزایش قدرت نرم‌افزار در مواجه با بار کاری بیشتر است. در این روش، فقط با بزرگ کردن سروری که نرم‌افزار روی آن نصب شده است، قدرت نرم‌افزار را برای مواجه این درخواست‌ها و بار کاری کاربران افزایش می‌دهند. در این روش، یک سرور عظیم با مجموعه‌ای زیادی CPU و صدها گیگ RAM به تنهایی برای یک نرم‌افزار در نظر گرفته می‌شود.

توسعه مقیاس به صورت افقی

در این نوع، بار ترافیکی نرم‌افزار و درخواست‌های کاربران بین چندین سرور کوچک‌تر

توزیع می‌شود که معمولا پشت یک Load balancer قرار دارند. به عبارت دیگر درخواست‌های کاربران ابتدا به دستگاه Load Balancer وارد می‌شود و دستگاه، این درخواست را به یکی از سرورهای نرم‌افزار منتقل و نتیجه را مستقلا به کاربر اعلام می‌کند.

به صورت کلی می‌توان توسعه مقیاس افقی را به دو دسته خودکار و دستی تقسیم‌بندی کرد، البته دسته‌بندی‌های دیگری نیز وجود دارد که در حوصله‌ی این نوشته نمی‌گنجد.

توسعه مقیاس به صورت دستی

در این روش بر اساس بار ترافیکی جاری یا پیش‌بینی شرایط آینده، مجموعه‌ای سرور را به سرورهای فعلی خود در کلاستر اضافه می‌کنید و آماده‌ی حضور کاربران جدید می‌شوید. به عنوان مثال، ادمین نرم‌افزار فروشگاهی هستید و تیم فروش تصمیم دارد که یک کمپین جامع را برای اهداف بازاریابی آغاز کند، طبعیتا می‌شود پیش‌بینی کرد که در زمان آغاز کمپین، بار ترافیکی کاربران از شرایط عادی بیشتر خواهد شد. در نتیجه باید قبل از شروع کمپین، مجموعه‌ای سرور به کلاستر اضافه کنید که با مشکلی یا کندی در نرم‌افزار روبرو نشوید، اکثر ارائه‌دهندگان PaaS امکاناتی را برایتان فراهم می‌کنند که بعد از اتمام کمپین، سرور مازاد را حذف کنید.

توسعه مقیاس به صورت خودکار

ادمین سیستم در این روش، مجموعه‌ای از شروط را تعریف می‌کند و سیستم به صورت خودکار، سرور مورد نیاز برای تحمل بار ترافیکی ایجاد شده را کم و یا زیاد می‌کند. به

عنوان مثال این شروط می‌تواند بر اساس تعداد همزمان کانکشن‌های HTTP به دستگاه Load Balancer یا مجموع مقدار CPU مصرف شده در وب‌سرورها باشد. این امر به راهکار اجازه می‌دهد که به صورت خودکار و بدون دخالت انسان و بر اساس شروط منطقی از پیش تعیین شده، مجموعه‌ای از سرورها در کلاستر را اضافه و یا حذف کند. این روش معمولا برای شرایطی استفاده می‌شود که بار کاری قابل پیش‌بینی نیست و در زمان‌های خاصی ممکن است این شرایط ایجاد شود؛ مثلا نوعی حمله روی سرورها وجود دارد که عملا کارکرد نرم‌افزار را کُند می‌کند، در بازه‌ای زمانی قبل از کشف و مقابله با حمله، وجود چند سرور جدید که خود به خود وارد مدار شده‌اند، کمک کننده است.

کدام روش رو انتخاب کنم؟

به عنوان یک توسعه دهنده باید زیرساختی را انتخاب کنید که هر دو روش توسعه مقیاس افقی نرم‌افزار به صورت خودکار و دستی را داشته باشد.

شرایط نگهداری نشست کاربران

اکثر ارائه‌دهندگان سرویس‌های PaaS از شما به عنوان توسعه‌دهنده انتظار دارند که نرم‌افزار با الگوهای Green Field توسعه داده شده باشند، و نسخه جدید نرم‌افزار باید برای زیر ساخت‌های ابری کاملا مستقل از نسخه‌های قبلی باشد و پیش‌نیازهای لازم برای مقیاس‌پذیر کردن را داشته باشد.

به عنوان مثال باید درنظر داشته باشید که نرم‌افزار نباید **مدیریت نشست‌ها** را به

threadهای هر سرور وابسته کند یا اینکه فایل‌هایی را روی دیسک‌های هر سرور ذخیره کند، چون عملا تعداد و نوع سرورها کاملا می‌تواند متغییر باشد و برای سازگاری با شرایط نرم‌افزار، مستقل از هر سرور به فعالیت خود ادامه دهد.

- برای حل این محدودیت‌ها عملا معماری نرم‌افزار باید یا بدون نشست باشد (که برای نرم‌افزارها پیچیده عملا امری نشدنی است) یا اینکه نشست‌ها خارج از سرور اصلی در کلاستری مجزا ذخیره شوند؛ به طوریکه فرآیند توزیع بار برای این نقش - session state- به صورت مجزا تنظیم و پیاده‌سازی شود.

- درباره فایل‌ها نیز همین روند باید مجزا طی شود. هیچ وقت فایل‌های مربوط به هر کاربر روی یک سرور ارائه‌دهنده‌ی ترافیک ذخیره نکنید؛ بلکه سعی کنید نقش filemanager را به سرویس ابری دیگری بسپارید.
می‌توانید از apiهای REST مخصوص ذخیره‌سازی فایل‌ها روی مجموعه‌ای مجزا از سرورها که روی کلاستر اصلی نیستند، استفاده کنید.

- قطعا اطلاعاتی که نیاز به نگهداری دارند روی کلاستری مجزا با سرویس‌های دیتابیس ذخیره شوند. با این شرایط به ادمین یا loadbalancer اجازه می‌دهید که با دست باز، تعداد سرور اصلی و کنترل بار ترافیکی را مدیریت کند. در مقاله‌های بعدی سعی می‌شود مباحثی درباره مدیریت Multitenancy و دیگر تکنولوژی‌های لازم برای ابزارهای ابری، ارائه شود.