remis®

۳۷
نشریه داخلی شرکت رمیس
۱۴۰۲

برنامه رمیس برای توسعه
پلتفرم‌های مدرن و DevOps

حرکت جمعی به‌سوی تحول

Container Orchestrators
یا
چه کسی بهتر می‌نوازد؟

موج کانتینر
کانتینر چیست و
چگونه کار می‌کند

کامووالت
راهکار جامع محافظت
از داده‌ها
در محیط‌های کوبرنتیس

به نام خدا

remis®

برفراز دنیای داده‌ها

فهرست مطالب

تحولی از جنس دواپس

در دنیایی زندگی می‌کنیم که همواره در حال دگرگونی و تحول است. پذیرش تغییر از بزرگ‌ترین عوامل دوام و بقا است. با مختصر نگاهی به تاریخ می‌توان دریافت که تنها آنان که خود را با امواج تحولات هماهنگ ساخته‌اند، توانسته‌اند دوام و بقای خود را حفظ کنند. اجتناب از تغییر اجتناب‌ناپذیر است.

دواپس یکی از این تغییرات و تحولات اجتناب‌ناپذیر در دنیای فناوری اطلاعات است. گاهی دواپس به درستی درک نشده و آن را معادل تکنولوژی‌ها و ابزارهای سازنده آن می‌دانند. دواپس فراتر از تکنولوژی و ابزار است. دواپس تحولی در فرآیند تولید محصول و حتی بالاتر از آن تحولی در روش زیست سازمان است. دواپس آمده تا مرزها را از میان بردارد. دیگر مرزی بین تولید، تست، ارزیابی، نگهداری و پایش نیست. همگی در طول چرخه حیات محصول در خدمت ارائه محصول با کیفیت بالاتر و زمان عرضه کوتاه‌تر هستند.

اما اینها در کجای منظومه کاری رمیس قرار می‌گیرند؟ پاسخ این سوال به صورت کامل در صفحات پیش روی شما آمده است. به صورت خلاصه باید گفت رمیس همیشه به دنبال گشودن درهای جدید و کشف دنیاهای تازه و ایجاد تجربیات بدیع برای مشتریان خود است و در این راه نه تنها به دواپس توجه می‌کند بلکه مفهوم DevOps Adoption را هم در دستور کار خود قرار داده است. امید که از این پنجره تازه گشوده شده، سهم خوبی نصیب فناوری اطلاعات ایران شود.

احسان پورمند
مدیرعامل

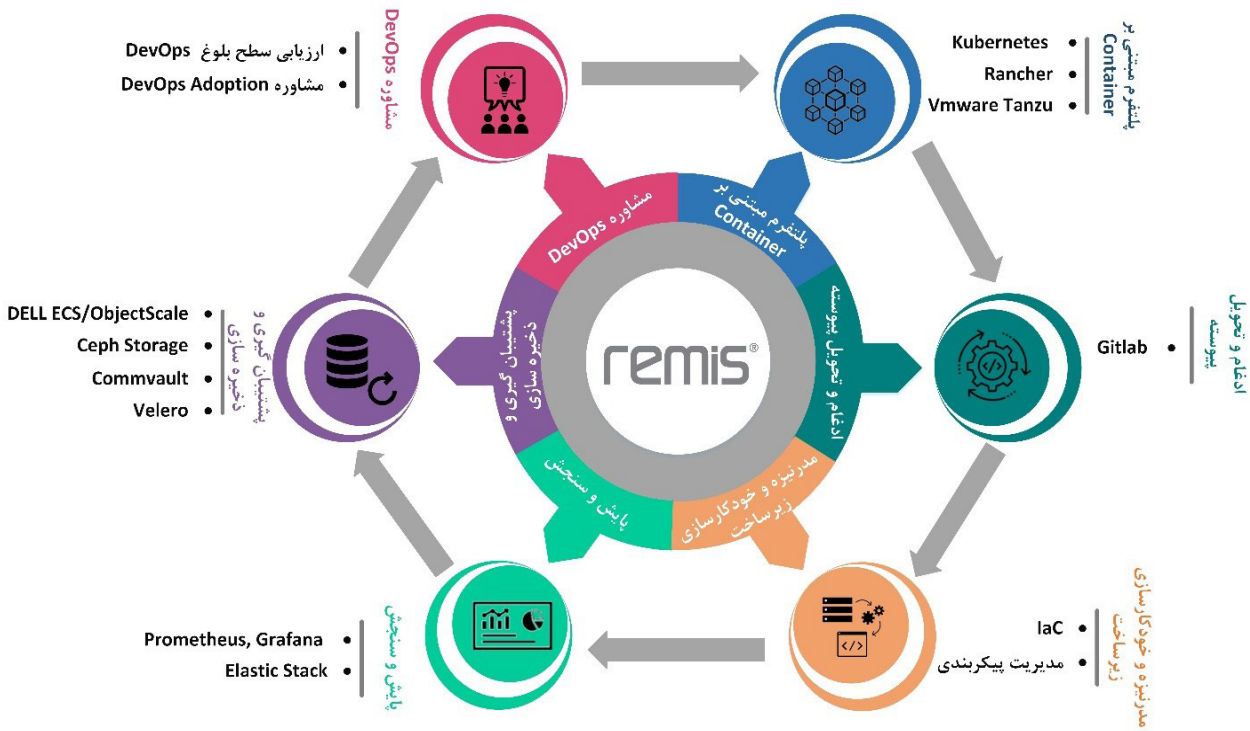
نشریه داخلی شرکت رمیس، شماره ۳۷، ۱۴۰۲

صاحب امتیاز: شرکت افزارپرداز رمیس
مدیرمسئول: مازیار نوربخش
تهیه و تولید: موسسه پرسش (پویندگان راز ستاره شمال)
همکاران این شماره: زهره رفعتی، آزاده دودانگه، شادی خرم‌شاهی
نشانی: تهران، خیابان ولیعصر(عج)، خیابان مطهری، خیابان سربداران، پلاک۲۸
تلفن: ۴۲۰۸۴۰۰۰ **دورنگار:** ۴۲۰۸۴۲۰۸

از خوانندگان و علاقمندان دعوت می‌شود در صورت تمایل مطالب خود را برای چاپ در نشریه به ایمیل info@remisco.com ارسال کنند.

راه‌کارهای DevOps و پلتفرم‌های مدرن

فیلسوف یونانی، هراکلیتوس معتقد بود تنها مولفه ثابت در جهان «تغییر» است.
این حقیقت در دنیای IT شاید بیش از جهان واقعی صدق پیدا کند. نمی‌توان در صنعت فناوری اطلاعات متوقف شد، حتی درنگی کوتاه نیز جایز نیست. دنیای IT پیوسته شاهد ظهور ابزارهای جدید و نوآوری در تکنولوژی است که برخی از آنها به پدیده‌ای در IT تبدیل می‌شوند. DevOps یکی از این پدیده‌هاست.
DevOps به صاحبان کسب‌وکار وعده محصولی با کیفیت بالاتر، قابلیت اطمینان بیشتر و امن‌تر می‌دهد. همچنین مدعی کاهش زمان عرضه به بازار است و علاوه بر تمامی اینها، عنوان می‌کند که سازمان‌ها، کارمندان و مشتریانی خوشحال‌تر خواهند داشت.
آنچه که DevOps را متمایز می‌سازد این است که کل چرخه حیات محصول را تحت مدیریت خود دارد. هدف صرفاً تولید سریع‌تر محصول نیست، زیرساخت و بستر ارائه محصول نیز از اهمیت ویژه‌ای برخوردار است. بر روی بستری که از معماری، کارایی و بلوغ مناسب برخوردار نباشد و راه‌اندازی و پیکربندی آن مبتنی بر شیوه‌های سنتی انجام شود، نمی‌توان به چابکی و کارایی موردنظر در نرم‌افزار و سرویس دست پیدا کرد. همچنین پایش و اطمینان از صحت عملکرد محصول، دریافت به موقع بازخوردها و کشف زودهنگام خطاها همگی از ستون‌های اصلی DevOps محسوب می‌شوند.
DevOps نسخه از پیش تعیین شده‌ای ندارد. هر سازمانی نیازمند نگرش، برنامه‌ریزی، هدف‌گذاری، طراحی، انتخاب زنجیره ابزار و پیاده‌سازی خاص خود است.
رمیس با بیش از بیست سال حضور چشمگیر و مستمر در حوزه فناوری دیجیتال و پیشرو در تکنولوژی‌های نوین، این بار نیز با افتخار در سفر DevOps همراه شما خواهد بود.



برنامه رمیس برای توسعه پلتفرم‌های مدرن و DevOps

حرکت جمعی به سوی تحول

رمیس به عنوان یک شرکت قدیمی در صنعت ارتباطات و فناوری اطلاعات گرچه همواره پایی در سخت‌افزار داشته؛ اما هیچ‌گاه خود را به این حوزه محدود نکرده و به عرصه‌های دیگر نیز توجه داشته که می‌توان فصل مشترک آنها را نوآوری دانست. حالا رمیس به تحول فکر می‌کند. تحولی که به نام DevOps شناخته می‌شـــود. دواپس چیست و رمیس در این زمینه قرار است چه فعالیتی داشته باشد و افق آن چیست؟ در گفت‌وگو با دو مدیر شرکت رمیس به بررسی برنامه‌ها و اقدامات این شرکت در زمینه و پلتفرم‌های مدرن و DevOps پرداخته‌ایم که در ادامه می‌خوانید.

رمیس را بیشـــتر به عنوان یک شـــرکت زیرساختی می‌شناســـیم؛ اما حالا حرف پلتفرم و DevOps بـــه میان آمده. رمیس قرار اســـت در این زمینه چه کاری انجام دهد؟ آیا این اساسا یک موضوع نرم‌افزاری نیست؟

محمد سلطانی، مدیر فنی رمیس: از چند منظر می‌توان به این موضوع نگاه کرد.

اول اینکـــه با تحولاتی که از یک‌ســـو در حوزه معماری نرم‌افزار و متدولوژی‌های توســـعه آن و از ســـوی دیگر در حوزه پلتفرم‌هـــا رخ داده، زیرساخت و پلتفرم بسیار به هم نزدیک شده‌اند و عملاًیـــک لایه برای اســـتقرار نرم‌افزار برروی آن را تشکیل داده‌اند. در این لایه، ماشین‌های مجـــازی و کانتینرها و ابزارهـــای مختلفی که تحت عنـــوان زنجیره ابزار DevOps شـــناخته می‌شـــوند، در کنار هـــم کار می‌کنند. رمیس کـــه در بســـیاری از پروژه‌ها، زیرســـاخت‌های مورد نیاز شـــرکت‌ها و ســـازمان‌ها را به منظور اســـتقرار نرم‌افزارهای کاربردی فراهم می‌کرد، در رویکردهای جدید می‌بایســـت زیرساخت‌ها، پلتفرم‌هـــا و زنجیره ابزارهـــای مرتبط را برای استقرار نرم‌افزارهای مدرن فراهم کند.

دوم اینکه DevOps یـــک بخش Ops دارد که عمدتاً در حوزه زیرســـاخت تعریف می‌شـــود. بخـــش Dev کـــه از همـــان Development یا توســـعه می‌آید بـــه فرآیند تولید و تســـت نرم‌افزار اشـــاره دارد که بعد از مدتی به مرحله اســـتقرار و پیاده‌سازی می‌رســـد. اتفاقی که بر روی یک زیرســـاخت رخ می‌دهد. نرم‌افزار پس از پیاده‌ســـازی و اســـتقرار به مرحله عملیاتی می‌رســـد و از آن پـــس باید دائمـــا مراقبت و نگهداری شـــود که همه این اتفاقات در بخش Ops رخ می‌دهـــد. طبیعتـــاً در اینجـــا موضوع زیرســـاخت و عملیاتی‌نگه‌داشـــتن آن کاملاً با حوزه کاری رمیس مرتبط اســـت. اینجا رمیس یک واحد پشـــتیبانی دارد که در آن نگهداری زیرساخت‌ها انجام می‌شود و طبیعتاً با متولیان نرم‌افزارهای ســـازمان‌های مشـــتری در ارتباط

نزدیک است. مثلاً یکی از پروژه‌هایی که رمیس عملیات نگهداری از زیرساخت آن را انجام می‌دهد «کارت ملی هوشمند» است. پروژه‌ای با چندین نرم‌افزار مختلف و مبتنی بر فریم‌ورک‌ها، پلتفرم‌ها و سیستم‌عامل‌های مختلف که در آن عملاً عملیات Ops این نرم‌افزار پس از استقرار به وسیله رمیس انجام می‌شود.

هدیه رزازان، سرپرست تیم دواپس و پلتفرم‌های مدرن:

سوالی که مطرح است این است که چرا دواپس مطرح شد. توسعه نرم‌افزار و عملیات و پشتیبانی مفاهیم جدیدی نیستند. تیم‌های توسعه و پشتیبانی همیشه وجود داشته و همیشه با هم همکاری کرده‌اند. ولی کاستی‌ها و مشکلاتی پیش می‌آمد که نشان می‌داد به چیزی فراتر از همکاری نیاز است. لازم است تیم‌های مختلف یکی شوند و یک تیم مسئولیت محصول در طول کل چرخه حیات آن را از کدنویسی و توسعه گرفته تا استقرار، پشتیبانی و عملیات را برعهده داشته باشد. این اتفاقی است که قرار است با دواپس بیفتد.

دواپس تلفیقی از نرم‌افزار، پلتفرم و پشتیبانی

> رزازان: رمیس در چند جایگاه می‌تواند ایفای نقش کند. یک نقش همان مشاوره برای DevOps Adoption یا تحول دواپس است. چون خلاء متدولوژی در پیاده‌سازی‌های دواپس کاملاً حس می‌شود. یک جایگاه دیگر هم در پیاده‌سازی DevOps Practicesهایی است که پلتفرم و عملیات در آنها پررنگ است

است؛ بنابراین نه یک شرکت نرم‌افزاری می‌تواند به تنهایی عهده‌دار آن شود و نه یک شرکت پلتفرمی مانند رمیس. لازم است یک اتحاد بین این دو برقرار شود.

تفاوت DevOps با نرم‌افزاری که ۱۰ سال یا بیشتر است نوشته شده و کار می‌کند در چیست که رمیس روی آن حساس شده؟

رزازان: الان دنیا، دنیای سرعت شده است. همه دنبال چابکی هستند که یکی از اهداف DevOps همین چابکی است. این چابکی با روش‌های مختلف حاصل می‌شود. یکی از این روش‌ها، معماری محصول است.

امروزه اپلیکیشن‌ها به سمت معماری میکروسرویس پیش‌رفته‌اند چون این چابکی را در ذات خودش دارد. معماری مونولیتیک این ویژگی را نداشته و کند است. توسعه، به‌روزرسانی و ارائه نسخه و... کندی در ذات خود دارد.

مسیر دیگر تحقق چابکی این است که

پلتفرم‌ها دارای این ویژگی باشند که در اینجا بحث IaC مطرح می‌شود. در IaC یا همان Infrastructure as Code Automation مطرح است. فرآیندهای Manual به اصطلاح Error Prone هستند. یعنی احتمال خطا و اشتباه بالاست. با اتوماتیک کردن فرآیندها علاوه بر سرعت‌بخشی، احتمال خطای انسانی نیز کاهش می‌یابد.

سلطانی: برای دستیابی به این چابکی یکی از محورهای مهم دیگر که در DevOps وجود دارد، Infrastructure as Code (IaC) است. در اینجا لایه‌های زیرساختی به صورت کد دیده می‌شوند که با یکسری کد ساده، فرآیند استقرار نرم‌افزار در زیرساخت سرعت می‌گیرد و به راحتی انجام می‌شود. برای چنین عملیاتی باید نسبت به زیرساخت موجود درک بالایی وجود داشته باشد تا بتوانیم بین همه لایه‌ها ارتباط منسجم و پیوسته برقرار کنیم. طبیعی است که یک شرکت زیرساختی در اینجا بسیار کمک‌کننده و تسهیل‌گر است.

رمیس به صورت عملیاتی در این حوزه چه کرده است؟

سلطانی: ورود ما به این حوزه، چند سال پیش با پروژه‌ای بود که در آن کارفرما (یکی از بانک‌های کشور) از ما خواسته بود پلتفرم‌های مورد نیاز برای اجرای یکی از نرم‌افزارهایی که به تازگی و به صورت Containerized توسعه داده بودند را ایجاد کنیم. ما در آن پروژه «ارکستریتور رنچر» را انتخاب کردیم و کلاسترهای کوبرنتس محیط‌های مختلف UAT و Production ایجاد کردیم. به مرور نیازهای مختلفی مانند لاگینگ، ذخیره‌سازی Read Write Many، مانیتورینگ، بکاپ، رجیستری امن و غیره مطرح شد که ما را به این نتیجه رساند برای موفقیت در این حوزه و پروژه‌ها مشابه باید به چرخه IaC، CI/CD، زنجیره ابزارها، معماری نرم‌افزارهای مدرن و غیره با رویکرد جامع و ۳۶۰ درجه، تسلط پیدا کنیم. از این جهت در خلال آن پروژه و چند پروژه بعدی سعی کردیم Competency فنی شرکت را در این زمینه‌ها تقویت کنیم و خوشبختانه در این مسیر تجربیات ارزشمندی کسب کردیم.

سلطانی: DevOps یک بخش Ops دارد که عمدتاً در حوزه زیرساخت تعریف می‌شود. بخش Dev که از همان Development یا توسعه می‌آید به فرآیند تولید و تست نرم‌افزار اشاره دارد که بعد از مدتی به مرحله استقرار و پیاده‌سازی می‌رسد. اتفاقی که بر روی یک زیرساخت رخ می‌دهد. نرم‌افزار پس از پیاده‌سازی و استقرار به مرحله عملیاتی می‌رسد و از آن پس باید دائما مراقبت و نگهداری شود که همه این اتفاقات در بخش Ops رخ می‌دهد

علاوه بر این، متوجه شدیم که برای موفقیت این پروژه‌ها، در سمت کارفرما هم نیاز است تحولاتی رخ دهد. با مطالعه پروژه‌های مشابه در کشورهای دیگر هم متوجه شدیم این نیاز در آنها هم حس شده و باعث شده مفهوم تحول DevOps (Devops Adoption) مطرح شود. متدولوژی‌های شرکت‌های بزرگ مشاوره و System Integrator دنیا را مطالعه کرده و در نهایت متدولوژی رمیس در حوزه تحول DevOps را طراحی کردیم.

DevOps Adoption که به عنوان یک پیش‌نیاز تحولی در سازمان‌ها و شرکت‌های مشتری برای عملیاتی‌کردن DevOps مطرح می‌شود و در واقع یک حرکت جمعی است.

برای آینده چه برنامه‌ای دارید؟

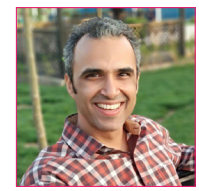
سلطانی: خیلی علاقمندیم روی موضوع DevOps Adoption بیشتر کار کنیم چون معتقدیم این موضوع نیاز جدی در آینده نزدیک خواهد بود. وقتی تعریف DevOps را از منظر همه شرکت‌ها، موسسات و سازمان‌های بزرگ نگاه می‌کنیم می‌بینیم کلید واژه Culture به معنی «فرهنگ» بین آنها مشترک است. طبیعتاً فرهنگ‌سازی یک پروژه زمان‌بر و بنیادین است و محورهای مختلفی اعم از نزدیکی تیم‌ها به هم، ایجاد نگاه محصول‌محوری بین آنها و ایجاد تغییر ساختار نیروی انسانی و تزریق دانش‌ها و فیلدهای جدید و آشنایی با زنجیره ابزارهای فراوان DevOps و... را شامل می‌شود که باید در تحول DevOps به آنها پرداخت.

موضوعات دیگری از جنس تکنولوژی است و شامل همین پلتفرم‌ها و ابزارها می‌شود. ایده دیگری درباره ایجاد چندین محصول به صورت باندل داریم که شامل همه المان‌هایی که درباره‌شان صحبت کردیم، می‌شود. به طوریکه بهره‌بردار این محصول خیلی درگیر تک‌تک ابزارها و استقرار آنها نباشد.

رزازان: رمیس در چند جایگاه می‌تواند ایفای نقش کند. یک نقش همان مشاوره برای DevOps Adoption یا تحول دواپس است. چون خلاء متدولوژی در پیاده‌سازی‌های دواپس کاملاً حس می‌شود. یک جایگاه دیگر هم در پیاده‌سازی DevOps Practicesهایی است که پلتفرم و عملیات در آنها پررنگ است. از قبیل IaC،Configuration Managment و مانیتورینگ که اتفاقاً نقش کلیدی در پیاده‌سازی دواپس دارد. IaC موضوعی است که سازمان‌ها مستقل از اینکه به سمت دواپس بروند یا نه به آن نیاز دارند چون دوره فرآیندهای دستی گذشته و رمیس نیز دانش خوبی در این زمینه دارد.

سلطانی: در آینده به دنبال این هستیم که پروژه‌های رمیس را از طریق Infrastructure as Code اجرا کنیم؛ مثلاً یک پروژه داشتیم که ۱۲۰ سرور را به مشتری تحویل دادیم. باید تست‌هایی روی اینها انجام می‌دادیم، به‌جای اینکه ۱۲۰ سیستم عامل را روی اینها نصب کنیم از طریق روش Infrastructure as Code آن را انجام دادیم. در یک پروژه دیگر باید دو هزار ماشین را از یک بستر به بستر دیگری مهاجرت می‌دادیم. در اینجا هم از همین روش Infrastructure as Code استفاده کردیم. در آینده در پروژه‌های غیر پلتفرمی و غیر DevOps هم از همین روش Infrastructure as Code بهره خواهیم برد.

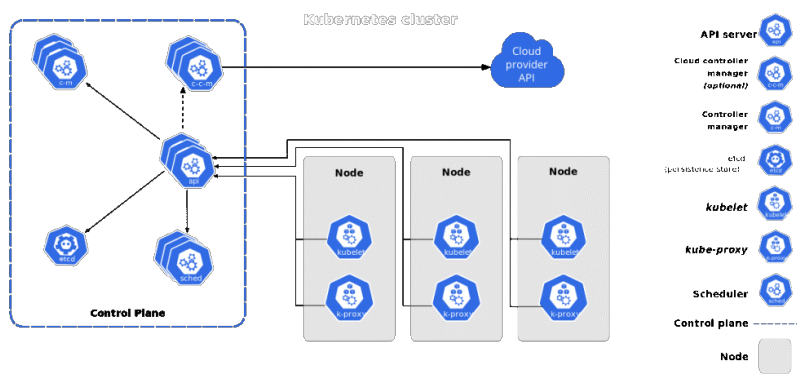
Container Orchestrators یا چه کسی بهتر می‌نوازد؟



خدایار دوستدار

هرچه تعداد عناصر دخیل در راه‌حل‌های فناوری اطلاعات بیشتر شوند به نحوی پیچیدگی راهبری این راه‌حل‌ها و همچنین پیچیدگی مدیریت چرخه عمر بیشتر می‌شود. هرچه پیچیدگی کاری بیشتر باشد سپردن آن به عملکرد انسانی پرخطرتر و پرخطاتر خواهد بود و چنین کارهایی معمولا بهترین گزینه‌ها برای شروع عملیات خودکارسازی هستند، البته در صورتی که بیش از حد ممکن برای خودکارسازی پیچیده نباشند! اگر این مقاله را می‌خوانید احتمالا در حوزه فناوری اطلاعات و مخصوصا حوزه دواپس یا حوزه‌های مشابه آن کار می‌کنید. اگر (و تنها اگر!) دستی هم در موسیقی داشته باشید به احتمال زیاد تا به حال شباهت بین بعضی راه‌حل‌ها و یک ارکستر توجه شما را نیز جلب کرده است: اینکه چطور واحد (کانتینر)هایی که در اولین نظر کاملا مستقل و جدا از هم به نظر می‌رسند باید در هماهنگی کامل با بقیه اجزا کار کنند، اینکه چقدر زمان‌بندی و دقیق‌بودن اجرا- حتی در حد صدم‌ثانیه!- برای گوش‌نواز بودن و «درست» اجرا شدن یک قطعه مهم است و اینکه چطور اگر یکی از اجزاء ارکستر (ژوست) درست و هم‌آهنگ با ارکستر ننوازد کار «خراب» می‌شود. اینها همه مواردی است که بین یک ارکستر و یک «راه‌حل پیچیده خودکار» مشترک است. بنابراین برای خوب کار کردن این ارکستر (مثل بقیه ارکسترها) یک رهبر ارکستر یا Orchestrator مورد نیاز است. کسی که مسئولیت هماهنگی بقیه اجزا با یکدیگر را به عهده بگیرد و آنها را با یکدیگر هماهنگ کند. اجزایی که درست مثل نت‌های موسیقی طول عمر کمی دارند (البته از نظر فیزیکی، اگرنه «تنها صداست که می‌ماند»!) و ممکن است چند ثانیه بعد دیگر کلا وجود نداشته باشند و به همین دلیل نمی‌توان از آنها انتظار مسئولیت‌پذیری داشت یا حتی وظیفه‌ای فرای عمر پیش‌بینی شده آنها به آنها ارجاع داد. اینجاست که پای محصولاتی به نام ارکستراتورهای کانتینری یا Container Orchstrator به بازار باز می‌شود. ابزارهای کانتینر یا همان Container Runtime های جدید و به‌روز مثل داکر، پادمن، Buildah و غیره امکانات گسترده‌ای را برای Containerize کردن و ارائه[1] کد نوشته شده ارائه می‌دهند، اما برای استقرار اپلیکیشن‌های پیچیده و همچنین خودکارسازی زیرساخت به یک ابزار مناسب Container Orchestrator نیاز خواهد بود.

تصویر ۱



مروری بر ابزارهای مدیریت کلاسترهای کانتینری یا ارکستراتورهای کانتینری

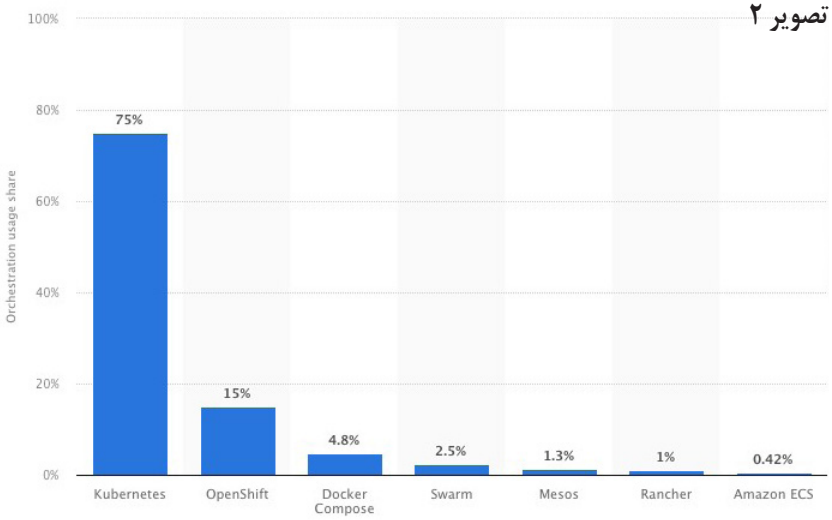
ابزارهای مدیریت کلاسترهای کانتینری شامل ابزارهای On-Premise و ابری هستند. با توجه به کاربری بیشتر ابزارهای On-Premise در ایران، در این مقاله به بررسی این دسته پرداخته می‌شود.

۱. Kubernetes

کوبرنتیس یک ابزار متن‌باز و Out-of-the-Box برای Container Orchestration است که از

یـک Scheduler و مدیریت منابـع عالی برای استقرار کاراتر اپلیکیشن‌های با دسترس‌پذیری بالا بهـره می‌بـرد. کوبرنتیس در بسـیاری از سازمان‌ها ارکستراتور کانتینری پیش‌فرض است. بر اسـاس CNCF Landscape بیـش از ۱۰۹ ابزار برای مدیریـت کانتینرها وجود دارد که ۸۹ درصد آنها از انواع کوبرنتیس استفاده می‌کنند. مدیریـت پـروژه کوبرنتیس برعهـده CNCF (Cloud Native Computing Foundation) است و مشـارکت‌کننده‌هایی از سراسر جهان در این پروژه همکاری می‌کنند، مشارکت‌کننده‌هایی از شرکت‌های بسـیار بزرگ تا توسعه‌دهنده‌های متن‌باز شـخصی. معماری کلان کوبرنتیس در تصویر ۱ نشـان داده شـده اسـت. کوبرنتیس امکانات زیـادی را فراهم می‌کند کـه ابزارهای رسـمی داکر ارائه نمی‌دهند، علاوه بر آن شروع به کار با کوبرنتیس راحت اسـت. در این زمینه سرویس‌های مدیریت‌شده نقش پررنگی را بازی می‌کنند. بر اسـاس گزارشـی از Datadog[۲] در مورد به‌کارگیری کوبرنتیس در سازمان‌ها، حدود ۹۰ درصد کاربران کوبرنتیس از سـرویس‌های مدیریت‌شده اسـتفاده می‌کنند. کوبرنتیس در نسخه ۱,۲۰ استفاده از داکر به عنوان Container Runtime را متوقف کرد، اما به این معنی نیست که Docker Imageها دیگـر در کوبرنتیس کار

تصویر ۲



نخواهند کرد، بلکه تنها به این معنی اسـت که Container Runtime پیش‌فـرض کوبرنتیس از داکر به CRI تغییر پیدا کرده است. همانطور که انتظار داریم کوبرنتیس بزرگ‌ترین سهم بازار را در میان محصولات همرده خود دارد. در تصویر ۲ سهم بازار کوبرنتیس بر اساس آمار سال ۲۰۲۰ استاتیستیکا مشاهده می‌شود.

۲. Openshift

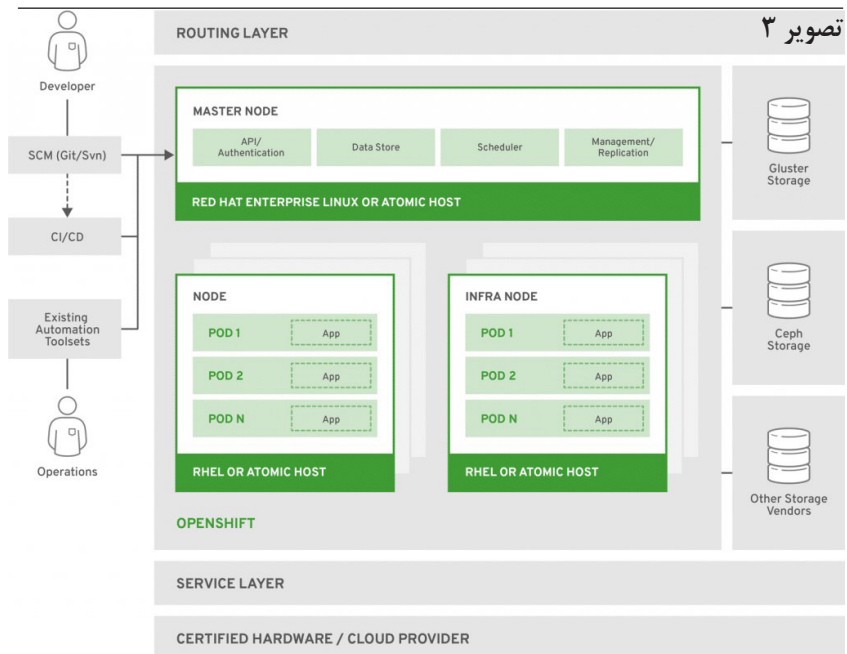
اوپن‌شـیفت بر اسـاس کوبرنتیس ساخته شده اسـت. هم یک نسـخه متن‌باز و رایـگان دارد (OpenShift Origin یا OKD) و هم یک نسخه سازمانی (Redhat OpenShift) که هردو توسط شـرکت Redhat مدیریت می‌شـوند. علاوه بر امکانات خود کوبرنتیس، اوپن‌شـیفت امکانات و محصـولات جانبی دیگری را هـم به صورت Out-of-the-Box ارائـه می‌دهـد. یعنی برای استفاده از امکاناتی مثل Ingress یا Persistent Storage یا پایگاه‌داده‌های مختلف نیازی به نصب و پیکربندی دستی این امکانات روی اوپن‌شیفت نیسـت و این موارد بر روی اوپن‌شـیفت خیلی راحت‌تـر از کوبرنتیـس قابل اجرا و اسـتفاده هسـتند. معماری کلان اوپن‌شیفت را در تصویر ۳ مشاهده می‌کنید.

اوپن‌شـیفت ماننـد کوبرنتیس هـم به صورت Managed Service روی سـرویس‌دهنده‌های عمومی یـا خصوصی ابـری و هم بـه صورت On-Premise قابل نصب است. (امکانات بیشتر Out-of-the-Box، همانطـور کـه اسـتفاده از اوپن‌شـیفت را راحت‌تر می‌کنـد ولی در همان حـال انعطاف‌پذیری آن را نصـب به کوبرنتیس پائین مـی‌آورد.) مخصوصا تنظیمـات امنیتی اوپن‌شیفت سخت‌گیرانه‌تر است و بنابراین همه برنامه‌ها و افزونه‌هایـی که روی کوبرنتیس قابل نصب هستند روی اوپن‌شیفت قابل اجرا نیستند. البته به دلیل سهم بازار بالای اوپن‌شیفت معمولا محصـولات و افزونه‌های محبـوب و بالغ روش نصب جداگانه‌ای برای اوپن‌شیفت ارائه می‌دهند.

۳. Rancher

تمرکز رنچر روی مفهوم چند-خوشه‌ای یا چند-کلاستری و چند-ابری است. به این معنی که در

تصویر ۳



صورتی که چند کلاستر کوبرنتیس با توزیع‌های متفاوت روی چند بستر ابری متفاوت داشته باشید همه آنها را می‌توان به صورت متمرکز با یک پنل رنچر مدیریت و حتی مستقر کرد. تمام توزیع‌های مورد تائید CNCF روی رنچر قابل مدیریت هستند. رنچر توزیع جداگانه و مستقل خودش از کوبرنتیس را هم دارد. این توزیع RKE نام دارد که مخفف Rancher Kubernetes Engine است. رنچر در حال حاضر به شرکت Suse تعلق دارد و در حال گسترش و ایجاد اکوسیستم محصولات خود است و بازار خوبی دارد و نصب و اجرای کوبرنتیس را بسیار ساده کرده است. مشابه با اوپن‌شیفت، رنچر هم امکان نصب، راه‌اندازی و مدیریت و پیکربندی افزونه‌های استاندارد و غیراستاندارد کوبرنتیس را به صورت مبتنی بر وب و گرافیکی فراهم می‌کند. تصویر ۴ معماری کلی رنچر را نشان می‌دهد.

۴. Docker Swarm

اکوسیستم داکر از ابزارهای مربوط به گسترش تا تولید تشکیل شده است. در فهرست این ابزار، داکر سوارم در رده «مدیریت خوشه» قرار می‌گیرد. مجموعه‌ای از Docker-Compose، داکر سوارم، شبکه Overlay و یک ابزار کشف سرویس[3] عالی مانند etcd یا Consul می‌تواند جهت مدیریت خوشه‌ای از داکر کانتینرها به کار گرفته شود. داکر سوارم کماکان از نظر کارایی‌ها در مقایسه با بقیه ابزارهای مدیریت خوشه کانتینری، در حال بلوغ است. با در نظر گرفتن بازه بزرگ مشارکت‌کنندگان در پروژه داکر، رسیدن به این بلوغ و توانایی رقابت با بقیه محصولات هم‌رده چندان دور به نظر نمی‌رسد. داکر برنامه خوبی برای اجرایی کردن و به تولید رساندن داکر سوآرم و استفاده از آن در محیط‌های عملیاتی مستند کرده است.

۵. Hashicorp Nomad

نومَد یکی از محصولات شرکت HashiCorp است که در رده محصولات پلتفرم‌های ارکستریشن گسترش می‌یابد. نومَد در زمینه مدیریت اپلیکیشن‌ها در مقیاس‌های مختلف، فلسفه و معماری‌ای شبیه به کوبرنتیس دارد. علاوه بر آن نومَد هم می‌تواند میزبان Workload های کانتینری باشد و هم Workloadهای غیرکانتینری. همانطور که می‌توان انتظار داشت نومَد قابلیت یکپارچگی بسیار خوبی با سایر محصولات هشی‌کرپ، Vault، Consul و ترافرم دارد. به صورت کلی موارد استفاده نومَد شامل فهرست زیر است:

- ارکستریشن کانتینرها
- ارکستریشن اپلیکیشن‌های غیرکانتینری
- خودکارسازی شبکه‌سازی سرویس‌ها[4] توسط Consul

۶. Mesos

مزوس در ابتدا توسط شرکت توئیتر برای زیرساخت این شرکت خلق شد و بعد متن‌باز شد. این ارکستراتور در شرکت‌هایی مانند Airbnb، eBay و غیره استفاده می‌شود. مزوس برخلاف اکثر محصولات هم‌رده‌اش (غیر از نومَد) مخصوص کانتینرها ساخته نشده است. مزوس را می‌شود برای مدیریت و ارکستریشن ماشین‌های مجازی یا حتی فیزیکی نیز به کار گرفت. از این جهت مزوس در راه‌حل‌های ابرداده[5] و شبیه آن نیز کاربرد دارد. حتی کلاسترهای کوبرنتیس را هم می‌توان بر روی مزوس اجرا کرد.

۷. VMWare Tanzu TKGI

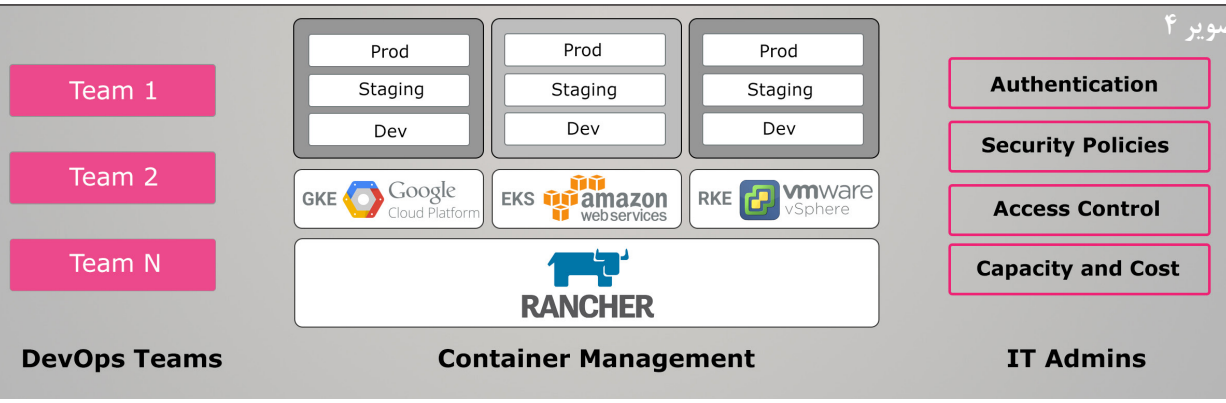
طی چند سال گذشته شرکت VMWare با خرید و مشارکت در پروژه‌هایی مانند Bitnami، Pivotal، Wavefront، Heptio و Cloud Foundry و معرفی محصول تانزو به بازار کانتینرها و ارکستراتورها قدم گذاشته است و در رده ارکستراتورها TKGI را معرفی کرده است که مخفف Tanzu Kubernetes Grid Integration است که قبلا به عنوان VMWare Enterprise Pivotal Container Service (PKS) شناخته می‌شد. TKGI با یک رجیستری اختصاصی Harbor و یک ابزار کامل مدیریت چرخه عمر کانتینر ارائه شده که هم به صورت On-Premise و هم به صورت ابری و مدیریت شده قابل استفاده است. چنانچه انتظار می‌رود این محصول قابلیت بسیار خوب- و حتی از نظر مالی- به صرفه‌ای برای یکپارچه شدن با بقیه محصولات شرکت VMWare دارد و برای شرکت‌هایی که روی محصولات VMWare سرمایه‌گذاری کرده‌اند می‌تواند گزینه جذابی باشد.

راهنمای انتخاب (مقایسه اجمالی)

در بازار ارکستراتورها، محصولات متنوع و پیچیده‌ای وجود دارند و هرکدام از این محصولات از جوانب متعددی قابل بررسی و مقایسه هستند. از همین رو انتخاب محصول مناسب معمولا چالش‌برانگیز است. برای چنین انتخابی از یک سو باید نیازمندی‌های هر مشتری به خوبی شناخته شده و از سوی دیگر امکانات و قابلیت‌های هرکدام از ارکستراتورها بررسی شده و با دیگر بازیگرهای بازار مقایسه شود.

شرکت‌های فروشنده این محصولات، برای رقابت

تصویر ۴



Product	Rancher	OpenShift	Tanzu
Cluster Operations			
Ease of install, Config & Maintenance	4	3	3
Intuitive UI	4	4	3
Multi-cloud	4	3	3
Multi-cluster	4	2	3
Edge Support	4	1	1
Hosted Kubernetes Support	4	1	2
Bare Metal, OpenStack & vSphere	4	3	2
Import Existing Clusters	4	3	3
High Availability and Healing	4	4	3
Load Balancing	4	2	2
Centralized Audit	4	3	2
Self-service Provisioning	4	2	2
Private Registry & Image Management	4	4	4
Cluster Upgrades & Version Management	4	4	2
Storage Support	4	3	4
Arm Support	4	0	0
Airgap Support	4	3	3
Etcd Backup and Restore	4	2	3
Security Policy and User Management			
Active Directory and LDAP Support	4	4	4
Pod and Network Security Policies	4	3	2
CIS Benchmark Adherence & Tracking	4	3	2
Global RBAC Policies	4	2	3
Shared Tools and Services			
Application Catalog	4	4	2
Provision with Config Management Systems	4	2	2
Integration with CI/CD Solutions	4	4	2
Advanced Monitoring	4	4	2
Alerts and Notifications	4	4	1
External Log Shipping	4	4	2
Windows Container Support	4	4	0
Integrated Service Mesh Support	4	3	1
Enterprise SLA	4	2	2
Community Traction	4	3	0
Additional from Platform9:			
Deployment Model(s) Supported	2	3	2
Breadth of Operating Systems Supported	4	1	4
Hybrid Cloud Integrations and APIs	3	3	4

بهتر و باقی‌ماندن در بـــازار، به صورت دوره‌ای و منظم محصول خود، نیازمندی‌ها و سمت‌و سوی بازار و همچنین محصول رقبـــا را ارزیابی کرده و گزارش‌های مبســـوطی ارائه می‌کنند. در این مقاله دو نمونه از این گزارش‌ها بررســـی شده و جدول روبرو براســـاس این دو گزارش که توسط شـــرکت‌های Platform9 و Suse منتشر و تهیه شده است. نکته جالب در هردوی این گزارش‌ها حضـــور بازیگر جدیدی از شـــرکت گوگل به نام Anthos اســـت که یک ابزار قوی Multi Cloud بـــوده و طبیعتاً به عنوان یک ارکســـتراتور قوی هم می‌تواند ایفای نقش نمایـــد. البته چون این محصـــول در بازار ایران فعـــلا جایگاهی ندارد از ذکر آن در جدول روبرو صرف‌نظر شـــده اســـت. ســـعی شـــده با تطبیق و ارزیابی هردو گزارش، یک جدول نهایی برای مقایســـه استخراج شود. با توجه به اینکه این گزارش توسط شرکت‌های تولیدکننده نرم‌افزار تهیه شـــده اســـت، طبیعتا منظور از هرکدام از این محصولات ذکر شـــده، محصول خریداری شده از شـــرکت و به صورت Supported است. طبیعی است در صورت نصب نســـخه‌های Community بایـــد انتظار مقداری تفاوت در نتیجه و تجربه را داشـــته باشـــیم. به خصوص در مورد محصولاتی مثل OpenShift و Tanzu که چندان در زمینه Community قابل اتکا نیستند. در مقابل در مورد Rancher به خاطر مدل کسب‌وکار متفاوت تجربه تقریبا یکسانی را در نسخه‌های Supported و Community شاهد خواهیم بود.

کلام پایانی

بـــرای انتخاب بهترین ابزار بـــرای هر محصول و پروژه می‌بایســـت نیازمندی‌ها به صورت دقیق بررســـی و با ویژگی‌ها، مزایا و معایب هر یک از ابزارها تطابق داده شـــود. ایـــن مقاله می‌تواند به عنوان راهگشـــای این بررسی مورد استفاده قرار گیرد.

پی‌نوشت:

۱- Ship

۲- https://www.datadoghq.com/container-report/#1

۳- Service Discovery Tool

۴- Service Network

۵- Big Data

موج کانتینر؛ کانتینر چیست و چگونه کار می‌کند؟

محمدعلی کمالیان

کانتینر امروزه نقش پررنگی در چرخه تولید و توسعه نرم افزار ایفا می‌کند. از مرحله ساخت گرفته تا استقرار سرویس، کانتینر به این چرخه روح و سرعت می‌بخشد. ذات سبک کانتینر کمک می‌کند تا با سرعت بالاتر و استفاده کمتر از منابع، محیط‌های متعدد با ساختار مختلف طراحی و ایجاد شود. در پلتفرم‌های سنتی، شرکت‌ها و سازمان‌ها ناگزیرند با پذیرش ریسک‌های امنیتی سرورهایی را به صورت اشتراکی برای سرویس‌های خود در نظر بگیرند یا برای هر سرویس یک سرور مجزا تامین کنند که در مقیاس بالا از نظر اقتصادی مقرون به صرفه نخواهد بود. از طرفی با معرفی معماری میکروسرویس نیاز به تکنولوژی که بتواند محیط‌هایی کوچک و چابک با ساختارهای متفاوت را با سرعت بالایی در اختیار سازمان قرار دهد بیش از پیش احساس می‌شد. در این مقاله نگاهی کوتاه به تکنولوژی کانتینر و مزایای آن خواهیم انداخت.

تاریخچه کانتینر

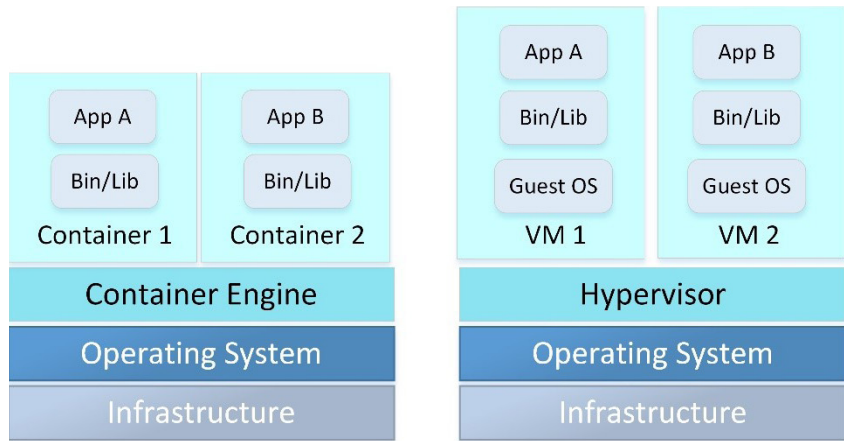
کانتینر از بستر چندین تکنولوژی مختلف بوجود آمده است که هر کدام از آنها سطحی از تفکیک‌پذیری[1] را فراهم می‌کنند. در سال ۱۹۷۹ زمانی که Unix Version7 درحال توسعه بود مفهومی به نام Chroot معرفی گردید که قادر بود Root Directory یک پروسس یا یک دستور را تغییر دهد. هدف اصلی از این کار محدود کردن سطح دسترسی سرویس به فایل‌های سیستمی بود. بسیاری Chroot را با عنوان Chroot Jail معرفی کردند به این دلیل که عملیات Chroot باعث می‌شود تا سرویس عملا در زندانی گرفتار شود که خبری از دنیای بیرون خود ندارد و تمامی نیازهای آن در همان محیط برایش فراهم شده است. امروزه از Chroot برای بالا بردن سطح امنیت بسیاری از سرویس ها استفاده می‌شود.

در سال ۲۰۰۰ شرکتی به نام R&D Associates به دنبال روشی بود تا بتواند سرویس‌های موجود خود که بر روی FreeBSD قرار داشتند را از یکدیگر مجزا کند تا طیف وسیعی از نگرانی‌های امنیتی و مدیریتی را برطرف سازد. برای رسیدن به این هدف به جای تغییر ساختار و اضافه کردن کانفیگ جدید در سطح سرویس، سیستم به بخش‌ها و پارتیشن‌های مختلف تقسیم شد و هر بخش به عنوان یک محیط مجزا در نظر گرفته شد که دربرگیرنده فایل، منابع و IP مشخص می‌بود و کاربر خاصی به آن دسترسی داشت. رویکرد FreeBSD Jails عملا انقلاب بزرگی در حوزه مجازی‌سازی ایجاد کرد و بسیاری از شرکت‌ها به دنبال استفاده از این رویکرد و اضافه کردن قابلیت‌های جدید به آن بودند تا بتوانند از آن برای پیشبرد اهداف خود استفاده کنند. به عنوان مثال در سال ۲۰۰۱ پروژه متن‌باز Linux VServer توانست با اضافه کردن قابلیت‌های جدید به مکانیزم Jail، مفهوم مجازی‌سازی در سطح سیستم‌عامل[2] را بیش از پیش تقویت کند. قدرت Jailسازی این پروژه نسبت به پروژه‌های قبل بسیار بالاتر بود به همین دلیل از این تکنولوژی برای ارائه خدمات VPS استفاده می‌شد. در سال‌های ۲۰۰۴ و ۲۰۰۵ دو پروژه دیگر به نام‌های Solaris Containers و Open Virtuozzo معرفی شدند که قابلیت‌های جدیدی به مجازی‌سازی در سطح سیستم‌عامل اضافه کردند تا اینکه گوگل در سال ۲۰۰۶ قدم اصلی را برداشت و مفهوم Process Containers را به دنیا معرفی کرد. هدف اصلی از طراحی این پروژه ارائه راهکاری یکپارچه و منعطف برای مجازی‌سازی در سطح سیستم‌عامل بود. به وسیله این قابلیت کاربر می‌تواند به راحتی به اختصاص منابعی مانند حافظه، پهنای باند شبکه و پردازنده یا ترکیبی از آنها به سرویس‌های خود بپردازد. قدرت مانوردهی این تکنولوژی در مدیریت منابع بسیار بالاست، شما می‌توانید میزان تخصیص و استفاده منابع هر سرویس را به راحتی مانیتور کرده و نیز تغییر دهید. از آنجایی که در آن زمان درک درستی از واژه «Container» در دنیای آی‌تی وجود نداشت، گوگل تصمیم گرفت نام این پروژه را به Cgroup (Control Group) تغییر دهد که امروزه به عنوان یکی از مهم‌ترین قابلیت‌های کرنل لینوکس به حساب می‌آید.

دو سال بعد یعنی در سال ۲۰۰۸ شرکت Red Hat قابلیت Namespace در کرنل لینوکس را معرفی کرد که با اعمال محدودیت در سطح پراسس‌ها، تفکیک‌پذیری در سطح بالاتری را قابل پیاده‌سازی کرد. به وسیله Namespace این امکان بوجود آمد تا کاربر تعیین کند که سرویس مورد نظر با چه سرویس‌های دیگری اجازه تعامل داشته باشد.

در این بین توسعه‌دهندگان شرکت IBM موفق شدند با استفاده از قابلیت Cgroup و Namespace محصولی جامع و یکپارچه برای مجازی‌سازی در سطح کانتینر به نام Linux Container یا همان LXC را به دنیا معرفی کنند. LXC اولین محصولی بود که به کاربر این اجازه را می‌داد که به راحتی با استفاده از قابلیت‌های کرنل لینوکس بر روی یک هاست چندین سیستم لینوکس مجزا ساخته و مدیریت کند. تمامی شاخصه‌های محیط ساخته شده نسبت به محیط دیگر تفکیک شده است و عملا کاربر محیط ساخته شده را می‌تواند به عنوان یک ماشین مجزا در نظر بگیرد. در دنیای LXC هر کدام از این محیط‌ها یک کانتینر در نظر گرفته می‌شود. این محصول به دلیل داشتن خط فرمان ساده به سرعت میان کاربران طرفداران زیادی پیدا

تصویر ۱



کرد و امروزه با وجود محصولات مختلف در این حوزه همچنان LXC طرفداران خود را دارد. بعد از LXC پروژه‌های زیادی مانند Warden در سال ۲۰۱۱ و LMCTFY در سال ۲۰۱۳ سعی کردند مفهوم کانتینر را در دنیا نهادینه کنند اما هیچ‌کدام به اندازه Docker که در سال ۲۰۱۳ معرفی شد، موفق نبودند به همین دلیل بسیاری کانتینر را با عنوان Docker می‌شناسند.

کانتینر چیست و چگونه کار می‌کند؟

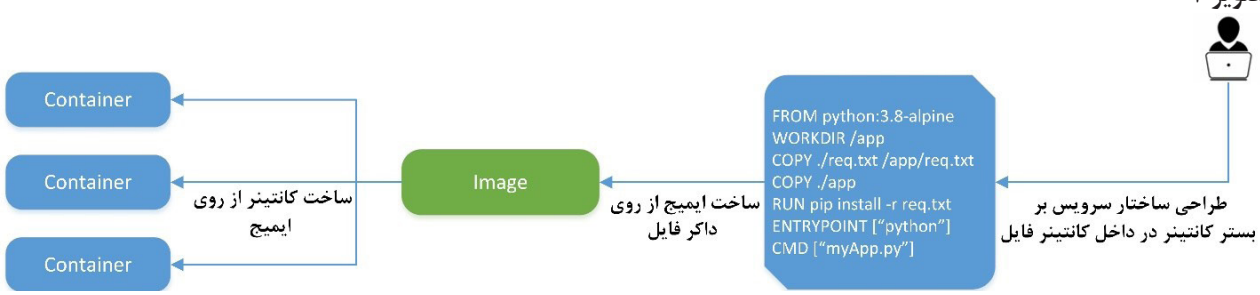
همانند تکنولوژی VMWare هدف اصلی از کانتینر ارائه محیط‌های تفکیک شده به ازای سرویس است با این تفاوت که نسبت به VMWare محیط‌های ایجاد شده سبک‌تر و از طرفی ساخت و ارائه آنها سریع‌تر است. همانطور که در بخش «تاریخچه کانتینر» ذکر شد، این تکنولوژی به عنوان مجازی‌سازی در سطح سیستم‌عامل تعریف می‌شود در حالی که VMWare مجازی‌سازی را در سطح سخت‌افزار پیاده‌سازی می‌کند.

«مجازی‌سازی در سطح سیستم‌عامل» الگویی در سیستم‌عامل است که به کمک قابلیت‌های کرنل به کاربر اجازه می‌دهد از سرویس خود چندین نمونه (Instance) تفکیک شده در سطح User-Space ایجاد و از آن استفاده کند. این نمونه‌ها در طول زمان نام‌های مختلفی پیدا کردند به طور مثال Chroot به عنوان Jail نامگذاری گردید و در دنیای تکنولوژی‌هایی مانند LXC و داکر به عنوان کانتینر نام‌گذاری شدند. اما سوال مهم این است که چرا کانتینر را ماهیتی با وزن سبک معرفی می‌کنند؟ برای جواب به این سوال باید کمی معماری مجازی‌سازی در دنیای کانتینر را بهتر درک کنیم. همانطور که در تصویر ۱ مشاهده می‌شود در محیط‌های کانتینری در سطح سیستم‌عامل لایه‌ای به اسم Container Engine وجود دارد که همان لایه مجازی‌سازی در سطح کانتینر است. به وسیله این لایه می‌توان با کمک کرنل سیستم‌عامل نمونه‌های متعددی از سرویس را در سطح کانتینر اجرا کرد. نکته مهم این است که کانتینر دارای یک سیستم‌عامل مجزا نیست و نیازمندی‌هایی مانند Binary و Library مربوط به سرویس را در دل خود به صورت یک پکیج کامل در اختیار دارد و این نه تنها باعث سبک شدن کانتینر می‌شود، بلکه وابستگی به لایه سیستم‌عامل به میزان قابل توجهی کاهش پیدا می‌کند. در تصویر ۱ تفاوت ساختاری مجازی‌سازی در سطح سیستم‌عامل و سخت‌افزار نشان داده شده است. اما کانتینر چگونه آن‌چه را که سرویس برای اجرا شدن لازم دارد را دریافت می‌کند؟ برای جواب به این سوال بهتر است مشخص کنیم کانتینر از چه ماهیتی ساخته می‌شود. در دنیای کانتینر ایمیج اصلی‌ترین نقش را بازی می‌کند، به این علت که حاوی تمامی نیازمندی‌های سرویس برای اجرا است. می‌توان ایمیج را مشابه Template در دنیای VMware در نظر گرفت که به تنهایی قابلیت سرویس‌دهی ندارد و برای اینکه تبدیل به محیطی زنده شود باید با دستوری از آن کانتینر ساخت.

به طور کلی ایمیج دربرگیرنده اپلیکیشن کد، کتاب‌خانه‌ها، ابزار و تمام وابستگی‌هایی است که ساختار و چگونگی استفاده از آنها در قالب یک کانتینر فایل تعریف می‌شود. در زمان طراحی ایمیج باید به این نکته توجه کرد که فقط وابستگی‌های سرویس در ایمیج قرار بگیرد تا اصلی‌ترین ماهیت کانتینر که سبک بودن آن است حفظ شود. معمولا طراحی و تعریف ساختار کانتینر فایل برعهده افرادی است که ساختار سرویس و اپلیکیشن را به خوبی می‌شناسند و از وابستگی‌های آنها کاملا آگاهی دارند. (تصویر ۲)

امروزه شرکت‌های بزرگی در سراسر دنیا برای استفاده از سرویس خود به کاربران نسخه کانتینر خود را در قالب ایمیج ارائه می‌کنند.

تصویر ۲



سایز این ایمیج‌ها براساس نوع سرویس متفاوت خواهد بود. بعضی از این ایمیج‌ها به عنوان Base Image سرویس خاصی استفاده می‌شود که معمولا حجم بسیار کمی دارند به عنوان مثال Alpine که حدود ۵ مگابایت است به عنوان یکی از پرطرفدارترین گزینه‌ها برای Base Image به حساب می‌آید.

تاثیر کانتینر در چرخه توسعه و تولید نرم‌افزار

استفاده از کانتینر در صورت رعایت شیوه‌ها و دستورالعمل‌های توصیه شده[3] در کنار سبک بودن می‌تواند فواید بسیار زیادی برای سازمان‌ها داشته باشد که نه تنها باعث تسریع چرخه توسعه و تولید نرم‌افزار می‌شود بلکه می‌تواند هزینه‌های زیرساخت را نیز کاهش دهد. ذات سبک کانتینر را می‌توان دلیل اصلی تمامی مزایای آن دانست زیرا نه تنها میزان مصرف منابع کاهش می‌یابد، بلکه سرعت انجام بسیاری کارها و فرآیندها نیز افزایش می‌یابد. کانتینر به علت وابستگی بسیار پایین به لایه سیستم‌عامل به راحتی قابل انتقال از یک محیط به محیط دیگر است بدین صورت که می‌توان از یک کانتینر اکسپورت گرفت و در قالب ایمیج به محیط دیگر منتقل کرد. سرعت ساخت و ایجاد کانیتنر و رسیدن به مرحله سرویس‌دهی بسیار بالا است و بعضی اوقات در حد چند ثانیه انجام می‌شود. در جدول زیر تفاوت‌های اصلی کانتینر و ماشین‌های مجازی ارائه شده است.

در پلتفرم‌های سنتی بدون کانتینر، عدم یکسان بودن محیط‌هایی که اپلیکیشن در آن توسعه، تولید و عملیاتی می‌شود، اغلب می‌تواند مشکل‌ساز باشد و بارها دیده شده که سرویس در محیطی به درستی کار کرده در حالی که در محیط دیگر عملکرد سرویس مختل شده است. بعضی از تغییرات بزرگ در سرویس معمولا منجر به تغییراتی در سطح سیستم‌عامل هم می‌شود. به عنوان مثال فرض کنید سرویسی برای اجرا نیاز به OpenJdk با ورژن ۸ دارد که بر روی همه محیط‌ها نصب شده، اما در نسخه جدید این نیاز به ورژن ۹ تغییر پیدا می‌کند و این تغییر باید بر روی تمامی محیط‌ها اعمال شود که این امر در صورت بالا بودن تعداد محیط‌ها خود می‌تواند کار سخت و زمان‌بری باشد ولی مشکل اصلی زمانی پیش می‌آید که سرویس دیگری در این محیط‌ها همچنان به همان ورژن ۸ برای اجرا نیاز داشته باشد. در چنین شرایطی مدیریت وابستگی‌ها به شدت امری پیچیده خواهد بود و در بیشتر اوقات به تعارض وابستگی[4] ختم خواهد شد. مشکلات این‌چنینی، باعث تاخیر در بسیاری از فرآیندها به خصوص به‌روزرسانی سرویس می‌شود.

از آنجایی که تکنولوژی کانتینر کم‌ترین وابستگی را به لایه سیستم‌عامل دارد، دغدغه عدم یکسان بودن محیط‌ها به حداقل خود می‌رسد به این دلیل که تمامی وابستگی‌ها در قالب ایمیج در دل کانتینر قرار دارد. به عنوان مثال یک کانتینر می‌تواند بر روی بستر OpenJdk 8 سرویس‌دهی کند در حالی که کانتینر دیگر در همان ماشین در حال سرویس‌دهی بر روی OpenJdk 9 باشد. وابستگی‌های این دو کانتینر کاملا از هم جدا بوده و هیچ‌گونه تعارضی باهم نخواهند داشت.

حال تیم توسعه بدون نگرانی از مشکلات محیط سنتی می‌تواند تمرکز خود را فقط بر روی طراحی سرویس بگذارد و آن را در محیط‌های مختلف (حتی لپ‌تاپ) توسعه داده و تست کند. در همین حال تیم عملیات با فراغ بازتر و عدم نگرانی از وابستگی‌های سرویس، می‌تواند بر روی مواردی مانند مدیریت محیط عملیات، مانیتورینگ، مقیاس‌پذیری، امنیت سیستم‌عامل و غیره متمرکز بوده تا در صورت موفقیت‌آمیز بودن عملیات تست، در سریع‌ترین زمان ممکن میزبان سرویس عملیاتی باشد. بالا رفتن کیفیت همکاری بین این دو تیم منجر به تسریع در چرخه توسعه و تولید نرم‌افزار می‌شود.

کلام پایانی

امروزه کانتینر در حوزه‌های مختلف IT نقشی پررنگ و غیرقابل کتمانی را ایفا می‌کند. به جرات می‌توان گفت پیدایش کانتینر نه تنها باعث تسریع بسیاری از امور در حوزه نرم‌افزار شده بلکه باعث گسترش تکنولوژی‌های مختلف مانند سرویس‌های ابری در سرتاسر دنیا نیز شده است به طوری که بسیاری از سرویس‌ها بدون اینکه بدانیم از تکنولوژی کانتینر استفاده می‌کنند. شرکت MarketsandMarkets طی پژوهشی که در سال ۲۰۱۷ انجام داد، ارزش بازار اپلیکیشن‌هایی که از پلتفرم کانتینر استفاده می‌کنند را تا سال ۲۰۱۸ حدود ۱.۲ میلیارد دلار تخمین زد و این ادعا را مطرح کرده است که تا سال ۲۰۲۳ سرویس‌های زیادی به محیط‌های کانتینری خواهند پیوست تا جایی که ارزش این بازار حدود ۴.۹۸ میلیارد دلار خواهد شد.

پی‌نوشت

۱- Isolation Level
۲- OS Level Virtualization
۳- Best Practices
۴- Dependency Conflict

	VM	Container
Weight	Heavy Wight(GB)	Light Wight (MB)
Software Dependencies	High level of dependency. Might face dependency conflict	All dependencies are packaged into container.
Portability	Hard to transfer	Easy to transfer
Service Isolation	High Cost	Low cost
Scalability	Hard to scale	Easier to scale
Creation & Spin up time	Takes long time (min)	Takes Short time (ms /sec)
Service deployment	Takes longer time and required dependency management	Short time as everything is packaged

کاموالت؛ راهکار جامع محافظت از داده‌ها در محیط‌های کوبرنتیس

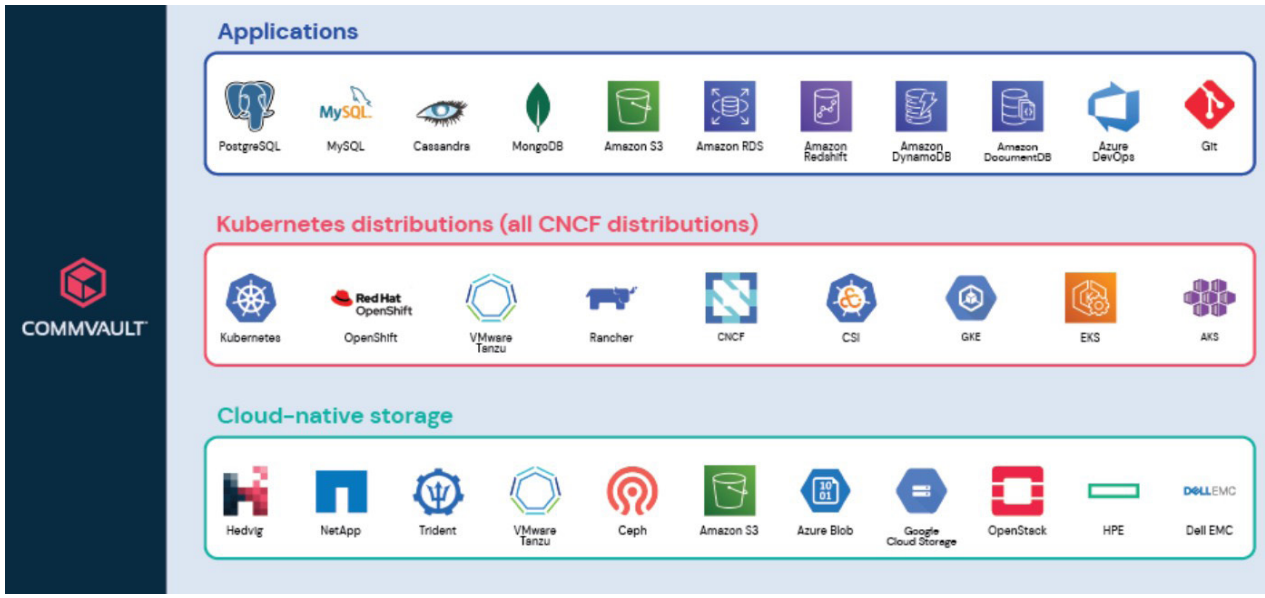
محمد سلطانی

بر اســاس مطالعات و پیش‌بینی‌های انجام شده توسط موسسه گارتنر، اســتفاده از نرم‌افزارهای مبتنی بــر کانتینر در محیط‌هــای عملیاتی از ۳۵درصد در ســال ۲۰۱۹ به ۸۵ درصد در سال ۲۰۲۵ افزایش خواهد یافت. در این سال تعداد کانتینرها نســبت به الان، حدود ۶ برابر شده و تعداد آنها به بیش از یک و نیم میلیارد می‌رسد. در حال حاضر حدود ۷۰ درصد از ســازمان‌ها از کوبرنتیس اســتفاده می‌کنند و بیش از نیمی از آنها، نرم‌افزارهای Stateful را در محیط عملیاتی مبتنی بر این بستر، استقرار داده‌اند. در گذشته این اعتقاد وجود داشــت که نرم‌افزارهایی که بر روی پلتفرم کوبرنتیس اجرا می‌شوند، می‌بایست دارای معماری Stateless باشــد اما با تغییرات تکنولوژی، به ویژه تحولاتی که در ســال ۲۰۲۰ رخ داد، این باور دگرگون شده و در حال حاضر پشتیبانی از ذخیره‌سازی و نرم‌افزارهای Stateful در کوبرنتیس به بلوغ رسیده است. قابلیت‌های HA و Replication داده‌هــا در کوبرنتیــس، احتمال از دست رفتن داده‌ها در مقابل اقدامات خرابکارانه و اشــتباهات ســهوی و انســانی را پوشش نمی‌دهد؛ از این رو در این محیط‌ها هم نیاز به راهکارهای بکاپ وجود خواهد داشــت. برای تهیه بکاپ از نرم‌افزارهای مدرن که تحت عنوان نرم‌افزارهای Containerized یا Cloud-Native هم شــناخته می‌شوند و بسیاری از آنها در کلاســترهای کوبرنتیس اســتقرار می‌یابند، چالش‌ها و نیازمندی‌هایی وجود دارد که موجب می‌شــوند نتوان از نرم‌افزارهای ســنتی بکاپ استفاده نمود:

ماهیت دینامیک: یکی از مزایای نرم‌افزارهای مدرن، ماهیت دینامیک آنهاســت. برای توزیع بهتر بار و پاسخگویی به افزایش بار، ممکن است کانتینرها بر روی نودهای مختلف Reschedule یا Auto-Scale شــوند و کامپوننت‌های مختلف به صورت مداوم در حال اضافه و حذف شــدن هســتند؛ از این رو نمی‌تــوان مانند محیط‌های ســنتی، سرورها و ماشــین‌ها را به صورت ثابت لیســت کرده و تعیین کنیــم کدامیک در چه زمانی بکاپ گرفته شوند.

رویکــرد Shift-Left در DevOps: در ایــن رویکرد ســعی می‌شــود حتی‌الامکان بخشی از تست‌ها و فعالیت‌ها در سمت چپ حلقه دواپس رخ دهند. در چنین شــرایطی، برنامه‌نویســان نیازمندی‌هــای نرم‌افزاری و زیرســاختی را به

تصویر ۱



صورت as-Code تعریف می‌کنند و ترجیح می‌دهند، زمان‌بندی‌ها و نحوه تهیه بکاپ از داده‌ها را به جای اینکه پس از استقرار نهایی، توسط ادمین‌های بکاپ تعریف شود، خودشان مشخص کنند. علاوه بر این داده‌هایی در سمت چپ حلقه تولید می‌شود که خود نیازمند محافظت است.

تعداد زیاد کامپوننت‌ها: نرم‌افزارهای سنتی تعدادی ماشین مجازی را در بر می‌گیرند اما در نرم‌افزارهای مدرن تعداد زیادی کامپوننت مانند والیوم‌ها، Podها، پیکربندی‌ها، سکرت‌ها، ایمیج‌ها و غیره وجود دارد که ممکن است تعداد آنها به هزاران مورد برسد که همگی نیازمند بکاپ‌گیری هستند.

محیط‌ها و کلاسترهای مختلف: در نرم‌افزارهای مدرن، محیط‌های مختلف توسعه، تست، Stage، پروداکشن و غیره و طبیعتاً متناظر با آنها، کلاسترهای کوبرنتیس وجود دارد که براساس پایپلاین‌های تعریف شده در چرخه CI/CD ایجاد شده‌اند و نرم‌افزارهای بکاپ می‌بایست با همه این محیط‌ها در ارتباط باشند.

پدیده Polyglot Persistence: پدیده «ماندگاری چندزبانه» به این معناست که نرم‌افزارهای مدرن برای برآورده کردن نیازهای خود در حوزه ذخیره و مدیریت داده، بسته به نیاز از تکنولوژی‌های مختلفی مانند پایگاه‌های داده رابطه‌ای، NoSQL، گراف، Message Queueها، In-Memory و /Batch Data Streaming استفاده می‌کنند و رایج شدن آن در محیط‌های کوبرنتیس به شدت در حال افزایش است. نرم‌افزارهای بکاپ علاوه بر دیسکاوری خودکار آنها، می‌بایست این تنوع بالا را پشتیبانی نمایند.

عدم استفاده از Agent: در محیط‌های فیزیکی بر روی سرورها ایجنت نصب می‌شود و در محیط‌های مجازی اغلب از APIهایی که زیرساخت مجازی‌سازی در اختیار قرار می‌دهد برای تهیه بکاپ استفاده می‌شود اما این امکانات در محیط‌های کوبرنتیس وجود ندارد.

محیط‌های هیبرید: بسیاری از سازمان‌ها اگر چه در لایه اپلیکیشن از کانتینر استفاده می‌کنند؛ اما همچنان بر این باورند که برای برخی از نرم‌افزارهای خود در لایه پایگاه داده بهتر است از روش‌های سنتی استفاده کرده و مثلاً داده‌های خود را در بانک اطلاعاتی اوراکل یا MSSQL ذخیره کنند. این رویکرد سبب می‌شود که یک محیط ترکیبی وجود داشته باشد و می‌بایست نرم‌افزار بکاپ این محیط را به صورت مجتمع و متمرکز پشتیبانی نماید.

مجوزها و Quiesce: می‌دانیم به منظور تهیه بکاپ در یک نقطه زمانی مشخص می‌بایست فعالیت‌های IO ساکت (متوقف) شود و انجام فعالیت‌های اینچنینی در محیط‌های کوبرنتیس نیازمند مجوزهای خاصی است. با توجه به تعدد اکانت‌ها و مجوزها، نرم‌افزارهای بکاپ مدرن می‌بایست این موضوع را به خوبی هندل کرده و حتی‌الامکان از قابلیت‌های تهیه Snapshot در لایه ذخیره‌سازی استفاده کنند.

بازیابی فاجعه: راهکارهای بکاپ مدرن می‌بایست علاوه بر تهیه بکاپ و بازیابی در سایت اصلی، قابلیت‌هایی را برای DR در کلاسترهای کوبرنتیس در سایت‌ها/ابرهای دیگر دارا باشند.

به منظور پاسخ به چالش‌ها و نیازمندی‌های فوق، به راهکارهای پشتیبان‌گیری مدرنی نیاز است که به جای اینکه مبتنی بر زیرساخت باشد، در اصطلاح Application-Centric باشد. این راهکارها تحت عنوان K8s-Native Backup Solutions شناخته می‌شوند. یکی از مطرح‌ترین این راهکارها، نرم‌افزار کامولت است. نرم‌افزار کامولت از سال ۲۰۱۱ در گزارش‌های مقایسه‌ای موسسه‌های گارتنر و فورستر در ناحیه Leaders قرار داشته و در بسیاری از سال‌ها دارای رتبه اول در حوزه بکاپ/ریکاوری در مراکز داده بوده است. این نرم‌افزار، راهکار یکپارچه و جامع محافظت از داده‌هاست که دارای قابلیت‌های متنوع و منحصر به فردی است؛ از جمله در پوشش انواع فایل‌سیستم‌ها، زیرساخت‌های مجازی‌سازی، بانک‌های

اطلاعاتی، بسترهای ابری و ویژگی‌هایی مانند آرشیو فعال، Deduplication، بازیابی فاجعه، Snapshot Backup و غیره. کاموالت بکاپ/ریستور ریزدانه داده‌های نرم‌افزارهای مدرن به صورت Application-Centric را با قابلیت‌های مهاجرت بین کلاسترها و DR در محیط‌های کوبرنتیس پشتیبانی می‌کند. بر اساس تحقیق و بنچمارک موسسه GIGAOM در سال ۲۰۲۱ در حوزه راهکارهای محافظت از داده‌ها در کوبرنتیس، نرم‌افزار کاموالت در نمودار راداری این موسسه، در ناحیه لیدرها قرار داشته و در محور بلوغ-خلاقیت دارای بیشترین امتیاز است.

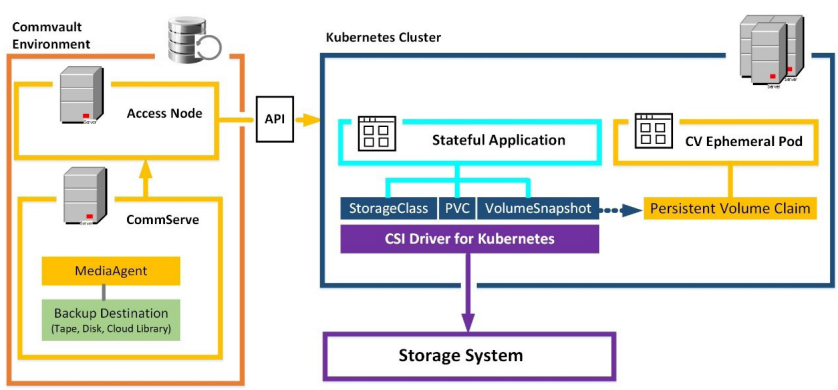
در اکوسیستم مبتنی بر کوبرنتیس معمولاً سه لایه نرم‌افزار/دیتا سرویس، توزیع‌های مختلف کوبرنتیس/اورکستریتور و ذخیره‌سازی وجود دارد که همگی توسط کاموالت پشتیبانی می‌شوند. (تصویر ۱)

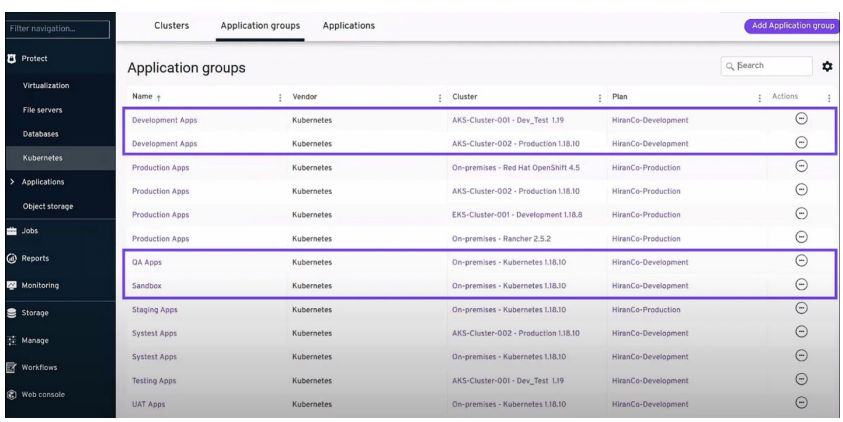
برخی از مهم‌ترین ویژگی‌های کاموالت در حوزه کوبرنتیس عبارتند از:

- دیسکاوری خودکار و محافظت از نرم‌افزارها مبتنی بر نام، لیبل، والیوم و غیره.
- پشتیبان‌گیری از آبجکت‌ها و ریسورس‌های کلاسترهای کوبرنتیس، سکرت‌ها، کانفیگ‌مپ‌ها، Namespaceها، کلاس‌های استوریج، PV و PVCها، نرم‌افزارهای مبتنی بر Helm-Chart، رجیستری ایمیج‌ها، گواهی‌ها، etcd و غیره.
- قابلیت بازیابی اپلیکیشن‌ها به زمان‌های قبل در همان کلاستر و کلاسترهای متفاوت، داده‌ها در PVC اصلی یا PVC متفاوت.
- بکاپ مبتنی بر اسنپ‌شات از PVهای مبتنی بر اسکریپت‌های مختلف تهیه شده برای پشتیبان‌گیری از نرم‌افزارهای مشهور مانند MySQL و PostgreSQL و همچنین قابلیت‌های درایورهای CSI سیستم‌های ذخیره‌سازی
- قابلیت بازیابی کامل کلاستر در صورت بروز خرابی در کل کلاستر و یا بازیابی کنترل پلین با استفاده از بازیابی گواهی‌ها و اسنپ‌شات‌های etcd.
- انتقال داده‌ها به سایت پشتیبان/ابر دیگر به منظور بازیابی فاجعه با استفاده از الگوریتم‌های فشرده‌سازی و Deduplication به منظور کاهش پهنای‌باند خطوط ارتباطی.
- پشتیبانی از انواع ذخیره‌سازی‌های ابری مبتنی بر پروتکل S3، دیسک و نوار مغناطیسی به عنوان مقصد داده‌های بکاپ.
- قابلیت محافظت از داده‌های پشتیبان در مقابل Ransom Ware.
- قابلیت برنامه‌نویسی و یکپارچه‌سازی با اورکستریتورها و نرم‌افزارهای مانیتورینگ مبتنی بر API.

همانگونه که در تصویر ۲ مشاهده می‌شود، زیرساخت کاموالت با کلاستر کوبرنتیس از طریق API Server ارتباط برقرار می‌کند و این ارتباط از طریق Access Node برقرار می‌شود. سرور CommServe نقش مدیریت زیرساخت بکاپ و جاب‌ها را برعهده داشته و داده‌ها از طریق Media Agent بر روی مدیای بکاپ، ذخیره‌سازی می‌شوند. برای نمونه، به منظور تهیه بکاپ از PVC، با شروع جاب بکاپ، کاموالت با استفاده از CSI یک اسنپ‌شات از PVC می‌گیرد. در صورت پشتیبانی سیستم ذخیره‌سازی، این اسنپ‌شات می‌تواند Array-Based باشد. سپس این اسنپ‌شات به یک Pod افمرال کاموالت که به صورت خودکار ایجاد شده است، متصل شده و داده‌ها از طریق MA بکاپ‌گیری می‌شوند. در حال حاضر تعداد بسیار زیادی از وندورهای سیستم‌های ذخیره‌سازی CSI را پشتیبانی می‌کنند. برای نمونه می‌توان به سیستم‌های ذخیره‌سازی یونیتی، نیمبل، پریمرا، پاوراستور، Ceph و Vsphere اشاره نمود. یکی از کاربردهای بکاپ در چرخه DevOps می‌تواند این سناریو باشد: برنامه‌نویس قصد دارد یک تغییر نسبتاً بزرگ را در کد اعمال نماید و بنا به هر دلیل می‌خواهد از نتایج آن در سیستم عملیاتی مطمین شود. او می‌تواند یک کپی از نرم‌افزار پروداکشن را در محیط توسعه (با استفاده از قابلیت ریستور Out-of-Place در کاموالت مثلاً در کلاستر محیط Dev/Test) به صورت کاملاً مجزا داشته باشد و پس از اعمال تغییرات و اطمینان از تاثیرات تغییر، آن را حذف کند.

تصویر ۲





رمیس؛ همسفر تحول دواپس

هدیه رزازان

یکی از عوامل کلیدی موفقیت یک سازمان در پیاده‌سازی دواپس، درک این حقیقت است که دواپس یک مقصد نیست، بلکه یک سفر است. چندان اغراق‌آمیز نیست اگر این سفر را سفر حماسی یا سفر قهرمانی سازمان نامید. در طول این سفر، سازمان در مسیر رشد و بلوغ دواپس گام برمی‌دارد. بهترین تشبیه برای چنین سفری، سفر از کوهپایه به سمت قله در مسیری مارپیچ که به گرداگرد کوه کشیده شده است. با هر چرخش به دور کوه، یک مرحله به قله نزدیک‌تر می‌شویم و همواره در طول مسیر نیز می‌توان از سفر لذت برد. مشابه با چنین سفری، در سفر دواپس نیز در هر مرحله و چرخش، با چالش‌های متفاوتی روبرو شده و نیازمند دانش، تخصص و ابزارهای متفاوت هستیم. این سفر، DevOps Adoption یا تحول دواپس نام دارد. تحول دواپس یک مسیر مستقیم رو به بالا نیست بلکه مسیری مارپیچ مانند است که در هر چرخه از آن، سازمان به سطح بلوغ بالاتری دست پیدا می‌کند.

از مزایای پیاده‌سازی دواپس، سخن بسیار گفته شده، اما این سکه روی دیگری نیز دارد: چالش‌های متعدد پیاده‌سازی دواپس که می‌تواند منجر به عدم تحقق اهداف تعیین شده گردد. برخی از مهم‌ترین این چالش‌ها عبارتند از:

- **چالش‌های حوزه فرهنگ و سازمان**
  - عدم وجود دیدگاه یکسان به دواپس
  - رهبری و حاکمیت دواپس در سازمان
  - مقاومت در برابر تغییرات و تغییر فرهنگ سازمان
  - حرکت از تیم‌های تخصصی به سمت تیم‌های چندعملکردی
- **چالش‌های حوزه نیروی انسانی**
  - تجربه پایین در صنعت فناوری اطلاعات در خصوص دواپس
  - کمبود نیروی انسانی
- **چالش‌های حوزه تکنولوژی و ابزار**
  - حرکت از زیرساخت‌ها و نرم‌افزارهای قدیمی و مونولوتیک به سمت میکروسرویس
  - تمرکز زیاد و نابه‌جا بر ابزارها
  - انتخاب ابزارها با توجه به تعدد و تنوع آنها
  - ابزارهای مجزا و غیریکپارچه Dev و Ops (پدیده تصادم زنجیره ابزارها) و یکپارچه‌سازی آنها در یک معماری واحد
  - پیاده‌سازی، خودکارسازی و حداقل‌سازی فعالیت‌های دستی

بهره‌گیری از تجارب مشاور دواپس، توان سازمان در مدیریت چالش‌های فوق را افزایش می‌دهد. از جمله دلایل نیاز به مشاور در ایجاد بستر دواپس می‌توان به موارد زیر اشاره کرد:

- همگراسازی و یکسان‌سازی دیدگاه‌های ذی‌نفعان در حوزه دواپس
- کمک به ایجاد بستر رهبری و حاکمیت دواپس در سازمان
- تجربه پایین در صنعت فناوری اطلاعات در خصوص دواپس و استفاده از تجربیات بیرون سازمانی
- ارزیابی سطح بلوغ سازمان مبتنی بر یک مدل بلوغ مشخص توسط یک عامل بی‌طرف بیرونی
- ایجاد بستر اندازه‌گیری موفقیت با پارامترهای درست و قابل ردیابی مبتنی بر یک مدل بلوغ مشخص
- انتخاب ابزارهای مناسب از میان زنجیره ابزار بسیار متنوع و متعدد
- تربیت نیروهای فنی مورد نیاز (آموزش‌های تخصصی با توجه به زنجیره ابزار انتخاب شده)
- کمبود نیروی انسانی و استفاده از نیروهای انسانی بیرون سازمانی
- اهمیت بسیار بالای فرهنگ در دواپس و نیاز به بازوی کمکی در فرهنگ‌سازی
- بسترسازی Practiceهای دواپس از زاویه‌ای فراتر از یک شرکت/تیم توسعه‌دهنده نرم‌افزار
- ایجاد بستر ثبت و نگهداری دانش مرتبط با دواپس

متدولوژی و رویکرد رمیس در تحول دواپس

یکی از مهم‌ترین عوامل موفقیت پیاده‌سازی دواپس اتخاذ رویکردی جامع است که با تحت پوشش قرار دادن تمامی جوانب و اجزای دواپس، چالش‌ها را به حداقل رسانده و شانس سازمان را در پیاده‌سازی موفق دواپس افزایش دهد. رمیس با تکیه بر مطالعات وسیع و تجارب اندوخته شده در این زمینه، متدولوژی تحول دواپس مبتنی بر مدل فرآیندی حلزونی دوسطحی، تدوین و طراحی کرده که در شکل نشان داده شده است.

متدولوژی تحول دواپس رمیس شامل یک چرخه اصلی و یک چرخه کوچک‌تر که در دل آن قرار گرفته، است. این چرخه‌ها شامل نظام‌های زیر هستند:

- همگرایی
- تحلیل
- چرخه درونی
  - جهت‌گیری
  - پیاده‌سازی
- اندازه‌گیری و بازخورد

در ادامه به بررسی فازهای فوق پرداخته شده است.

همگرایی

یکی از عوامل اصلی در موفقیت تحول دواپس در یک سازمان، همراهی و حتی بیش از آن تعهد نسبت به دواپس در ذی‌نفعان کلیدی سازمان است. برای نیل به این مقصود لازم است هم‌کلامی و درک یکسانی از دواپس در

بین افراد وجود داشته باشد. برای این منظور گردهمایی دو/چند روزه‌ای با حضور ذی‌نفعان، مدیران و کارشناسان کلیدی سازمان که با نظر کارفرما و پیشنهاد مشاور مشخص می‌شوند، با هدف ایجاد هم‌کلامی در خصوص دواپس و اهداف پروژه، برگزار می‌گردد. در طول برگزاری گردهمایی، مفاهیم دواپس، اصول و مزایای آن تشریح شده و درک متقابلی از فرهنگ و بسترهای سازمان ایجاد می‌شود. با مشارکت پویای افراد در مباحث، به درک دیدگاه‌ها، تعهدات و مشکلات ذی‌نفعان کلیدی و سازمان دست یافته می‌شود. همچنین کمیته راهبری ایجاد خواهد شد که مسئولیت پیشبرد تحول دواپس در سازمان را برعهده خواهد داشت.

تحلیل

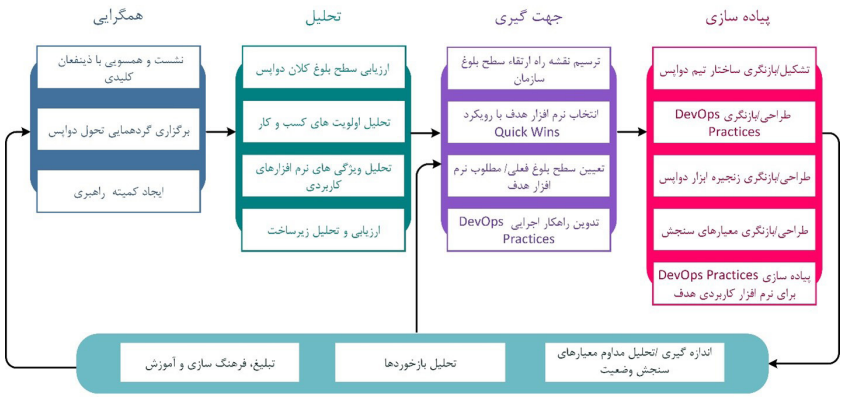
پیش از شروع یا ادامه مسیر لازم است مشخص شود سازمان در چه سطح بلوغی از دیدگاه دواپس قرار دارد و همچنین سطح بلوغ مطلوب دواپس در این تکرار (Iteration) چیست. در ابتدا مدل مرجع مناسب جهت ارزیابی سطح بلوغ مشخص شده، پس از آن از طریق برگزاری جلسات هم‌فکری، تکمیل پرسشنامه‌ها و بررسی مستندات و پیشینه موجود، سطح بلوغ فعلی سازمان مشخص می‌شود. همچنین لازم است، استراتژی‌های کلان سازمان، برنامه‌ها و اهداف آتی، اولویت‌های کسب‌وکار و همچنین توانمندی‌های سازمان بررسی و تحلیل شود تا بتوان سطح بلوغ مطلوب سازمان را نیز مشخص کرد. در هر تکرار از تحول دواپس، لازم است اولویت‌های کسب‌وکار مورد تحلیل و ارزیابی قرار بگیرد. نرم‌افزارهای کاربردی از دیدگاه‌های مختلف از جمله نیازمندی زمان ارائه به بازار، میزان و تواتر تغییرات، معماری نرم‌افزار، رویکرد مدیریت محصول و... مورد بررسی و تحلیل قرار خواهند گرفت. همچنین ضروری است لایه‌های مختلف زیرساخت نیز مورد شناسایی و تحلیل قرار بگیرند. بررسی‌ها و تحلیل‌های انجام شده به منظور تعیین اهداف تحول دواپس در هر تکرار است.

جهت‌گیری

در این مرحله، ضروری است اهداف اصلی ارتقاء سطح بلوغ دواپس در چرخه جاری مشخص شود. لازم به ذکر است اهداف معادل سطح بلوغ مطلوب نیستند، ولی در راستای رسیدن به آن تعریف می‌شوند. به بیان دیگر سطح بلوغ مطلوب در یک تکرار قابل دستیابی نیست، بنابراین تکرارها به صورت مجزا ولی هم راستا هدف‌گذاری می‌شوند. در ادامه با رویکرد Quick wins نرم‌افزارهای کاربردی هدف تعیین می‌شود. در رویکرد Quick Wins ابتدا اهدافی ساده و سهل الوصول تعریف شده، سپس توسعه داده می‌شود. با این روش، نتایج مثبت و سریعی که در ابتدا حاصل می‌شود، خود مبلغ و مشوق دواپس در سازمان خواهد بود. همچنین به درک چالش‌ها نیز کمک خواهد کرد. پس از تعیین نرم‌افزار هدف، به صورت خاص سطح بلوغ جاری و مطلوب آن تعیین خواهد شد. همچنین لازم است برای DevOps Practiceها نیز راهکارهای اجرایی تدوین شود.

پیاده‌سازی

در مرحله پیاده‌سازی، اقدامات اجرایی در راستای برآورده‌سازی اهداف تعیین شده، صورت می‌پذیرد. در ابتدای امر ساختار تیم اجرای دواپس طراحی شده یا ساختار موجود مورد بازنگری قرار می‌گیرد. در ادامه لازم است براساس راهکار اجرایی DevOps Practices، مستندات طراحی Practiceها تدوین یا مورد بازنگری و بهبود قرار گیرد. در هر مرحله از سطح بلوغ بخشی از DevOps Practices اولویت بالاتری پیدا می‌کند. بر این اساس طراحی انجام یا بازنگری شده، سپس به پیاده‌سازی آنها اقدام می‌گردد. به منظور اطمینان از نحوه انجام و پیشرفت پیاده‌سازی دواپس، بایستی بتوان آن را براساس معیارهای سنجش، متناسب با اهداف و سطح بلوغ ارزیابی کرد. تعریف یا بازنگری معیارهای سنجش در این مرحله انجام می‌شود.



اندازه گیری و بازخورد

اندازه‌گیری و بازخورد

پس از هر مرحله پیاده‌سازی، لازم است مشخص شود چه میزان از اهداف تعیین شده، برآورده شده‌اند. دواپس تا چه میزان بر بالا بردن کارایی، قابلیت اطمینان، چابکی، رضایت مشتریان و غیره تاثیرگذار بوده است. همچنین بازخوردهای ذی‌نفعان نیز مورد تحلیل و ارزیابی قرار خواهد گرفت. نتایج حاصل بر برنامه‌ریزی چرخه‌های بعدی تاثیر مستقیم خواهد داشت. همچنین لازم است در راستای تحکیم فرهنگ دواپس سازمان، نتایج پیاده‌سازی و تاثیرات آن با سایر بخش‌ها به اشتراک گذاشته شده و تشریح و تبیین شود. راه‌اندازی سیستم پایش مداوم و سیستم تعامل و اعلام و دریافت بازخورد نیز در این مرحله انجام می‌شود. آموزش و ارتقاء دانش افراد نیز از دیگر فعالیت‌های مهم است. پس از انجام هر چرخه پیاده‌سازی دواپس می‌توان به درک بهتری از چالش‌ها و از سوی دیگر مشوق‌های پیاده‌سازی دواپس در سازمان رسید. تدوین استانداردهای تحول دواپس نیز از فعالیت‌های کلیدی است که لازم است در هر چرخه انجام و بهبود یابد.

شرکت‌های بزرگی که DevOps را پیشه خود کردند

دیواری که دیگر در میان نیست

نقطه اشتراک روایت‌هایی که شرکت‌های مختلف از چرایی هجرتشان به DevOps می‌گویند، دلسردی از سیستم‌های سنتی است. DevOps آنقدر عملکرد شرکت‌ها را به طور اساسی متحول می‌کرد که در کوتاه‌مدت با استقبال زیادی مواجه شد. ساختارهای مدیریتی متمرکز در این رویکرد بدل به سیستم مدیریتی سازگاری می‌شود که چرخه‌های توسعه در آن کوتاه‌تر است، سرعت اجرا و پیاده‌سازی بالاتر است و مدت‌زمان رساندن محصول به بازار یا ارائه آن نیز سریع‌تر است. در عین حال از آنجا که چنین مسیری نیاز به ارتباط زیاد با همکاران، همکاری و نوآوری دارد، می‌تواند به تغییر فرهنگی جدی منجر شود.

در این گزارش نگاهی می‌اندازیم به مسیری که DevOps در عمر کوتاه خود طی کرده است. همچنین به صورت موردی بررسی می‌کنیم که چرا شرکت‌های بزرگ دنیا DevOps را انتخاب کردند.

DevOps چیست؟

DevOps توسعه نرم‌افزار و عملیات (Operations) را با هم ترکیب می‌کند. توسعه در واقع وقتی اتفاق می‌افتد که توسعه‌دهندگان نرم‌افزار اپلیکیشنی را برنامه‌نویسی و آزمایش می‌کنند. تیم عملیات نیز مسئول اجرا و نگهداری اپ روی سرور هستند. ساده‌ترش این است که آدم‌ها در کنار هم و با هم کار می‌کنند تا نرم‌افزار امنی را طراحی کنند، بسازند و اجرا کنند؛ نرم‌افزاری که سرعت و کیفیت قابل توجهی هم داشته باشد.

رویکرد DevOps متکی به تغییر ذهنیت و رویکرد فرهنگی فناوری اطلاعات است؛ رویکردی که می‌خواهد فرهنگی را ایجاد کند که در آن همکاری بهبود می‌یابد و همدلی و مسئولیت بیشتر تقویت می‌شود.

راه نجات شرکت‌های بزرگ

در ادامه نگاهی به چند شرکت بزرگ می‌اندازیم که با تجربه این تغییر و تحول ناشی از DevOps مشکلات خود را حل کردند و توانستند محصولات خدماتشان را پایدارتر، منعطف‌تر، باکیفیت‌تر و سریع‌تر ارائه دهند.

آمازون: چابکی

آمازون در ۲۰۰۱ شرکت کوچکی با معماری مونولیتیک سنتی بود: چهارچوبی که در آن همه فرایندها با هم یکی و به عنوان یک خدمت اجرا می‌شدند. به مرور زمان با رشد شرکت و افزایش فایل‌ها، مقیاس‌بندی، حفظ و نگهداری و به‌روزرسانی برنامه‌ها در سرورهای فیزیکی، بسیار

بغرنج شد.
یکی دیگر از مشکلات آمازون، پیش‌بینی خرید تجهیزات برای برطرف کردن مشکلات ترافیک بود. حدود ۴۰ درصد از ظرفیت سرور آمازون بلااستفاده می‌ماند، به‌ویژه در اوقات تعطیل که ترافیک سه‌برابر می‌شد و وضعیت بدتر و پولی که صرف این سیستم سنتی شده بود، هدر می‌رفت. اما وقتی این خرده‌فروش آنلاین سراغ خدمات وب آمازون یا AWS (Amazon Web Services) رفت، مهندسان توانستند ظرفیت را کاهش و افزایش دهند. با این کار علاوه بر کاهش هزینه‌هایی که صرف سرور می‌شد، این امکان برای توسعه‌دهندگان فراهم شد که کد خودشان را هروقت که خواستند و نیاز بود، اجرا کنند.
افزایش درآمد و کاهش مدت‌زمان خروجی، نتیجه این رویکرد چابک برای آمازون بود.
در حال حاضر آمازون از معماری مایکروسرویس استفاده می‌کند که ابتدا مشتری درخواستی می‌فرستد، سپس درخواست بررسی می‌شود و به مایکروسرویس مناسب خود، ارائه می‌شود. هر مایکروسرویسی یک گروه هدف دارد. در آمازون سه نوع سرویس با نام‌های کاربران (Users)، Threads و پست‌ها (Posts) داریم.
ابزارها و فیچرها: توسعه‌دهندگان آمازون، تغییرات مداوم اما کوچکی را از طریق ابزارهای کنترل از جمله گیت و گیت‌هاب روی کدهایشان اعمال می‌کنند. به این ترتیب اشکالات رفع می‌شوند و نرم‌افزار عملکرد بهتری دارد.
AWS CodeDeploy یکی از چنین خدماتی است که فرایندها را پیگیری می‌کند. آمازون همچنین از آپولو (Apollo) استفاده می‌کند که مجموعه‌ای از نرم‌افزارها را در میان گروهی از هاست‌ها مستقر می‌کند.

اچ‌پی: بی‌نهایت کد

واحد LaserJet Firmware اچ‌پی (hp) در سال ۲۰۰۶ محصولات مختلفی مثل پرینتر و اسکنر تولید می‌کرد. اما تقریبا ۵ درصد از زمان صرف توسعه و پشتیبانی از فیچرهای جدید می‌شد و مابقی صرف برنامه‌ریزی، یکپارچه‌سازی و تست می‌شد.
این واحد بیش از ۴۰۰ توسعه‌دهنده در کشورهایی مثل ایالات متحده آمریکا، برزیل و هند داشت، اما سالانه فقط دو نسخه نرم‌افزاری منتشر می‌کرد. چون مدل‌های LaserJet، پایگاه‌های کد جداگانه داشت که منجر به ناکارآمدی‌های بی‌شمار می‌شد: باگ نرم‌افزارها از طریق تست منوال شش هفته بعد از نوشتن کد، کشف می‌شد. رفع مشکل یک کد، هفته‌ها طول می‌کشید و آن هم وابسته به کار زیاد و خسته‌کننده توسعه‌دهندگان بود. در این نقطه بود که نیاز به رویکرد جدیدی مطرح شد تا گره از کار باز شود.
راه‌حل چه بود: تیم اچ‌پی پایپ‌لاین یکپارچه‌سازی مداوم و تحویل پیوسته یا استقرار پیوسته (CI/CD) و همچنین خودکارسازی تست را اعمال کرد. اولین گام تیم ایجاد پلتفرمی بود که از همه مدل‌ها و محصولات پشتیبانی کند؛ رویکردی که به آن یکپارچه‌سازی مداوم (Continuous Integration) یا توسعه Trunk-Based می‌گویند و دردسر یکی کردن کدهای مختلف را از میان برداشت.
تیم همچنین تست خودکار را راه‌اندازی کرد که مدت‌زمان بررسی را کاهش می‌داد و در نتیجه باعث بهبود کیفیت محصول و افزایش سرعت دریافت بازخورد می‌شد. اچ‌پی در این مرحله از ابزاری به نام Stopped the Line استفاده می‌کند که در صورت مشکل کد، به توسعه‌دهنده هشدار می‌دهد. همه این اقدامات مبتنی بر DevOps منجر به ۱۰۰ تا ۱۵۰ کد و ۷۵هزار تا ۱۰۰هزار تغییر کد در یک روز شد.

نتفلیکس: آماده در برابر خطا

نتفلیکس با تغییر مدل کسب‌وکار خود از DVD فیزیکی به پخش ویدئویی مبتنی بر وب، مثل کشتی‌ای بود که در آب‌های ناشناخته پیش می‌رود. هنوز هیچ ابزار تجاری‌ای در دسترس نبود که به زیرساخت عظیم ابر (Cloud) شرکت اجازه دهد که پایدار و روان کار کند؛ به همین دلیل بود که نتفلیکس سراغ راه‌حل‌های متن باز رفت.
به کمک صدها توسعه‌دهنده داوطلب، Simian Army ساخته شد که مجموعه‌ای از ابزارهای خودکار است و زیرساخت نتفلیکس را آزمایش می‌کند؛ Simian Army امکان شناسایی فعال آسیب‌پذیری‌ها و رفع آنها را ممکن می‌سازد، آن هم قبل از اینکه برای مشتری دردسری ایجاد شود. نتفلیکس تا امروز به اتوماسیون و متن باز وفادار بوده و مهندسانش هر روز هزاران کد را اجرا می‌کنند. حالا دیگر شرکت کوچک سابق، آنقدر بزرگ شده که در سه‌ماهه آخر سال ۲۰۲۲، تعداد بیش از ۲۳۰ میلیون مشترک خدمات پولی، را به نام خود ثبت کند و از بزرگ‌ترین رسانه‌های جهان به حساب آید.
اما نتفلیکس نیز مثل دیگر سازمان‌ها و شرکت‌های آن دوره، معماری مونولیتیک را انتخاب کرده بود. مدیریت حجم بالای ترافیک مشترکان، نیاز به ابزارهای تجاری متعددی داشت. اولین اقدام نتفلیکس مهاجرت به معماری مایکروسرویس مبتنی بر ابر AWS بود. شرکت از ۷۰۰ مایکروسرویس استفاده کرد تا هر کدام بخشی از خدمات را تحت کنترل قرار دهند. در این معماری، تیم‌های مهندسی از همدیگر جدا می‌شوند، خدمات را می‌سازند و تست و اجرا می‌کنند. این اتعطاف در کار، باعث افزایش سرعت می‌شود.
آمادگی در برابر شکست: آمادگی بهترین راه‌حل در مقابل اتفاق‌های پیش‌بینی‌ناپذیر و شکست‌ها و خطاهاست. در این راستا نتفلیکس ابزاری به نام Chaos Monkey را ساخت که به تست و پایداری اپلیکیشن کمک می‌کند. Chaos Monkey همزمان با اینکه توسعه‌دهندگان در حال کارند، خطاهایی را به‌عمد و به طور نامنظم در سرورهای نهایی ایجاد می‌کند. نتفلیکس تلاش می‌کند با چنین ابزارهایی تحت عنوان Netflix Simian Army، راه‌حل‌هایی برای مشکلاتی بیابد که پیش‌بینی‌ناپذیرند.
می‌گویند اقدامات نتفلیکس در حوزه DevOps از بهترین‌هاست، زیرا تغییرات را در حین فرآیند توسعه اعمال می‌کند که خروجی‌اش برنامه‌های باکیفیت، درجه یک و منعطف است.
نتفلیکس همچنین یک ابزار مدیریت Container به نام تیتوس (Titus)، توسعه داده است که برنامه‌های موجود را بدون ایجاد تغییر در Container اجرا می‌کند. این ابزار، ظرفیت اشتراک‌گذاری منابع را مدیریت می‌کند و آن را با خدمات وب آمازون یکپارچه می‌سازد. تیتوس به نتفلیکس در زمینه سیستم‌های پخش، پیشنهاد و محتوا کمک می‌کند.

نگاهی به سبک مدیریت جن سون هوانگ، مدیر انویدیا

باید باخت تا پیروز شد

جن سون هوانگ (Jen-Hsun Huang) انویدیا را در سال ۱۹۹۳ تاسیس کرد و از همان اول تا امروز، مدیر این شرکت است که به طراحی واحدهای پردازش گرافیکی (GPU)، واسطهای برنامه‌نویسی کاربردی و همچنین واحدهای سامانه روی یک تراشه (SoC) برای رایانش موبایلی و بازار خودرو می‌پردازد. به‌کارگیری هوش مصنوعی باعث شد چیپ‌ها بدل به مغز رایانه‌ها، ربات‌ها و اتومبیل‌های خودران شوند. هوانگ می‌گوید در این بیست‌وشش سال پیچیدگی در گرافیک رایانه ۵۰۰ برابر بیشتر شده است.

نوآوری‌های بی‌امان شرکت او در این سال‌ها نتیجه داد: ارزش بازار شرکت انویدیا در فوریه ۲۰۲۲، با بیش از ۲۲ هزار و ۴۷۳ کارمند، ۶۱۸.۲ میلیارد دلار اعلام شد. تحلیلگران صنعت معتقدند یکی از دلایل اینکه هوانگ شاهد نتیجه فوق‌العاده بوده، این است که شم خوبی دارد، با کارمندانش همدل است و از آموزش در حوزه فناوری و علم پشتیبانی می‌کند. در مجموع تیم‌های شرکت او وضعیت خوبی دارند و موقعیتشان طوری است که به نظر می‌رسد دهه‌های آینده را نیز وقف پیشرفت فنی خواهند کرد.

معمولاً سبک مدیریت او به دلیل توجه بیشتر عموم به کسانی مثل جف بزوس یا ایلان ماسک، مورد توجه قرار نمی‌گیرد؛ اما بی‌گمان موفقیت انویدیا این غول نیمه‌رسانا و بازیکن بزرگ صنعت بازی، مرهون حضور اوست که الگوی «مثل رهبر عمل کن» را قبول ندارد و در مقابل از تواضعی استقبال می‌کند که در کنارش، اعتمادبه‌نفس درخشانی نیز دیده می‌شود.

فلسفه شکست

سبک رهبری هوانگ با تجربه شکست اولیه انویدیا آغاز شد: جن سون و شرکایش سرمایه جذب کردند، افراد مورد نیاز را استخدام کردند، محصول را توسعه دادند، اما محصولی که تمام هم‌وغم‌شان بود و به آن ایمان داشتند، نتیجه نگرفت. احتمالاً اگر اکثر رهبران به جای جنسون بودند، با تجربه چنین شکستی عقب می‌کشیدند؛ اما او ناامید نشد. خودش می‌گوید در چنین موقعیت‌های دشواری است که فرهنگ حقیقی و ارزش‌های اصلی شرکت ساخته می‌شود، چون اگر شرکتی ایده جذابی دارد، نباید تسلیم شود، باید پرورشش دهد، خود را سازگار کند و «در حالی که شکست را تجربه می‌کند، جلو برود».

در واقع «صداقت فکری» است که باعث می‌شود بپذیریم استراتژی‌ای شکست خورده است و سرمایه‌گذاری بیشتر نتیجه‌بخش نخواهد بود. هوانگ در گفت‌وگویی با نیویورک‌تایمز در سال ۲۰۲۰ به همین مساله اشاره کرد: «به نظرم بخش هیجان‌انگیز رهبری این است که بدترین کارت‌های بازی را در دست دارید، اما می‌دانید که پیروز خواهید شد».

جن سون می‌داند در مقام رهبر نباید از شکست هراسید و باید از موقعیت، دانش لازم را کسب کرد تا در آینده بتوان از آن استفاده کرد. او توضیح می‌دهد که «وقتی گیم بازی می‌کنید، می‌بازید. آنقدر می‌بازید تا بتوانید ببرید. بازی‌ها اینطورند؛ اما در نهایت می‌برید... و معمولاً جالب‌ترین بخش بازی، موقعی است که دارید می‌بازید».

فرهنگ سازمانی

جن سون معتقد است فرهنگ نتیجه اقدامات و حال‌وهوای رهبران شرکت است، در عین حال ناشی از احساسی است که آدم‌ها در حین کار با این رهبران دارند. در واقع فرهنگ، اقدامات جمعی شرکت و انتخاب‌های رهبران است.

فرهنگ سازمانی اهمیت زیادی دارد، زیرا به نظر او «شخصیت شرکت» است و در واقع این شخصیت نشان می‌دهد که ارزش شرکت چقدر است. به نظر جن سون ارزش‌های بنیادین شرکت و نقش این ارزش‌ها در تصمیم‌گیری‌هاست که به ضرر و سود کارمندان تمام می‌شود. او می‌گوید «نمی‌شود ارزش‌های بنیادین را صرفاً به صورت کتبی نوشت، باید بر اساسشان عمل کرد».

سبک‌وسیاق رهبری: به سوی موفقیت

چه چیز به جن سون هوانگ انگیزه می‌دهد؟ خودش می‌گوید: «کارمندان: ایجاد شرایطی که کارمندان بتوانند کارشان را انجام دهند و در عین حال خود یا چیزی را قربانی نکنند، بسیار مهم و حیاتی است... باید بر عملی که در لحظه انجام می‌دهیم تمرکز کنیم، نه نتیجه و خروجی. خروجی حاصل کاری است که انسان‌های فوق‌العاده انجام می‌دهند». جن سون تاکید دارد «هیچ‌کس رئیس نیست. پروژه رئیس است».

بسیاری از کارمندان او را فردی مدبر و قدرتمند می‌دانند. او از آن دسته مدیرانی است که همیشه در دسترس است، گفت‌وگوها و بحث‌های آزاد را تشویق می‌کند، دفتری ندارد که درش را ببندد و معمولاً همه‌جای شرکت حضور دارد. اگر مشکلی پیش بیاید، شخصاً به تیم کمک می‌کند تا به مسیر برگردد.

سبک رهبری هوانگ را این اصول بنیادین تعریف می‌کند: صداقت فکری و مدارا با خطرپذیری. به نظر او «تحمل در برابر ریسک کردن برابر است با درس گرفتن از اشتباه‌ها». شاید این اصول در وهله اول، کلیشه‌ای به نظر برسند اما وقتی رهبران پیاده‌شان کنند، فراتر از تکرار کلیشه‌ها خواهند بود.

اگر بخواهیم سبک رهبری او را جمع‌بندی کنیم، می‌توان به مواردی که در ادامه می‌آید، اشاره کرد:

۱- درس گرفتن از شکست‌ها: مسیر رهبری یک شرکت، راهی دراز و پرنشیب است. جن سون با شکست‌های بسیاری مواجه بوده، اما از تک‌تک آنها استقبال کرده است.

۲-توسعه سبک رهبری اصیل و واقعی: هیچ‌کس یک‌شبه رهبر نمی‌شود. مشکل این است که خیلی‌ها می‌خواهند از سبک رهبری بقیه تقلید کنند تا به‌نظر رهبری واقعی برسند: از طرز پوشش گرفته تا رفتارها و عادات. جن سون در مقابل به واقعیت تکیه دارد: او افرادی را که خود واقعی‌شان هستند، تحسین می‌کند. به نظر

او برای ارزیابی موقعیت‌ها بایـد صادق بود؛ اگر سبک‌وسیاق رهبری تقلیدی باشد، نمی‌توان بر تیم‌ها و کارمندان تاثیر گذاشت، الهام‌بخش بود و به سوی موفقیت پیش رفت.

۳-تربیـت رهبـران جدیـد: همه رهبـران با محدودیت‌هایـی مواجه‌انـد: رشـد واقعی یک شرکت، نتیجه تربیت نسل جدید رهبران است. حتی کسـب‌وکارهای کوچک هم نباید فقط به بنیان‌گذار خود تکیه کنند.

جن سون معتقد است که تربیت رهبر جدید، از وظایف اصلی رهبر است تا ایده‌های جدید ایجاد شوند و کسب‌وکارهای نو، توسعه یابند. او با توجه به همین مسـاله، با مدیران جوان جلسه برگزار می‌کند تا اسـتراتژی‌ها و چالش‌هـا را به کمک یکدیگر بررسـی کنند و بتواند تجارب خود را با این افراد به اشتراک بگذارد.

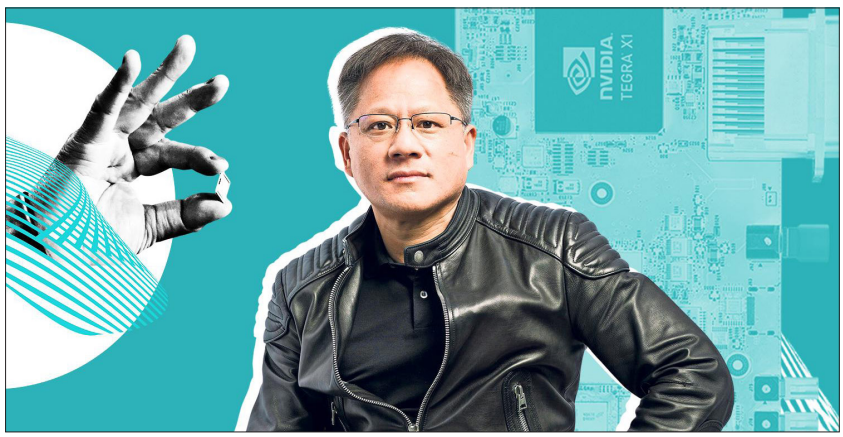
۴- بهترین بـودن: از روی تصادف نیسـت که انویدیا تولیدکننده پیشتاز در حوزه چیپ است. جن سون و دیگر مدیران شرکت انویدیا از همان اول آگاه بودند که انتخاب‌هایشان باید در جهت تبدیل شرکت به سازمانی بزرگ باشد. آنها نسبت به کارشان شوروشـوق دارند و فضای شرکت به آنها این امکان را می‌دهد که بهترین باشند. این فرهنگ باعث می‌شود استعدادهای برتر صنعت، جذب شرکت شوند و بخواهند در چنین محیطی کار کنند.

راهبرد شـرکت در ایـن زمینه، بـر این اصول تمرکز دارد: اول اینکه آنها سعی دارند پروژه‌های معناداری را بیابند که هرگز انجام نشده و تاثیری مثبت بـر جامعـه دارد. به این ترتیـب، قاعده جن‌سـون در ساخت شـرکتی بزرگ این است: کاری را انجام دهیم کـه اهمیت دارد، تخصص تیم است و برای همه، لذت به ارمغان می‌آورد.

۵-روش انجام کار، کلید موفقیت است: بهترین برنامه‌ها نیز ممکن است به دلیل مدیریت اجرایی ناکارآمد یا نادرست اشتباه از آب درآید. جن‌سون می‌داند روش انجام و مدیریت بسـیار مهم‌تر از برنامه‌ای درخشان است.

فلسـفه او در این زمینه این اسـت که پیگیری ایده‌ای سـاده با اجرای فوق‌العاده بهتر از ایده‌ای فوق‌العاده اسـت که پیاده‌سازی‌اش بسیار دشوار باشد.

جن سـون در مقام مدیر انویدیا هرروز شنونده



ایده‌های فوق‌العاده‌ای است اما می‌داند نمی‌تواند تمامی‌شـان را پیاده کند و بایـد تصمیم بگیرد کدام پروژه‌ها را براساس شدنی بودنشان، انتخاب کند. او می‌داند شرکتش نمی‌تواند دنیا را یک‌شبه تغییر دهد و این کار را می‌توان طی سـال‌های سال و با یک محصول واحد سودده انجام داد.

اما و آیاهای استخدام

جن سون در حوزه اسـتخدام خود را محدود به مدارک دانشـگاهی نمی‌کند و ترجیح می‌دهد علاقه متقاضیان را بسنجد. او می‌گوید فردی که می‌گوید کاری را دوسـت دارد و از لغات مرتبط استفاده می‌کند، نسبت به کار هیجان دارد.

شوروهیجان، استفاده از فرصت‌ها، اشتباه کردن و دیدن دنیا از چشمان بچه‌ها کیفیت‌هایی است که در آدم‌ها برای او جذاب اسـت و در استخدام کارمندان نیز به دنبال همین خصیصه‌هاست.

جن سـون در عین حال معتقد است استخدام کار دشواری است. مساله مهم این است که آدم بتواند تحت شرایط دشوار کار کند. تجربه کار او به عنوان پیشخدمت در رستوران در اوایل جوانی همین نکته را تاکید می‌کند: باید بتوان در ساعت شلوغ رستوران کار کرد.

کنترل، هرگز

جن سـون در مصاحبه با پرسش نیویورک‌تایمز درباره سـبک رهبـری‌اش می‌گویـد معمولاً پاسـخ‌های زیادی در چنته ندارد و بیشتر سوال می‌پرسـد. به نظر او امکان ندارد مدیر در مورد همه‌چیز اطلاعات داشـته باشـد؛ فرد در سمت مدیریت فعالیت می‌کند چـون می‌تواند بهتر از بقیه جزییـات و زوایا را ببیند و شـهود بهتری دارد. در واقـع به نظر او مدیـر می‌تواند بهتر از پس پیچیدگی‌ها بربیاید و نگرشی منحصربه‌فرد دارد. او تاکید دارد که با پرسـیدن سـوال‌های درست می‌توان حقیقت ماجرا را دریافت کرد. او می‌گوید امکان ندارد بیشتر از اعضای تیمش به موضوع مسلط باشد: نقش او صرفاً این است که با پرسیدن سوالات به تیم کمک کند و وجوهی را ببینند که قبلاً درک نکرده‌اند.

رهبری از ته دل

جن سـون معتقد اسـت «فرد با مهارت‌هایش مدیریت می‌کند، با قلبش رهبری... اگر خودتان کاری را که انجام می‌دهید دوست نداشته باشید، نمی‌توانید بر دیگران اثر بگذارید تا کار را دوست داشـته باشـند. نمی‌توانید راهی به بزرگ بودن بیابید. به همین دلیل اسـت کـه آدم باید از ته دلش رهبری کند».

جن سون از صمیم دل رهبری می‌کند و به پاس قدردانی از دستاوردهایش در صنعت نیمه‌هادی، موفق به دریافت جایزه Robert N.Noyce سال ۲۰۲۱ شـد. نام او در سال ۲۰۲۱ بین ۱۰۰ فرد تاثیرگذار مجله تایم بود. در سـال ۲۰۰۳ جایزه رهبـری Dr. Morris Chang Exemplary را از انجمن نیمه‌رسـانای Fabless دریافت کرد و در همین سـال نیز جایزه کارآفرین سـال را از آن خود کـرد. او بی‌گمان رهبری بزرگ اسـت که سروصدای چندانی مثل همتایان خود ندارد، اما راهی را ادامه می‌دهد که به نظرش پر از شکست و تجربه‌های تلخ و شـیرین اسـت، درست مثل گیم‌ها.

حال خوب را چگونه در سازمان خود ایجاد کنیم؟

آزاده دودانگه

گفتیم «حال خوب»، امـا حال خوب از نظر ما چه معنایـی دارد؟ اینطور به نظر می‌رسـد که حال خوب تنها به معنای شـادی و احسـاس نشاط قلمداد نمی‌شود و هر شخص یا سازمانی برای حال خوب، تعریفـی دارد. آنچه ما تحت عنوان حال خوب می‌شناسـیم، شـاید خیلی نزدیک به این توضیح باشـد که نشاطِ همراه با انگیزه برای انجام مسـئولیت‌های کاری‌مان در رمیس وجود داشته باشد.

با این نـگاه، ایجاد حال خوب بـرای همکاران دغدغه همیشـگی ما در واحد منابع انسـانی و کل سازمان است و همواره تمام تلاش ما بر این استوار است که چه کنیم تا حال همکاران مان خوب باشد؟ بی‌شک به این موضوع توجه داشته و اقداماتی نیز در این رابطه انجام داده‌ایم اما این مسئولیت مهم، ما را بر آن داشت تا در آبان ماه گذشـته با برخی از فعالین حوزه منابع انسانی گردهم آییم و از دغدغه‌هـا و نگرانی‌هایمان با یکدیگر گفت‌وگو کنیـم و راه حلی بیابیم برای بهبود حال همکارانمان.

دوشـنبه ۲۳ آبان ۱۴۰۱ همراه با تیم «مدرسه اکسـیژن» میزبـان گروهـی از مدیـران و کارشناسان منابع انسـانی شرکت‌های مختلف بودیم تـا ضمن بازگو کـردن تجربیات رمیس در رابطه با موضوع قدردانـی به عنوان یکی از ابزارهای مناسـب در جهت تقویت حال خوب همکارانمان، با سـایر دوسـتان نیز همفکری نموده و از اقدامات و تجربیات انجام شده توسط سایر شرکت‌ها، یاد بگیریم.

این دورهمی کـه با مسـاعدت و همراهی اتاق بازرگانی تهران برگزار شـد، در سه بخش معارفه و آشـنایی مهمانان با یکدیگر، به اشتراک‌گذاری تجربیات رمیس با مهمانان و میز تجربه انجام شد.

به اشتراک‌گذاری تجربیات رمیس با مهمانان

در این گفت‌وگو یکـی از موضوعاتی که به آن پرداختیم، «بررسـی اثر پیگمالیون و قدردانی» بود که توسـط غزل یکی از هـم تیمی‌هایمان ارائه شد.

اثـر پیگمالیون پدیده‌ای اسـت کـه طی آن، انسـان‌ها رفتارشـان را با انتظـارات و توقعات دیگـران تطبیـق می‌دهند تا ایـن تصورات را برآورده کنند. یکی از نـکات مهمی که در این بررسی نیاز به تامل دارد، توجه بر اثر پیگمالیون در راسـتای بهبود عملکرد کارکنان اسـت که برای این کار کافی است به چند نکته زیر توجه کنیم:

۱- از همـکاران انتظـارات درسـت و به‌جایی داشته باشیم.

۲- انتظارات منفی را از عملکرد آنها بزداییم.

۳- به طور پیوسته از تیم‌ها حمایت کنیم.

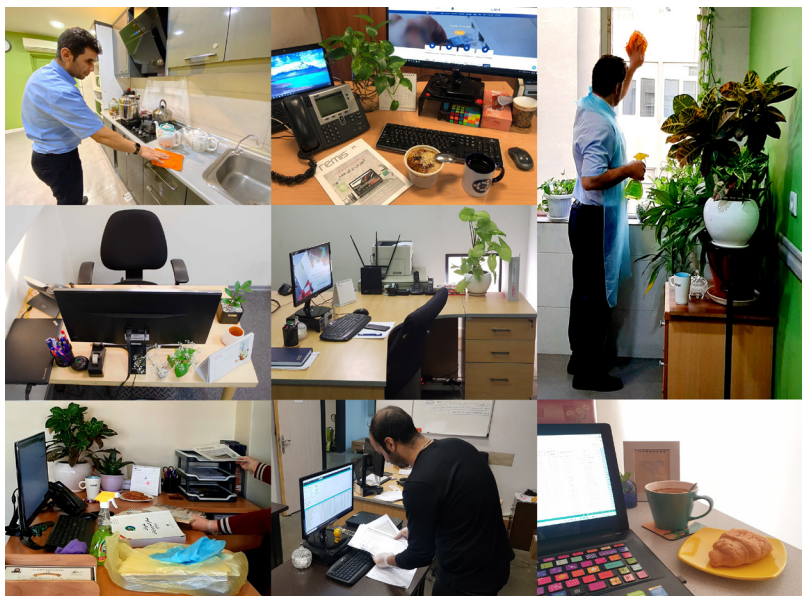
۴- بازخوردهای درست و کارساز ارائه دهیم.

۵- در زمان درسـت و به شـیوه‌ای درسـت قدردانی کنیم.

در چنین شرایطی می‌توانیم امید داشته باشیم که ناخـودآگاه همـکاران براسـاس انتظاراتی درسـت، شـرایط خـود را بهبود بخشـیده و عملکردی متناسب و درسـت با شرایط کار از خود نشان دهند.

پایـان صحبت‌های ما در تیم منابع انسـانی با

لذت کار کردن در یک فضای کاری منظم و تمیز را تجربه کرده‌اید؟



چند سالی است که با نو شدن سال میلادی و آغاز ماه ژانویه، ما نیز همسو با برخی سازمان‌ها رویداد جهانی «تمیز کردن میز کار» را در رمیس برگزار می‌کنیم. اجرای این رویداد در دومین دوشنبه از ماه ژانویه، فرصتی است برای تجربه یک روز شاد و بانشاط همراه با ساماندهی میزها و فضای کاری‌مان در رمیس. چرا که باور داریم داشتن یک فضای کاری تمیز و مرتب علاوه بر افزایش بهره‌وری و کاهش استرس، فوایدی را نیز به همراه دارد از جمله اینکه:

- در ایجاد یک زندگی کاری رضایت‌بخش اثرگذار است.
- به داشتن تمرکز، پردازش اطلاعات و ایجاد ذهنی مرتب همراه با آرامش کمک می‌کند.
- می‌تواند شروع یک نگرش کاملاً جدید برای تمیز کردن، بایگانی کردن و غلبه بر درهم ریختگی باشد.
- زمان بیشتری برای پروژه‌ها و اولویت‌های حیاتی کسب‌وکار صرف و وقت کمتری در بهم ریختگی‌ها تلف می‌شود.

امسال این رویداد همزمان در دفتر مرکزی، سایر دفاتر و در برخی از سایت‌های مشتری که همکارانمان در آنها حضور دارند با مشارکت فعال رمیسی‌ها برگزار شد. در این رویداد رمیسی‌ها علاوه بر تمیز کردن میزکار شخصی خود و چیدمان دوباره آن، ساماندهی فایل‌های ذخیره شده بر روی سیستم‌ها و لپ‌تاپ‌های شخصی، اسکن نمودن و بایگانی اسناد مهم، ساماندهی فایل‌ها و کارتابل‌ها در قفسات و کمدهای شرکت و همچنین ساماندهی کابل‌های مربوط به سیستم‌ها، چاپگرها، تلفن‌ها و... را برعهده گرفتند و برخی از اشیا و کتاب‌هایی که به آنها نیازی نداشتند را به ما تحویل دادند تا بدین طریق با خلوت نمودن فضای کاری خود، فرصت استفاده از آنها را به سایر همکاران خود بدهند. علاوه بر اقدامات انجام شده، یک مسابقه عکاسی نیز با موضوع میزکاریِ تمیزِ من برگزار و به همکارانی که در این رویداد مشارکت و همراهی داشتند، جوایزی اهدا شد.

تاکید بر روی این موضوع بود که «ترویج حال خوب به عنوان مسئولیت اجتماعی شرکت رمیس، با اهمیت است» و انتقال سخاوتمندانه تجربیات در این زمینه، از رسالت های ما در حوزه کارکنان خواهد بود.

اگر شراب خوری جرعه ای فشان بر خاک
از آن گنه که نفعی رسد به غیر چه باک؟
(حافظ)

میز تجربه

در بخش میز تجربه از مهمانان درخواست کردیم تا هر یک در رابطه با چالش‌ها و دغدغه‌هایی که با کارکنان در سازمان خود دارند و اینکه چه اقداماتی در راستای ترویج حال خوب در سازمان انجام می‌دهند گفت‌وگو نمایند. شنیدن نظرات و تجارب هر یک از مهمانان، فرصتی از یادگیری را برای ما فراهم نمود تا از زاویه دید آنها نیز به مسائل بیاندیشیم.

آنچه در ادامه می‌خوانید گزیده‌ای از نظرات و دغدغه‌های مهم مطرح شده در میز تجربه در رابطه با ایجاد حال خوب همکاران در سازمان بود:

۱- در بحران‌های اجتماعی احساسات بد همکاران را به رسمیت بشناسیم و فارغ از نگرش و اعتقاد هر فرد، برخورد همدلانه‌ای با آنها داشته باشیم.

۲- تا انسانی نباشد، کاری هم پیش نمی‌رود. حتی اگر به سود شرکت‌مان هم فکر می‌کنیم، نگاه انسان‌محور و قدرشناسانه از همکاران را در اولویت قرار دهیم.

۳- برقراری تماس‌ها و ارتباطات کاری با بسامدهای کوتاه‌تر، حس دیده‌شدن را در همکاران به‌خصوص برای افراد دورکار تقویت می‌کند.

۴- حس شنیده شدن و دیده‌شدن برای همکاران بسیار ارزشمند است و ما می‌توانیم به روش‌های مختلف به این احساسات پاسخ دهیم.

در انتها نیز از حضور مهمانان عزیزمان و همچنین همکاران اتاق بازرگانی تهران که این فرصت را به ما دادند تا این انتقال تجربه رخ دهد قدردانی نمودیم تا در رویدادهای بعدی پذیرای تجربیات سایر دوستان باشیم.

در بهبود اثربخشی خودم و رمیس مشارکت می‌کنم

یکی از مسائلی که مدیران سازمان‌ها با آن مواجه هستند، رضایت کارکنان است؛ موضوعی که تلاش مدیران را معطوف به خود ساخته است. اما سوالی که وجود دارد این است که «آیا به صرف تامین رضایت کارکنان، آنان خوب کار می‌کنند و بازدهی لازم را دارند؟» برای پاسخ به این سوال، شرکت هی گروپ (معتبرترین شرکت ارائه‌دهنده خدمات مشاوره مدیریت و منابع انسانی) مدلی را تدوین کرده است با نام «اثربخشی کارکنان» که صرفا به بحث رضایت نمی‌پردازد بلکه در پی شناخت و اندازه‌گیری بهتر نگرش‌های کارکنان (احساسات، برداشت‌ها و سطح مطلوبیت) و همچنین بهبود اثربخشی سازمان از طریق تعیین اولویت‌های بهبود است. مدل اثربخشی کارکنان که در حال حاضر یکی از معتبرترین مدل‌های مورد استفاده توسط سازمان‌های بزرگ و بین‌المللی است و دو متغیر اصلی تعلق و تعهد سازمانی (Engagement) و توانمندی کارکنان (Enablement) که هر کدام شامل ۶ متغیر زیرمجموعه هستند و بر این دو متغیر تاثیرگذار هستند را مورد سنجش قرار می‌دهد.

نتایج حاصل از این سنجش، تصویری از وضعیت کارکنان یک سازمان به مدیریت ارائه می‌دهد و به سازمان کمک می‌کند تا نقاط قوت و قابل بهبود خود را در حوزه کارکنان شناسایی کند.

از نکات جالب توجه در این مدل، فراهم شدن امکان مقایسه نتایج حاصل با نتایج سال‌های گذشته خود است. بی‌شک این مقایسه مدیریت سازمان را قادر می‌سازد دریابد در مقایسه با سال‌های گذشته در چه وضعیتی قرار دارد و سپس جهت انطباق، پروژه‌های بهبود را در حوزه‌های مختلف منابع انسانی تدوین نماید.

مطابق با همبستگی موجود میان مولفه‌های مدل، وجود کارکنان اثربخش سبب افزایش در نتایج مرتبط با دستاوردهای مالی، رضایت مشتری و عملکرد مطلوب کارکنان می‌شود.

با این نگاه شرکت رمیس سالانه جهت آشنایی با نگرش‌های کارکنان و شناسایی نقاط قوت و قابل بهبود سازمان که لازمه برنامه‌ریزی جهت بهتر نمودن فضای کاری شرکت و همچنین پاسخ به اعتماد همکاران‌مان است، طرح سنجش نگرش کارکنان مدل هی گروپ را برگزار می‌کند. آنچه در زمان انتخاب این مدل و سپس اجرای آن در رمیس برای ما اهمیت داشت، تمرکز مدل در اندازه‌گیری «نگرش» به جای رضایت بود. چرا که نگرش، نگاه وسیع‌تری نسبت به رضایت دارد و در واقع احساسات و برداشت‌های افراد (خوشایند یا ناخوشایند) را نسبت به یک موضوع، افراد، اشیا یا رویدادها اندازه‌گیری می‌کند اما رضایت (که در دل نگرش نهفته است) تعریفی بسیار کلی و گسترده است و اینطور بیان می‌شود که وقتی احساسی مثبت نسبت به یک موضوع، افراد، اشیا یا رویدادها را تجربه می‌کنیم، رضایت داریم.

در مهر ماهی که گذشت برای ششمین سال متوالی این طرح در رمیس اجرا شد. بدیهی است که نتایج حاصل از تکمیل پرسشنامه‌ها در یک زمان مشخص شده، تنها تصویری از همان لحظه یا لحظات نزدیک به زمان تکمیل پرسشنامه‌ها را نشان می‌دهد و مربوط به تمامی لحظات یک سال کاری نیست. بنابراین ما در واحد منابع انسانی رمیس برای مدیریت بهتر تجربه کارکنان و جمع‌آوری نگرش‌ها و برداشت‌های آنها در طی سال، علاوه بر برگزاری سالانه طرح سنجش نگرش از مسیرهای و ابزارهای مختلف دیگری همچون نظرسنجی‌های نبضی، بازخوردهای موردی، قدردانی‌های سازمانی، جلسات توافق و ارزیابی عملکرد، مصاحبه خروج و... استفاده می‌نماییم تا بتوانیم تجربیات همکاران را در طول سفر کاری خود که در رمیس تجربه می‌کنند بسنجیم و اندازه‌گیری کنیم.

امسال نیز پرسشنامه‌های تکمیل شده، توسط مشاور طرح تحلیل و ارزیابی و طی گزارشی نهایی در سطوح راهبری، مدیران ارشد و همکاران واحدهای سازمانی رمیس ارائه شد. در گام بعدی همانند سال‌های گذشته پس از برگزاری جلسات Focus Group در واحدهای سازمانی و جمع‌آوری بازخوردهای دریافتی، پروژه‌های بهبود تعریف و پس از تصویب توسط هیات مدیره به منزله اهداف و اقدامات واحد منابع انسانی و سایر واحدهای سازمانی در رمیس اجرایی می‌شوند.