


Kubernetes

همون طور که متوجه شدی اگه کل شرکت که شامل کلی تیم هست سیستم های خودش رو بخواد دیپلوی کنه باید کلی سرور بخریم و هر تیم روی همون سرور سرویس های خودش رو بالا بیاره. این کار چندتا مشکل داره. مهم ترینش اینکه خیلی گرون میشه و از کل سیستم خریداری شده لزوماً به صورت بهینه استفاده نمی شه. دومین مشکل بزرگش اینکه همه توسعه دهنده ها نیاز دارن که دغدغه های مشترکی رو چند بار حل کنن و نیاز به دانش خیلی زیادی برای حل این مسائل مربوط به محیط پروداکشن (مثل شبکه، پایداری، امنیت و ...) دارن. برای حل یه جا و مشترک این مشکل زیرساخت جدیدی ایجاد شد که کانتینرهای اپلیکیشن ها رو می گیره و خودش مدیریت می کنه که با توجه به خصوصیات اون کانتینر روی چه سیستمی بالا بیاد. این یه جا جمع شدن مسئولیت دیپلوی و نگهداری درست محیط سیستمی پروداکشن باعث افزایش بهره وری سازمانی و حتی فنی می شه. اینجوری تیم ها می تونن مستقل از اینکه متوجه بشن برنامه شون کجا و چجوری اجرا می شه به راحتی اون رو دیپلوی کنن. مدیریت کانتینرها و کارهای جانبی اون (مثل کانفیگ شبکه مورد نیاز و پایش منابع و ...) رو همون ابزار زیرساختی که اصطلاحاً **orchestration** رو انجام می ده بر عهده داره. یعنی خلاصه اینکه روی یه سری سرور ما این ابزار رو می ریزیم و از اون ور میگیریم این کانتینرها با این تنظیمات اجرا بشن و بقیه کارها رو به بهینه ترین روش خودش انجام می ده. یکی از بهترین های این ابزارها، کوبرنتیزه.

مطالعه بیشتر

واقعا چرا از این زیرساخت ها خویه استفاده کنیم؟ لینک های زیر توضیح جامعی رو می ده:

 [PaaS Explained](#)

[What is Kubernetes? | Kubernetes](#)

[What is Platform as a Service \(PaaS\) & Why Use It? - Salesforce EMEA](#)


حالا کوبرنتیز چی چی هست؟

Kubernetes یک ابزار متن باز سامان دهی Container است که بسیاری از کارهای لازم برای پیاده سازی، مدیریت و مقیاس دهی برنامه های مبتنی بر Container را به شکل خودکار انجام می ده. به بیان دیگر، با کلاستر کردن ماشین های اجراکننده Container و استفاده از Kubernetes می شه این کلاسترها را به سادگی و با بهره وری بالا مدیریت کرد. Kubernetes ماشین های حقیقی و مجازی در دسترسش را به شکل کلاستر در یک شبکه ی یکسان کنار یکدیگر جمع می کنه. کلاستر در واقع بستری است که تمامی اجزا، قابلیت ها و بار کاری کوبرنتیز در آن کانفیگ می شه. از آنجا که این ماشین ها می تونن از طریق سرویس های ابری خصوصی و عمومی در کنار زیرساخت های دیگر تامین شوند، Kubernetes ابزاری مناسب برای میزبانی از برنامه های ابری است که نیاز به سرعت بالا در تغییر مقیاس و انتقال داده دارند. این فناوری برای اولین بار توسط مهندسان گوگل توسعه داده شده است.

 [Kubernetes in 5 mins](#)

الان توی شرکت ما، تیم زیرساخت به صورت مستقیم مسئول مدیریت و نگهداری از کوبرنتیزی هست که تیم ها استفاده می کنن.

این کورس که حدود ۴ ساعت از وقتت رو می گیره تقریباً همه چیز رو به زبان ساده توضیح می ده:

 [Kubernetes Tutorial for Beginners \[FULL COURSE in 4 Hours\]](#)

پیشنهاد می‌کنم که تمامی کامندهایی که اجرا می‌شود رو هم لوکال تست کنی و انجام بدی. برای تمرین‌های بعدی به محیطی که minikube بالا اومده باشه نیاز داری. اگه با minikube به مشکل خوردید میتونید از microk8s استفاده کنید فقط موقع استفاده اش shecan.ir رو هم فعال کنید.

تقریباً در تمامی مراحل راه‌اندازی به vpn نیاز داری. پس قبل از هرکاری اجراش کن.

قسمت مربوط به Helm و Ingress رو می‌تونی بپری.

مطالعه بیشتر

مفاهیم مربوط به کوبرنتیز رو هم اگه بدونی خیلی خوب می‌شه. بخش‌های overview تا configuration رو به نگاهی بنداز؛

[Concepts | Kubernetes](#)

همینطور این ویدئو در مورد معماری داخلی کوبرنتیز و اینکه چجوری کار می‌کنه اطلاعات خوبی می‌ده:

 [Kubernetes architecture explained](#)

مطالعه خیلی بیشتر

همینطور توی این رفرنس مربوط به خود کوبرنتیز با جزئیات مطرح شدن:

[Cluster Architecture | Kubernetes](#)

تمرین

توی این تمرین می‌خوایم سرویس nginx-echo-header رو دیپلوی کنیم. برای این کار مراحل زیر رو به ترتیب انجام بده:

1. یه فایل deployment.yaml بساز و توش مانیفست یه دیپلویمنت رو بنویس که از این [ایمپج](#) استفاده کنه.

a. برای شروع خوبه که حداقل ۳ تا رپلیکا ازش داشته باشیم.

b. پورت ۸۰۸۰ رو هم باز کن.

2. دیپلویمنتی که نوشتی رو روی کلاستر می‌نی کیوب دیپلوی کن.

مطالعه بیشتر

خوبه که در مورد تفاوت ۲ تا دستور apply و create، اینجا رو بخونی:

[kubectl apply vs kubectl create? - Stack Overflow](#)

3. لیست پادها رو بگیر و مطمئن شو همه پادها running باشن.

4. سعی کن به پادهایی که ساختی پورت-فوروارد کنی. ([منبع](#))

بعد از پورت-فوروارد میتونی روی یه ترمینال دیگه با curl localhost:8080 به پاد درخواست بدی و هدرهای ریکوئستت رو ببینی.

5. یه فایل service.yaml بساز و توش مانیفست یه سرویس رو بنویس که بیاد پورت ۸۰۸۰ از کانتینر رو به پورت ۸۰ سرویس مپ کنه.

6. سرویسی که نوشتی رو روی مینی کیوب دیپلوی کن و چک کن که سرویس بالا باشه.

7. این بار به جای پاد، به سرویسی که نوشتی درخواست ارسال کن. (برای این کار نیاز داری که از توی پاد درخواست رو ارسال کنی. مثلاً با دستور `kubectl exec` و `curl` به shell بزن).

مطالعه بیشتر: Ingress

از همون ویدئوی قبلی در مورد ingress ببین و سعی کن باهاش پیش بری:

 [Kubernetes Tutorial for Beginners \[FULL COURSE in 4 Hours\]](#)

همچنین می‌تونی از این رفرنس استفاده کنی:

[Ingress | Kubernetes](#)

تمرین:

برای اینکه بتونیم از بیرون کلاستر به داخل کلاستر ارتباط برقرار کنیم، از اینگرس استفاده می‌کنیم. تو یه فایل `ingress.yaml`، یه مانیفست اینگرس بساز که از سرویسی که تو تمرین قبلی نوشتی به عنوان بکند استفاده کنه و دیپلویش کن و بدون پورت-فوروارد کردن سعی کن به سرویسی که نوشتی ریکوئست بزنی.

مطالعه‌ی خیلی بیشتر: Security

[این جیت‌شیت](#) با اینکه برای داکر نوشته شده ولی چون محیطی که کانتینر بالا میاد تقریباً همیشه کوبرنتیز هست، از طریق ریسورس‌ها و کانفیگ‌های کوبر می‌تونیم امنیت کانتینرها رو تامین کنیم. میتونی بهش یه نگاهی بندازی.

حالا که این بخش رو تموم کردی ممنون می‌شم که فرم [یازخورد](#) رو پر کنی. 