|  |  |  |
| --- | --- | --- |
| دانشگاه صنعتي اميركبير **(پلي تكنيك تهران)** | **باسمه تعالي**  فرم تعريف پروژه  فارغ التحصيلي دوره كارشناسي | **دانشكده مهندسي كامپيوتر و فناوري اطلاعات** |

**تاريخ: ...........................**

**شماره: ..........................**

|  |
| --- |
| **عنوان پروژه: ارکستریشن خودکار برای دسترس‌پذیری بالا در محیط محاسبات ابری با استفاده از ابزارهای متن باز** |
| **استاد راهنماي پروژه: دکتر بخشی امضاء:** |
| **مشخصات دانشجو:**  **نام و نام خانوادگي: محمد پویا خرسندی گرايش: فناوری اطلاعات**  **شماره دانشجوئي: 9331910 ترم ثبت نام پروژه: دوم 97-98** |
| **داوران پروژه:**  **1- امضاء داور:**  **2- امضاء داور:** |
| **شرح پروژه** (در صورت مشترك بودن بخشي از كار كه بعهده دانشجو مي باشد مشخص شود)**:**  ارائه سرویس های سازمانی عموما بر پایه اپلیکیشن های مبتنی بر وب انجام می‌گیرد. در دسترس بودن سرویس‌های وب سازمانی از اهداف مهم برنامه‌ریزی زیرساخت‌های IT سازمان محسوب می گردد تا بتواند با کیفیت مناسب حتی در زمان پیک بار کاری به کاربران پاسخگویی مناسب انجام دهد. کاهش کیفیت خدمت با افزایش بار و یا از دسترس خارج شدن آن صدمات مالی و اعتباری زیادی می تواند به بار بیاورد. سرویس های ابری امروزه برای دریافت امکانات محاسباتی بر حسب نیاز استفاده می شود. مدیریت تعداد سرورهای مجازی مورد نیاز با توجه به بار کاری وب سایت و ارکستریشن خودکار آنها روشی برای افزایش دسترس پذیری و کاهش هزینه مدیریت و نگهداری سرویس است. در این پروژه یک وب سایت پایه برای ارائه محتوی چندرسانه ای راه‌اندازی و سپس آن را بر روی یک محیط ابری بارگذاری کرده و با استفاده از ابزار‌های تولید بار روی این وب‌سایت، دسترس پذیری بالای آن را تست و ارزیابی میکنیم. سپس محیط ارکستریشن را راه‌اندازی کرده و امکان مقیاس شدن خودکار و دسترس پذیری بالا‌‌ی آن را فراهم می کنیم. |
| **وسائل مورد نياز:** |
| **محل انجام پروژه: دانشگاه صنعتی امیرکبیر تاريخ شروع: اسفند 97** |

اين قســمت توســط دانشــكده تكميــل ميگــردد:

|  |
| --- |
| **تاريخ تصويب در گروه: اسم و امضاء:**  **تاريخ تصويب در دانشكده: اسم و امضاء:**  **اصلاحات لازم در تعريف پروژه:** |

**توجه**: پروژه حداكثر يك ماه و نيم پس از شروع ترمي كه در آن در درس پروژه ثبت نام بعمل آمده است بايد به تصويب برسد.

|  |  |  |
| --- | --- | --- |
| نسخه1- دانشكده | نسخه 2- استاد راهنما | نسخه 3- دانشجو |

# **ضمیمه فنی پروژه** **ارکستریشن خودکار برای دسترس‌پذیری بالا در محیط محاسبات ابری با استفاده از ابزارهای متن باز**

# **1-مقدمه**

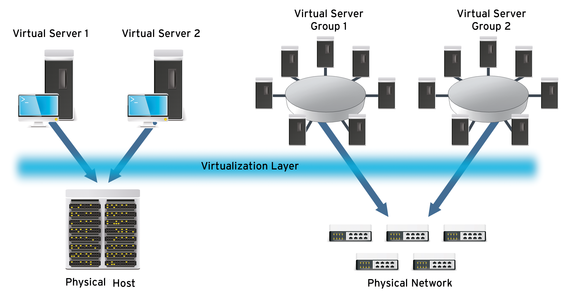
در محیط محاسبات ابری مقیاس بزرگ، مراکز داده ابر و کاربران نهایی از نظر جغرافیایی در سراسر جهان توزیع شده‌اند. بزرگترین چالش برای مراکز داده ابر این است که چگونه میلیون ها درخواست را که بطور مداوم از کاربران نهایی می رسند، بطور صحیح و موثر رسیدگی و سرویس داده شوند. در محاسبات ابری تعادل بار برای توزیع مساوی بار کاری پویا بین همه گره‌ها است. تعادل بار برای رسیدن به بیشترین سطح رضایت کاربر و افزایش نرخ استفاده از منابع با تضمین تخصیص کارا و عادلانه هر منبع محاسباتی کمک می کند. تعادل بار مناسب در کم کردن مصرف منابع، پیاده سازی عدم خرابی، توانایی مقیاس پذیری، اجتناب از گلوگاه‌ها و ارایه بیش از حد و غیره کمک میکند.

# **2- مدل محاسبات ابری**

**2-1- مجازی سازی سرورها وشبکه**

#### مجازی‌سازی سرور (Server Virtualization) تکنیکی برای تقسیم‌بندی یک سرور فیزیکی به چندین سرور مجازی کوچکتر، به کمک یک نرم افزار مجازی سازی می‌باشد.در مجازی سازی سرور، هریک از سرورها چندین سیستم عامل را در یک زمان واحد اجرا می‌کنند .امروزه مراکز داده تجاری متشکل از چندین سرور می‌باشند. تعدادی از این سرورها مدت زمان زیادی از روز بیکار هستند چرا که بار کاری بر روی تنها تعدادی از  سرورهای توزیع شده است. این امر سبب اتلاف منابع سخت افزاری گران قیمت، مصرف برق، نگهداری و سیستم سرمایشی می‌شود. مجازی سازی سرور تلاش می‌کند تا میزان بهره وری از منابع را با استفاده از تقسیم‌بندی سرورهای فیزیکی به چندین سرور مجازی، که بر روی هریک از آن‌ها سیستم عامل و اپلیکیشن‌ها مختص به خودش نصب شده است، افزایش دهد. مجازی سازی سرور می‌تواند هر سرور مجازی را وارد به رفتاری مانند یک سرور فیزیکی کند، که این امر موجب افزایش ظرفیت هر ماشین فیزیکی می‌شود.

#### مجازی سازی شبکه را در واقع روشی جهت جداسازی منابع شبکه و منابع محاسباتی همین طور واگذاری آن به یک شبکه مجازی دانست که این کار جهت یکسان کردن چندین شبکه مجازی قابل برنامه ریزی و مستقل صورت می گیرد. در این حالت از مجازی سازی شبکه تمامی سخت افزارها و همچنین نرم افزارهای درون یک شبکه مجازی شکل می‌گیرد. همان طور که در شکل (1) نشان داده شده است نوع دیگری از مجازی سازی شبکه به این شکل کار می‌کنند بخش‌های مختلف مجزا شده شبکه به وسیله یک بخش فیزیکی معلوم به اشتراک گذاشته می‌شود. مجازی سازی شبکه را می توان به ایجاد شبکه‌های مجازی داخل یک سرور عنوان کرد که می‌تواند باعث ایجاد ارتباط سیستم‌های مجازی با یکدیگر شود.

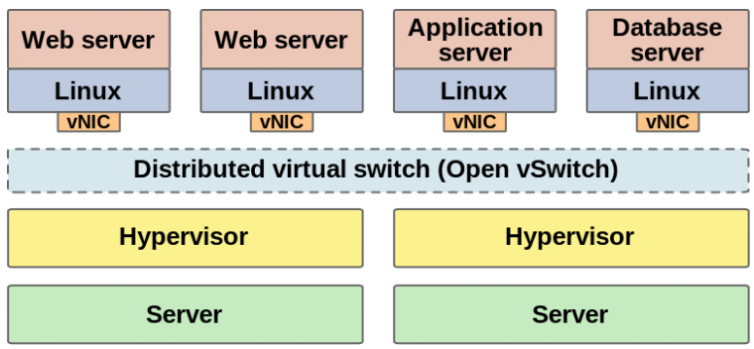


شکل 1- مجازی سازی سرورها و شبکه

**2-2-محیط‌های متن باز برای پیاده‌سازی سرویس‌های محاسبات ابری**

**سوییچ OVS :**

سوئیچ Open vSwitch ابتدا در قالب پروژه Xen پیاده سازی شد و بعدا بطور فراگیر برای پیاده سازی شبکه مجازی بین ماشین های مجازی به کار رفته است. Open vSwitch یا به اختصار OVS یک سوئیچ مجازی چندلایه‌ای است که بسیاری از پروتکل‌ها و واسط‌های شبکه در آن پیاده‌سازی شده است. این سوئیچ که به طور نرم‌افزاری پیاده شده است، یک سوئیچ متن‌باز تحت حمایتApache License 2.0   است. این سوییچ پورت‌های مجازی شبکه در اختیار ماشین‌های مجازی قرار می دهد و امکان شبکه سازی بین ماشین های مجازی بصورت توزیع شده را ایجاد می کند و سیاست های مسیریابی را می توان در آن تعریف نمود و به مانند یک روتر قابل برنامه ریزی می تواند عمل کند.



شکل 2- جایگاه OVS در پشته محاسبات ابری

# **2-3- مقیاس پذیرکردن سرویس‌ها برای دسترس پذیری بالا در محیط ابر**

در محیط محاسبات ابری، مقیاس پذیری به صورت افقی بوده به این معنا که افزودن و کاهش ماشین‌های مجازی بنا به بار کاری آن ها می‌تواند انجام شود. به طور مثال فرض کنید یک سرویس شبکه­ای بر روی 2 ماشین مجازی در حال پاسخ به درخواست کاربران است، اگر در این سناریو تعداد درخواست‌های وارد شده به این سرویس بیشتر از حد آستانه‌ای که قبلاً تعریف شده است، شود یک ماشین مجازی به این مجموعه اضافه و تعداد ماشین‌های در حال سرویس دهی را به 3 عدد می‌رساند. به طور مشابه هنگامی که بار کاری به حالت قبل برگردد و در یک بازه زمانی مشخص از حد آستانه کمتر بماند ماشین ساخته شده بعد از این مدت مشخص پاک و پیکربندی محیط بروزرسانی می‌شود.

توزیع بار یک فرایند مهم در زمینه پردازش بار کاري در سیستم‌هاي توزیع شده بر روي منابع محاسباتی است. هدف توزیع بار بین گرههاي محاسباتی موجود، به دست آوردن بهره وري بالا و کاهش هزینه‌ها نسبت به مدلهاي سنتی میباشد. توزیع بار از محدودیتهاي سیستمی که به واسطه‌هاي بارهاي نامتوازن ایجاد میشود جلوگیري میکند. همچنین توزیع بار مکانیزمی را فراهم میکند که به کمک آن نمونه‌هاي جدید از برنامه هاي کاربردي میتواند به صورت پویا و بدون تغییر در پیکربندي شبکه اضافه شود. این عملیات با توجه به ویژگیهاي منحصر به فرد هر گره مانند قدرت پردازشی آن، حافظه و ویژگیهاي ارتباطی هر گره میتواند صورت پذیرد. جدول (1) فهرستی از مزایا و چالشهاي موجود در فرایند توزیع بار را نشان می‌دهد.

جدول 1- توزیع بار بین گره های محاسباتی: مزایا و چالش ها

|  |  |
| --- | --- |
| مزایا | چالش‌های پیش رو |
| * تقویت عملکرد کلی سیستم * استفاده بهینه از منابع انرژی و مدیریت مصرف * خدمات کیفیت سرویس * اجتناب از ازدحام | * مهاجرت ماشین‌های مجازی * پیچیدگی متوازن کننده‌های بار * پراکندگی جغرافیایی گره‌های توزیع شده در ابر * پایداری متوازن کننده‌های بار و جلوگیری از ایجاد گلوگاه |

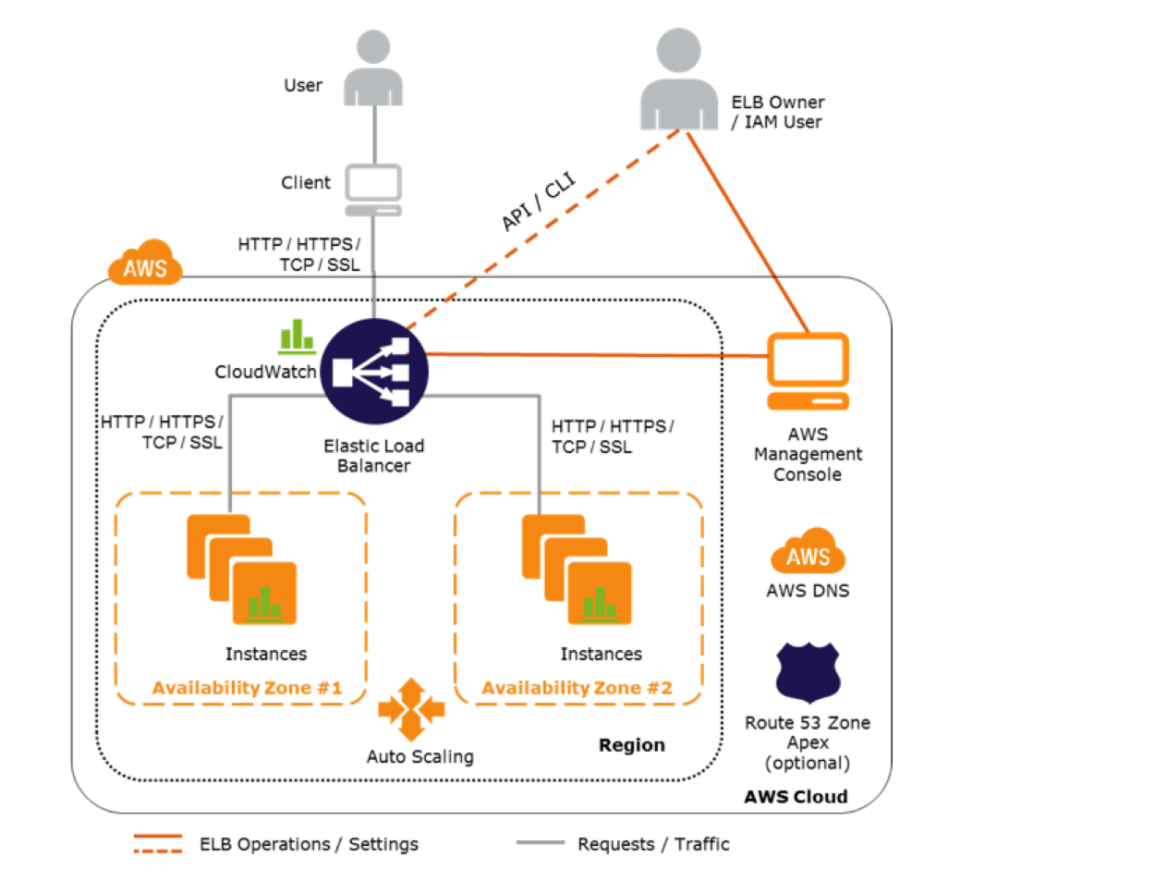
# **3- ارکستریشن سرویس‌های ابری: محیط‌های تجاری و متن باز**

**3-1- ارکستریشن و زمان بندی**

هنگامی که اپلیکیشن‌ها روی چند سیستم میزبانی مقیاس می‌یابند، توانایی مدیریت هر سیستمِ میزبان و کنترل چرخه عمر هر یک از نمونه‌ها از زمان ایجادشدن (onboarding) تا زمان انقضا آن از اهمیت فراوانی برخوردار است. هماهنگی (Orchestration) اصطلاحی کلی است که به زمان‌بندی کانتینرها (Containers) و ماشین های مجازی و مدیریت خوشه‌های محاسباتی شامل نمونه‌های وابسته به هم و نیز به ایجاد کردن یا حذف میزبان‌ها اشاره دارد. در این محیط، منظور از زمان‌بندی (scheduling) توانایی یک مدیر برای بارگذاری یک فایل سرویس روی یک سیستم میزبان و تعیین شیوه اجرای کانتینر مربوط به آن سرویس است. با این که زمان‌بندی به عمل خاص بارگذاری سرویس اشاره دارد؛ اما در یک معنای کلی‌تر زمانبندی مسئول مدیریت میزبان‌ها و سرویس‌ها در ظرفیتی که لازم باشد می‌باشد.

**3-2- ارکستریشن در سرویس ابر شرکت آمازون AWS**

سرویس ابر شرکت آمازون تحت عنوان Amazon Web Service (AWS) بزرگترین سرویس ابر تجاری در دنیاست. در محیط AWS، سرویس EC2، استفاده و مدیریت سرورهای اختصاصی مجازی که سیستم عامل­های لینوکس یا ویندوز را بر روی Xen Hypervisor اجرا می­کنند، میسر کرده است. نمونه­ های ماشین با توان­های پردازشی مختلف موجود می­باشد و بر اساس محاسبات/ساعت اجاره می­شوند. برنامه­های مستقر بر روی این ماشین­ها بسیار توسعه پذیر و با تحمل پذیری بالای خطا می­باشند. ذکر تفاوت میان یک نمونه ماشین و یک تصویر ماشین می­تواند به درک مفاهیم موجود در سرویس آمازون کمک کند. به طور کلی نمونه ماشین در واقع تقلید یا همسان­سازی سکوی سخت­افزاری مانند x86 و غیره بر روی لایه نرم­افزار مجازی Xen می­باشد. در حالی که تصویر ماشین، نرم افزار و سیستم عاملی است که در سطح یک نمونه ماشین اجرا می­شود و می­توان به محتویات یک درایو راه‌­انداز تشبیه نمود.



شکل 3- ارکستریشن در محیط سرویس ابر آمازون (AWS)

همان طور که در شکل بالا مشخص است ما با استفاده از سیستم ارکستریشن آمازون میتوانیم وضعیت شبکه و بار شبکه در هر قسمت از دیتا سنتر خود را مشاهده کنیم و بر حسب نیاز ماشین‌های مجازی خود را افزایش داده یا کم میکنیم و با دادن تنظیمات لازم به سیستم مقیاس پذیری سرویس های ابری مد نظر صورت میگیرد.

برای سهولت برنامه ریزی سرویس های ابری در آمازون، محیط تجاری CloudFormation توسعه یافته است.

CloudFormation برای ایجاد مجموعه‌ای از ماشینهای مجازی آماده‌ی استفاده به همراه نرم‌افزارهای از پیش نصب شده) به عنوان مثال برنامه های کاربردی وب( استفاده میشود. تعریف سرویس با استفاده از یک زبان اسکریپتی مانند Yaml یا JSON انجام می گیرد و از طریق یک سوکت برنامه نویسی به محیط AWS برای پیکربندی ماشین های لازم منتقل می شود.

**3-3- محیط Cloudify**

Cloudify یک پلتفرم برای تنظیم و هماهنگ سازی مولفه­های مختلف یک سرویس شبکه در بستر ابر و با تکیه بر زبان TOSCA است که از معماری Model-Driven بهره می­گیرد.

Cloudify یک معماری توسعه پذیر و سازگار با ارائه‌دهندگان  IaaS/PaaSمختلف همچون AWZ، Openstack و Microsoft Azure دارد. این پروژه متن باز بوده و در repositoryهای عمومی نگه­داری می­شود و قابل دسترسی است و توسعه دهنده اصلی آن Gigaspaces است.

Cloudify از مولفه­های مختلفی تشکیل شده است که در شکل زیر نشان داده شده است:

* manager cloud همان orchestrator است که توسعه و مدیریت چرخه حیات برنامه­ها را برعهده دارد.
* cloudify agent ها نیز برای اجرای چرخه­های کاری در ماشین­های­ مجازی استفاده می­شوند.
* Elastic search به عنوان یک پایگاه داده برای برای ذخیره اطلاعات وضعیت توسعه (deployment state) شامل داده­های اجرایی (runtime) و لاگ­های رسیده از کامپوننت­های دیگر است.
* Logstash برای پردازش لاگ­های رسیده از کامپوننت­ها و agentها استفاده می­شود.
* Reimann به عنوان یک policy engine برای پردازش تصمیمات اجرایی در مورد availability ، SLA و مانیتورینگ استفاده می­شود.
* RabbitMQ به عنوان یک async transport برای ارتباطات میان کامپوننت­ها شامل remote agent ها استفاده می­شود.

عملکردهای مورد نیاز برای orchestration، توسط پروژه ARIA TOSCA و با تعریف TOSCA-based blueprint  و deployment workflow engine انجام می­شود. کامپوننت­ها و پلاگین­های مربوط به خود cloudify با زبان python توسعه داده شده است.

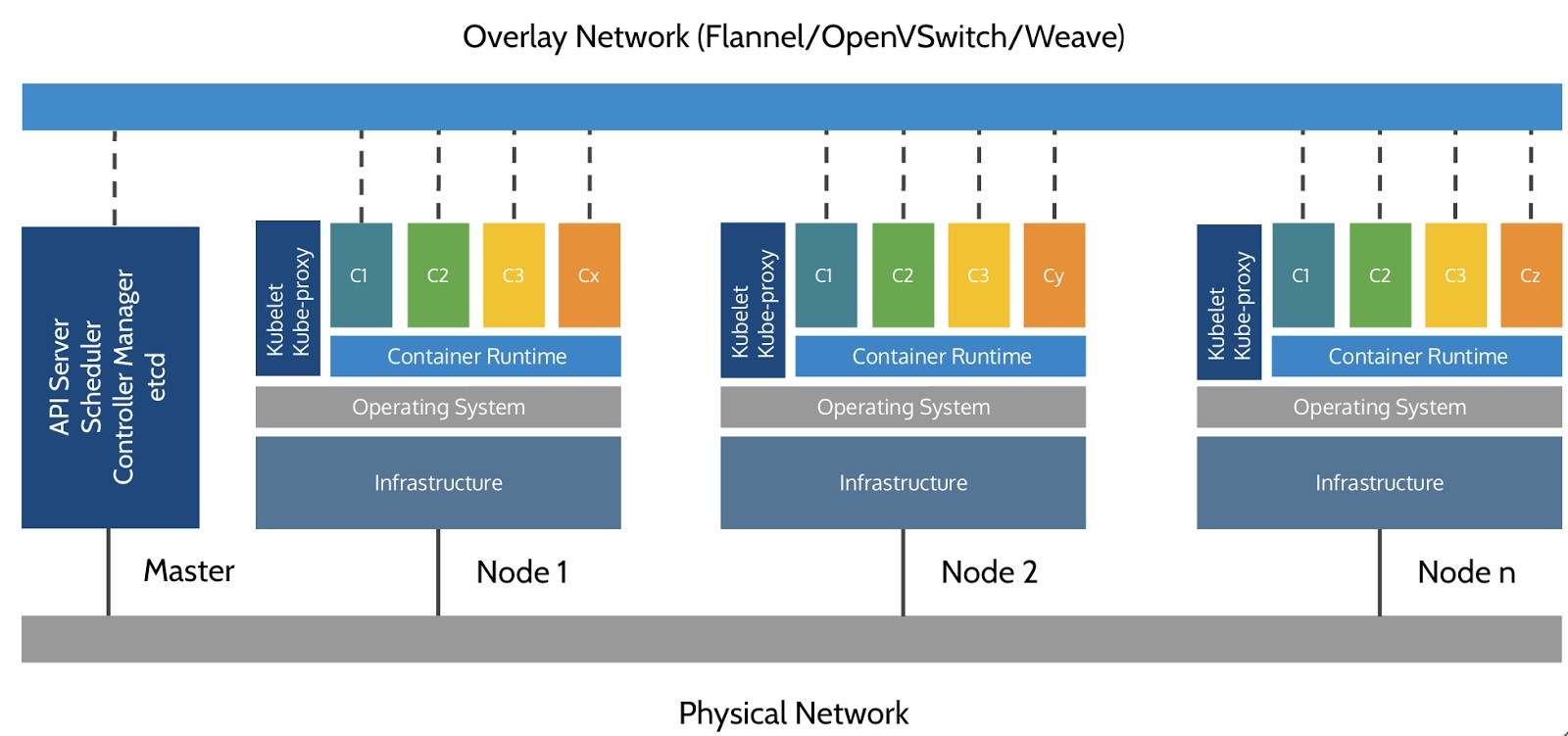
**3-4- محیط Kubernetes**

Kubernetes پروژه دیگری است که به نوعی می­تواند جایگزین OpenStack باشد و در واقع یک container orchestration tool است. این پروژه برای ارائه برنامه­های مبتنی بر تکنولوژی کانتینر مناسب است و هدف آن مدیریت کلاسترهایی از کانتینرها است. همچنان که داکر چرخه حیات کانتینر را مدیریت می­کند، Kubernetes توسعه و جای دهی کانتینرها در ماشین­های مجازی موجود یا جدید و هماهنگی میان آن­ها را مدیریت می­کند.

Kubernetes ارتباط خود را با ماشین های مجازی یا ماشین های فیزیکی از یک شبکه ی اشتراکی مانند"flannel " که در هر ماشین وجود دارد استفاده می کند. در شبکه ی کلاستری Kubernetes تمامی اجزا، قابلیت‌ها و Workload پیکربندی می شوند. هر ماشین در اکوسیستم Kubernetes یک وظیفه و نقش را بر عهده دارد. یک ماشین (سرور) در کلاستر به انتخاب ما باید به عنوان Master در نظر گرفته شود. این سرور با ایجاد یک IP برای مدیر سیستم راه ورود به کلاستر و مدیریت آن را باز می کند، علاوه بر این تست سلامت و پایداری دیگر سرور ها و تعیین نقش و همچنین ارکستراسیون بین اجزای مختلف Kubernetes را به عهده دارد. در واقع سرور Master نقطه اصلی ارتباط با کلاستر است و مدیریت و نظارت متمرکز در سراسر سیستم Kubernetes را فراهم می‌کند.

دیگر ماشین هایی که در کلاستر Kubernetes ایجاد می شوند سرور Nodeنام دارند. این سرور ها وظیفه ی اجرا کردن بار کاری با استفاده از منابع داخلی یا خارجی را دارند. برای کمک به ایزوله سازی، مدیریت و انعطاف پذیری Kubernetes برای اجرای اپلیکیشن‌ها و سرویس‌ها از Container استفاده می‌شود، بنابراین هر ماشین Node در کلاستر نیاز به دارا بودن به یک سیستم و نرم افزار کانتینری مانند Docker است . rkt Node در کلاستر دستور العمل‌ها را از سرور Master دریافت میکند و بر اساس تنظیمات، کانتینر‌ها را ایجاد یا آن‌ها را حذف میکند و همچنین تنظیمات مناسب شبکه‌ی کلاستر و مسیریابی ترافیک را تنظیم می‌کند.

همانطور که اشاره شد اپلیکیشن‌ها و سرویس‌های ما در کلاستر Kubernetes با کاینتر اجرا می‌شوند. کاربر جهت تعامل با کلاستر از IP سرور اصلی یا کلاینت و یا کتابخانه استفاده می‌کند. برای اجرای اپلیکیشن‌ها یا سرویس‌ها یک برنامه اعلامیه در فایل‌های JSON یا Yaml ارایه می شود که مشخص کننده این است که چه چیزی باید ایجاد شود و چگونه باید آن را مدیریت کرد. سرور Master فایل برنامه را می گیرد و چگونگی اجرای آن در زیرساخت را با بررسی شرایط و وضعیت فعلی سیستم مشخص می کند. این گروه از برنامه های تعریف شده توسط کاربر که بر اساس یک نقشه خاص اجرا می شوند نتیجه‌ی نهایی Kubernetes را نشان می‌دهند.



شکل 4- مدل مفهومی پیاده سازی محیط Kubernetes

**اجزا نرم افزاری Kubernetes:**

**etcd**

یک جزء مهم Kubernetes که جهت ذخیره‌سازی و نگهداری فایل های پیکربندی کلاستر است. پروژه ی etcd توسط تیم Core Os توسعه داده شده است، که بسیار سبک است و یک مکانی جهت نگهداری Key-Value ها می باشد که می تواند در سراسر کلاستر و Node ها توزیع شود. Kubernetes از etcd برای نگهداری فایل ها و داده‌های پیکربندی کلاستر استفاده می‌کند که قابل دسترسی توسط هر Node در کلاستر می باشد. این می تواند جهت جست و جوی سرویس‌ها ، پیکربندی یا پیکربندی مجدد خودشان جهت بروز نگه داشتن اطلاعات استفاده شود.

**Kube-apiserver**

یکی دیگر از سرویس‌های مهم در سرور Master، api-serverاست. این نقطه‌ی اصلی مدیریت و ورود به Kubernetes است که به کاربر اجازه می‌دهد Kubernetes را پیکربندی کند. در واقع kube-apiserver پلی بین اجزای مختلف با هدف نگهداری و وحذف سلامت کلاستر و انتشار اطلاعات و اجرای دستور عمل‌ها است. api server یک رابط RESTfull ایجاد می‌کند که به این معنی است، بسیاری از ابزار‌ها و کتابخانه‌ها به راحتی می‌توانند با آن ارتباط برقرار کنند. راه ارتباطی کاربر از طریق ابزار Kubectl مهیا می شود که به صورت پیش فرض در سیستم وجود دارد.

**Kube-Controller-Manager**

Control Manager سرویسی است که وظایف زیادی دارد، از جمله کنترل کننده های وضعیت کلاستر، چرخه‌ی حیات کلاستر و همچنین انجام وظایف دیگر. به عنوان مثال یک Replication-Controller تضمین می‌کند که تعداد کپی‌ها (نسخه های یکسان) درست تعریف شده، یا تعداد مورد نظر Pod ها در کلاستر یکی باشند و به درستی در کلاستر قرار گرفته باشد. جزییات این عملیات در etcd نوشته می‌شود جایی که Controller Manager برای تغییرات به آن نگاه می‌کند. زمانی که تغییرات در کلاستر ایجاد می‌شوند،Controller اطلاعات جدید را می‌خواند و دستوری را اجرا می‌کند.

**Kube-Scheduler**

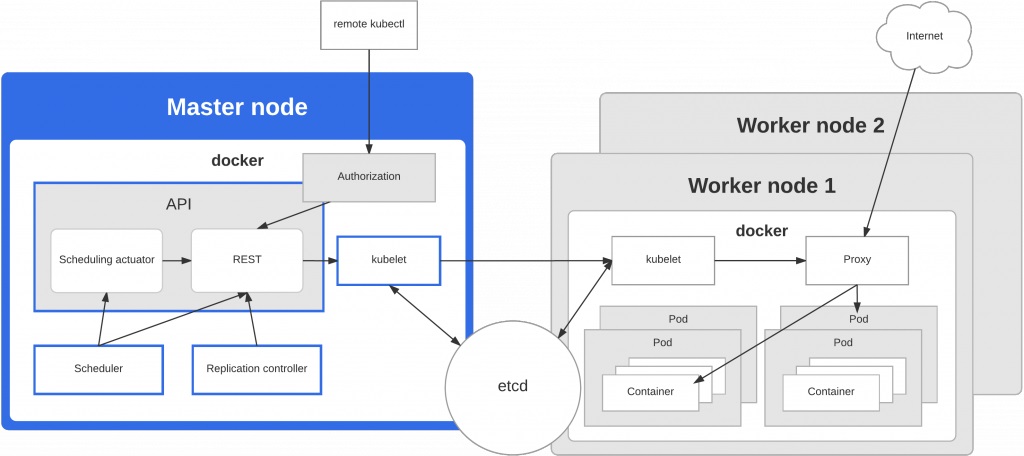
Kube-Scheduler فرایندی است که در واقع مسولیت و وظیفه را به Node های خاص در کلاستر اختصاص می‌دهد. این سرویس شرایط عملیاتی را می‌خواند همچنین تجزیه و تحلیل محیط زیرساخت فعلی و جایگزین می کند و وظیفه به Node قابل قبول داده می شود. Scheduler مسئول بررسی ظرفیت موجود در هر Node است تا مطمئن شود که حجم کاری اختصاص داده شده بیش از منابع موجود نباشد Schedular . باید ظرفیت کل و همچنین منابعی که قبلا در هر سرور اختصاص داده شده است را بشناسد..

**Kubelet**

نقطه تماس اصلی برای هر Node با کلاستر سرویس Kubeletاست این سرویس مسئول انتقال اطلاعات(دستور) به سرویس های کنترل کلاستر هست و از طریق آن در ارتباط با etcd Data Base برای خواندن جزییات پیکربندی یا نوشتن مقادیر جدید استفاده می‌کند. سرویس Kubelet در هر Node با اجزای اصلی کلاستر(سرور Master ) ارتباط برقرار می‌کند تا در کلاستر احراز هویت شود و دستورات و وظایف را از سرور Master دریافت کند. فرایند Kubelet مسولیت نگهداری و دریافت وظایف است. در این زمان Container ها توسط فرمان های رسیده از سرور Master ایجاد یا حذف می شوند.

**Kube-proxy**

برای مدیریت شبکه و زیرشبکه (subnet) هر سرور در کلاستر و فعال سازی سرویس های دیگر یک سرویس کوچک، به نام Kube-Proxy درهر Node ایجاد می شود. این فرایند، درخواست ارتباطی را به Container مورد نظر داده و می‌تواند Load Balance اولیه را را انجام داده و و مطمئن شود که شبکه ی کلاستر در دسترس و قابل پیش بینی و همچنین ایزوله می باشد.



شکل 5-محیط اجرایی و اجزا Kubernetes

# **4- تعریف پروژه**

**4-1- هدف پروژه**

بطور سنتی سامانه های وب سازمانی بر روی سرورهای فیزیکی سازمان یا سرورهای استیجاری راه‌اندازی می شود. این راهکار برای سامانه های با مراجعات پایین و بار محاسباتی کم راهگشا است. اما اگر بار محاسباتی ارائه سرویس برای هر مشترک بالا باشد مانند ارائه خدمات تجارت الکترونیکی و محتوای چندرسانه‌ای، با افزایش تعداد مراجعات به سامانه و افزایش بار کاری، سرویس ممکن است از دسترس خارج شود. از دسترس خارج شدن سرویس تبعات تجاری زیادی بدنبال دارد و باعث عدم رضایت مشترکین و غیر قابل اطمینان بودن سرویس می شود.

هدف از این پروژه راه اندازی یک وب سایت بر پایه محاسبات ابری و ارکستریشن خودکار این سامانه با استفاده از ابزار های متن باز است. اینکار به نحوی صورت می پذیرد که با استفاده از تولید بار روی سامانه خود بتوانیم دسترس پذیری بالای سامانه خود را حفظ کنیم و سامانه از دسترس پذیری بالا برخوردار باشد. برای این منظور باید با نظارت بر نمونه‌های ایجاد شده از ماشین‌ها یا کانتینرها، وضعیت بار آنها سنجیده شود و درصورت بالا رفتن زمان پاسخ و یا خرابی آنها به ایجاد نمونه های جدید اقدام شود. همچنین در زمان های کم کاری و بیکاری لازم است برای کاهش هزینه سازمان به کاهش تعداد نمونه‌ها اقدام گردد. تنظیم سیاست‌های مسیریابی و VLAN ها با ایجاد نمونه‌های جدید و نیز پیکربندی توزیع بار بین نمونه‌ها لازم است انجام شود.

# **4-2- مراحل انجام کار**

مراحل انجام کار از موارد زیر تشکیل می شود:

الف- راه‌اندازی و انالیز یک سرویس وب سازمانی و آنالیز آن

ب- بسته بندی سرویس وب مورد نظر در قالب ماشین مجازی یا کانتینر

ج- تهیه معماری نرم افزاری سیستم و ابزارهای مورد نیاز

د- راه اندازی، برنامه نویسی و پیکربندی ابزارهای مورد نظر

ه- اجرا و تست سرویس

## **4-3- ابزارها و مهارت‌های موردنیاز**

در این پروژه با ابزارها و زبان‌های متعددی سرو کار خواهیم داشت که از دانش برنامه نویسی، شبکه، محاسبات، مهندسی نرم افزار و معماری فناوری اطلاعات فراگرفته شده در دوره کارشناسی در اجرای آن استفاده به عمل خواهد آمد.

# **5- مراجع**

[1] Al Nuaimi, K., et al. A survey of load balancing in cloud computing: Challenges and algorithms. in Network Cloud Computing and Applications (NCCA), 2012 Second Symposium on. 2012. .IEEE

[2] 'Tutorials', July 21, 2018. [Online]. Available: <https://kubernetes.io/docs/tutorials/>. [Accessed: 1- June- 2019].

[3] Openstack docs: [Online] https://docs.openstack.org

[4] 'Tutorials'. [Online]. Available: <https://aws.amazon.com/getting-started/>. [Accessed: 27- May- 2019].

[5] White Paper, “Introduction to Cloudify: Cloud Orchestration in Today’s Landscape”, Cloudify Corporation, 2017.