

دانشگاه صنعتی اصفهان

مرکز ابررایانش

گزارش کارآموزی



مرکز ابررایانش ملی شیخ بهایی

National High Performance Computing Center



نویسنده: مجتبی ملائی

استادکارآموزی: دکتر فاطمه دلدار
سرپرست کارآموزی: آقای محمد صالح محمدی

۳۱ شهریور ۱۴۰۴

چکیده

من و همکارانم در این دوره کارآموزی ضمن آشنایی با مرکز ابررایانش و افراد متخصص شاغل در این مرکز با استفاده از ابزار های kubernetes, kubervirt و ceph بستری ایجاد کردیم تا بتوانیم ماشین های مجازی ویندوز را با هدف در دسترس بودن کامل^۱ و با قابلیت هایی مانند تخصیص منابع به صورت برخط و امکان خودکار سازی^۲ به صورت ایمن در اختیار مصرف کنندگان قرار دهیم.

فهرست مطالب

۳	۱ معرفی
۳	۱.۱ پیش زمینه من
۳	۲.۱ مرکز ابررایانش شیخ بهائی
۳	۱.۲.۱ منظور از یک خوشه HPC چیست؟
۴	۲.۲.۱ معماری
۴	۳.۲.۱ نحوه کار
۵	۳.۱ پروژه
۶	۲ آماده سازی
۶	۱.۲ آماده سازی کامپیوترها
۶	۲.۲ آماده سازی سرورها
۶	۱.۲.۲ Proxmox
۸	۳ ارکستراسیون
۸	۱.۳ معرفی Kubernetes
۸	۲.۳ نحوه کار
۹	۳.۳ کاربرد در پروژه
۹	۴.۳ جزئیات نصب k8s
۱۱	۴ پیاده سازی شبکه
۱۱	۱.۴ Calico
۱۱	۲.۴ Multus
۱۲	۳.۴ Linux bridge
۱۲	۴.۴ kubemacpool
۱۳	۵ بستر مجازی سازی در k8s
۱۳	۱.۵ Kubevirt
۱۳	۲.۵ منابع ماشین ها
۱۳	۳.۵ انتقال زنده
۱۳	۴.۵ تخصیص منابع به صورت زنده
۱۴	۶ Windows image
۱۴	۱.۶ گرفتن اطلاعات
۱۴	۲.۶ تنظیمات اضافی
۱۵	۷ kubevirt-manager
۱۶	۸ نظارت
۱۶	۱.۸ Prometheus
۱۶	۲.۸ Graphana

آ اطلاعات اضافی

۱۷

فصل ۱

معرفی

۱.۱ پیش‌زمینه من

من کارآموزی‌ام را در بعد از ترم ششم آغاز کردم. در این دوران با برخی از گرایش‌ها، زمینه‌ها و مسیرهای مختلف در حوزه کامپیوتر آشنا شدم که از جمله آنها برنامه نویسی سمت سرور^۱، امنیت سایبری، شبکه و هوش مصنوعی بود. در این میان به فناوری‌های سمت سرور مانند Linux علاقه‌مند شدم و این موضوع باعث شد تا تصمیم بگیرم که کارآموزی‌ام را در حوزه‌های مربوط به سرورها بگذرانم.

۲.۱ مرکز ابررایانش شیخ بهائی

شبکه ملی محاسبات ابری در سال ۱۳۸۸ با هدف ارائه خدمات کامل و جامع محاسباتی به صنایع و مراکز علمی و تحقیقاتی کشور و با حمایت معاونت علمی و فناوری ریاست جمهوری افتتاح شد. مرکز ملی محاسبات ابری^۲ (NHPCC) به عنوان یکی از چندین مرکز ابررایانه کشور، در آن زمان بخشی از این طرح ملی را در قالب تأسیس مرکز ملی ابررایانه شیخ بهائی اجرا کرد. تمهیدات در نظر گرفته شده در مرکز ملی محاسبات ابری شیخ بهائی به گونه‌ای است که امکانات سخت‌افزاری و نرم‌افزاری جامع و کامل و همچنین خدمات آموزشی و مشاوره‌ای فنی و تخصصی برای استفاده بهینه (در محل یا از راه دور) در دسترس باشد.

۱.۲.۱ منظور از یک خوشه HPC چیست؟

محاسبات با کارایی بالا یا HPC^۳ هنر داشتن مجموعه‌ای از گره‌های پردازشی قدرتمند است که به عنوان یک خوشه (یا گروه‌هایی از آنها) به هم متصل شده‌اند تا مجموعه داده‌های عظیم یا مسائل ریاضی پیچیده چند بعدی را به صورت موازی و در سریع‌ترین زمان ممکن حل کنند. بنابراین HPC در مورد محاسبات است. همچنین ممکن است اصطلاح ابررایانه را بشنوید که مترادف خوشه HPC (یا گروه‌هایی از آنها) است. این نام بیشتر به دیدگاه سخت‌افزاری و قدرت آن اشاره دارد که معمولاً به عنوان یک واحد ممیز شناور به نام فلاپ (مانند ترافلاپس، پتافلاپس و ...) محاسبه می‌شود.

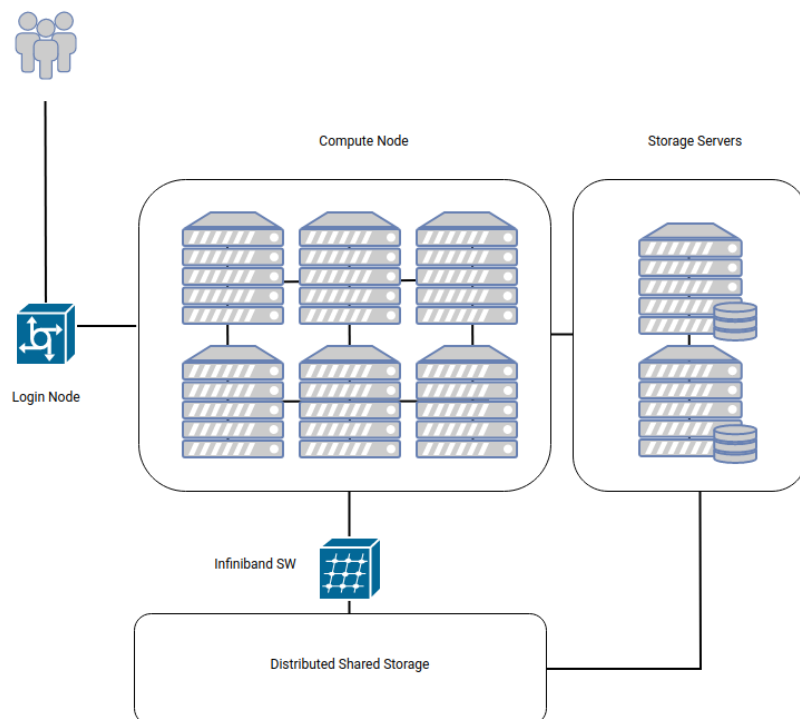
به عنوان مثال، می‌توانید دانش‌آموزی را در نظر بگیرید که یک کد بزرگ (نوشته شده در بیش از هزاران خط) دارد و اگر تمایل داشته باشد کد خود را در لپ‌تاپ شخصی خود اجرا کند، چند سال طول می‌کشد تا پردازش تمام شود و نتیجه نهایی را ببیند؟ بنابراین اگر کد خود را به ابررایانه منتقل کنند، در مدت زمان بسیار کوتاهی به نتایج دست می‌یابد.

^۱back-end

^۲National High Performance Computing Center

^۳High Performance Computing

۲.۲.۱ معماری



شکل ۱.۱: اجزای یک خوشه

معمولاً یک خوشه پایه از این اجزای اصلی تشکیل شده است:

- یک گره ورود (گره سر)
- انواع گره‌های محاسباتی
- یک فضای ذخیره‌سازی مشترک توزیع‌شده
- سرورهای GPU (برای بهره‌مندی از محاسبات گرافیکی عظیم)
- یک سوئیچ Infiniband برای اتصال گره‌ها
- یک زمان‌بند کار و یک سیستم صف‌بندی
- یک گره انتقال
- یک گره پس‌پردازش

۳.۲.۱ نحوه کار

کاربران به گره ورود (گره سر) وارد می‌شوند و فایل‌های ورودی خود را آپلود می‌کنند. سپس کارهای خود را به زمان‌بند ارسال می‌کنند و در صف انتظار قرار می‌گیرند. سپس زمان‌بند شروع به ارسال کارهای آنها به تمام گره‌های محاسباتی دیگری که کاربر مجاز به دسترسی به آنها است، می‌کند.

۳.۱ پروژه

در برخی از موارد مرکز نیاز دارد تا ماشین‌های مجازی ویندوزی را در اختیار مشتریان خود بگذارد. دلیل استفاده از ویندوز راحتی و سادگی بیشتر برای دانشجویان و مشتریانی است که رشته و تخصص آن‌ها در حوزه کامپیوتر نیست و نیاز دارند تا از قدرت محاسباتی مرکز استفاده کنند. برای این افراد کار با سیستم عامل Linux که به طور معمول در خوشه‌ها استفاده می‌شود بسیار دشوار است.

نیاز دیگر این بود که اختصاص ماشین‌ها به مشتریان به صورت خودکار و با کمترین مداخله ممکن از طرف یک نیروی انسانی صورت بگیرد. به همین دلیل استفاده از ابزارهایی مانند Proxmox که در این مرکز استفاده می‌شود ممکن نبود. از طرفی در دسترس بودن و سلامتی این ماشین‌ها مهم‌ترین رکن آن‌ها است. در واقع این پردازش‌ها به دلیل زمان‌بر بودن آن‌ها از اهمیت خاصی برخوردار هستند. به عنوان مثال در صورتی که سرور میزبان دچار خطا شود، ماشین مجازی ویندوز نیز از کار می‌افتد و تمام منابع و زمانی که برای پردازش استفاده شده بود به هدر می‌رود. از دید یک مصرف‌کننده این اتفاق کاملاً غیر قابل تحمل است زیرا که ممکن است مصرف‌کننده زمان لازم برای این اجرای دوباره این پردازش را نداشته باشد. هدف این پروژه برطرف سازی این مشکلات و نیازها بود. در کنار این‌ها قابلیت‌هایی مانند تخصیص منابع به صورت زنده، اضافه کردن و حذف دیسک‌ها به صورت زنده و برقرار سازی سطح مناسبی از امنیت از اهداف مهم دیگر این پروژه بودند.

فصل ۲

آماده سازی

۱.۲ آماده سازی کامپیوترها

در اولین ورود به مرکز، نیاز بود تا کامپیوتر های مورد نیاز خودمان را شخصی سازی کنیم. در واقع میبایست سیستم عامل دیگری بر روی کامپیوترها نصب می کردیم تا اطلاعات قبلی آنها پاک شود و همچنین بتوانیم ابزار های مورد نیاز را روی آنها نصب کنیم. کامپیوترهای مرکز دارای BIOS هستند و همچنین از کارت های گرافیکی قدیمی Nvidia استفاده می کنند. به همین دلیل هنگامی که میخواستیم سیستم عامل های جدید را بر روی آنها نصب کنیم دچار خطا می شدند. بنابراین نهایتا مجبور به استفاده از ISO های قدیمی شدیم. پس از نصب سیستم عامل Manjaro بر روی سیستم ها، باید نرم افزار ها و ابزار های لازم مانند مرورگر، ابزارهای مدیریت بسته، درایورها، کامپایلرها، ابزارهای نصب، ویراشگرهای کد و مدیریت را نصب می کردیم.

۲.۲ آماده سازی سرورها

پس از آماده سازی کامپیوترها، یک سرور قدرتمند به نام "رخش" در اختیار ما قرار گرفت. این سرور شامل ۶۴ هسته پردازشی و ۱ ترابایت حافظه RAM می شد. از ما خواسته شد تا با استفاده از ابزاری به نام Proxmox که یک ابزار برای مدیریت مجازی سازی سطح یک است، چند ماشین مجازی لینوکس ایجاد کنیم. این ماشین های مجازی لینوکسی برای ما سرور های واقعی را شبیه سازی می کردند. در واقع در این پروژه در فاز توسعه، به جای استفاده از چند سرور فیزیکی، از یک سرور فیزیکی چند سرور مجازی ساختیم و از آنها برای توسعه پروژه استفاده کردیم. در این سرور های مجازی ما از Ubuntu Server 2022 به دلیل سازگاری، پایداری بهتر و سادگی استفاده کردیم.

۱.۲.۲ Proxmox

Proxmox VE یک پلتفرم مجازی سازی همه کاره و منبع باز^۱ است که دو فناوری مجازی سازی مبتنی بر کرنل^۲ و مجازی سازی در سطح سیستم عامل^۳ را به صورت یکپارچه در کنار هم ارائه می دهد. این ابزار قدرتمند با ارائه یک رابط مدیریتی تحت وب یکپارچه و کاربر پسند، امکان مدیریت متمرکز ماشین های مجازی و حامل ها،^۴ پیکربندی شبکه های پیچیده، تنظیم سیستم های ذخیره سازی متعدد (مانند Ceph، ZFS) و پیاده سازی قابلیت های خوشه بندی و جابه جایی زنده^۵ را فراهم می کند تا راه حلی مقیاس پذیر، باثبات و حرفه ای برای دیتاسنترهای کوچک و بزرگ ایجاد کند.

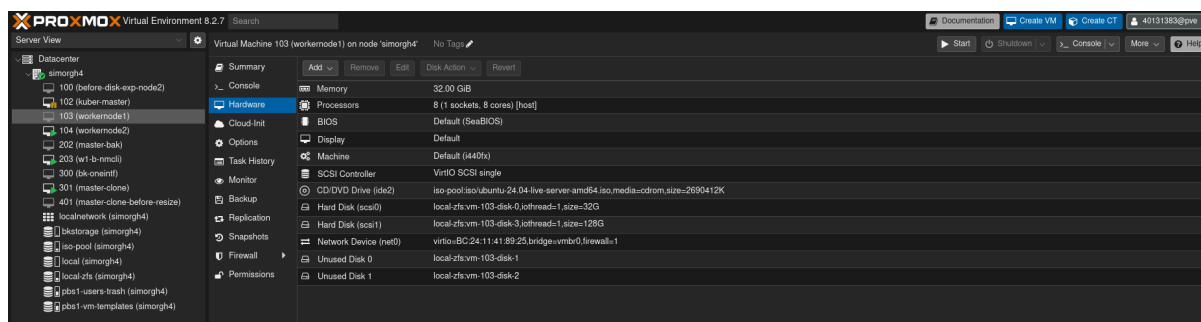
^۱ open-source

^۲ KVM

^۳ LXC

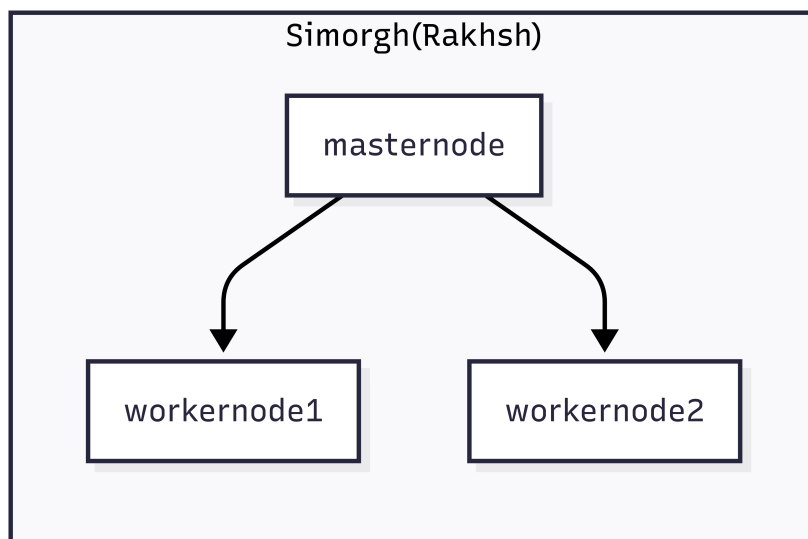
^۴ Containers

^۵ Live Migration



شکل ۱.۲: تصویری از صفحه مدیریت سرورها

همانطور که در تصویر بالا دیده می‌شود. از برخی ماشین‌ها back-up گرفته شده است و برخی از آنها متوقف شده‌اند. امکان back-up گیری و تغییر زنده منابع از مهم‌ترین قابلیت‌های این ابزار است. توپولوژی مورد استفاده ما در این پروژه به شکل زیر بود.



شکل ۲.۲: توپولوژی ماشین‌ها

در واقع ماشین‌های مجازی ویندوز بر روی workernode ها ایجاد می‌شوند و ماشین masternode ماشین‌های دیگر را کنترل می‌کند. در بخش ارکستراسیون به صورت مفصل نحوه این ارتباط توضیح داده خواهد شد.

فصل ۳

ارکستراسیون

ارکستراسیون در دنیای فناوری اطلاعات به فرآیند خودکارسازی استقرار، مدیریت، مقیاس گذاری و شبکه سازی برای حامل ها اطلاق می شود. در یک معماری مبتنی بر میکروسرویس ها، که یک برنامه به ده ها یا صدها جزء مستقل تقسیم می شود، ارکستراسیون نقشی حیاتی در هماهنگ سازی این اجزا، توزیع بار پردازشی، و تضمین در دسترس بودن و مقاومت در برابر خطا^۱ برنامه ایفا می کند. این مفهوم، مدیریت چرخه عمر حامل ها را از سطح یک میزبان منفرد فراتر برده و آن را در مقیاس یک خوشه از سرورها تعریف می کند.

۱.۳ معرفی Kubernetes

Kubernetes یک پلتفرم ارکستراسیون مبتنی بر حامل^۲ متن باز و گسترده است که به عنوان یکی از اصلی ترین راه حل ها در این حوزه شناخته می شود. این پلتفرم با به کارگیری یک معماری Master-Worker^۳، امکان تعریف حالت مطلوب^۴ برنامه را از طریق فایل های اعلانی^۵ فراهم می سازد و سپس به صورت خودکار وظایفی مانند زمان بندی حاملها بر روی گره های کارگر، نظارت بر سلامت آن ها، و بازیابی خودکار را انجام می دهد.

مزایای کلیدی Kubernetes شامل مقیاس پذیری الاستیک^۶ و خودکار (هم به صورت افقی و هم عمودی)، بهینگی در مصرف منابع، قابلیت حمل بالا^۷ بین محیط های ابری مختلف، و تسهیل فرآیندهای DevOps^۸ و CI/CD^۹ است. در نتیجه، این پلتفرم به عنوان استاندارد اصلی برای اجرا و مدیریت برنامه های کاربردی مدرن و مقیاس پذیر در محیط های هایبرید^{۱۰} و چندابری^{۱۱} تبدیل شده است.

۲.۳ نحوه کار

Kubernetes با دریافت تعریف حالت مطلوب برنامه از کاربر، به صورت خودکار فرآیندهای مدیریت و هماهنگی حامل ها را انجام می دهد. کاربران مشخص می کنند که برنامه شان باید چگونه اجرا شود و Kubernetes مسئولیت رساندن سیستم به آن حالت و حفظ آن را بر عهده می گیرد.

سیستم به طور مداوم وضعیت فعلی را با حالت مطلوب مقایسه می کند. اگر بین این دو تفاوتی وجود داشته باشد، Kubernetes به صورت خودکار اقدامات لازم را انجام می دهد. به عنوان مثال، اگر یک حامل از کار بیفتد، سیستم به طور

^۱resilience

^۲Container Orchestration Platform

^۳Control Plane and Worker Nodes

^۴Desired State

^۵Declarative YAML/JSON

^۶Elastic Scalability

^۷High Portability

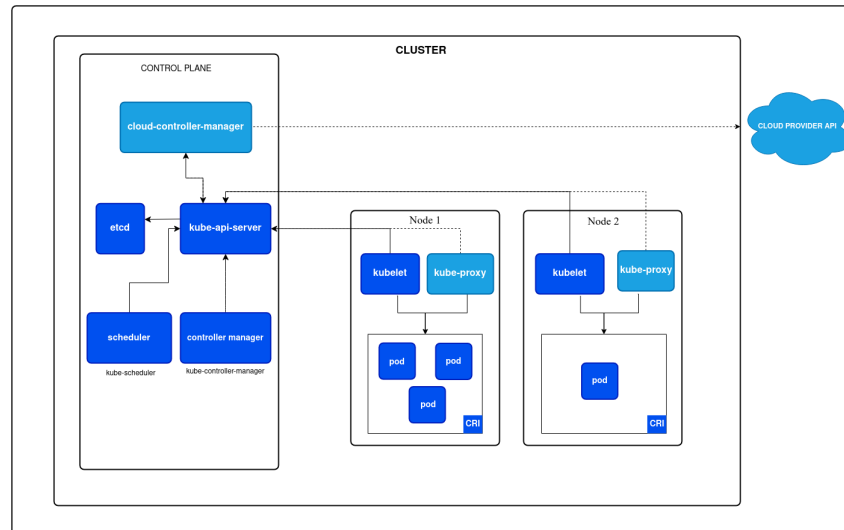
^۸Development and Operations

^۹Continuous Integration and Continuous Delivery/Deployment

^{۱۰}Hybrid Cloud

^{۱۱}Multi-Cloud

خودکار حامل جدیدی را جایگزین آن می‌کند تا برنامه همیشه در حالت مطلوب باقی بماند. Kubernetes حامل‌ها را بین گره‌های مختلف توزیع می‌کند و اطمینان حاصل می‌نماید که از منابع سخت‌افزاری به بهترین شکل ممکن استفاده شود. همچنین شبکه‌های لازم برای ارتباط بین حامل‌ها و تعادل بار بین آنها را به صورت خودکار مدیریت می‌کند تا برنامه به صورت یکپارچه و با کارایی بالا قابل دسترسی باشد.



شکل ۱.۳: معماری کوبرنتیز

ممکن است نام K8s را هم به جای kubernetes شنیده باشید که معنای آن حرف **k** در ابتدا و **۸** حرف میان آن و حرف **s** در انتها است.

۳.۳ کاربرد در پروژه

k8s به دلیل مقاومت در برابر خطا، بازگشت به حالت مطلوب پس از وقوع خطا، مقیاس پذیری و انعطاف پذیری خوب و ساختار ماژولار^{۱۲} آن یک گزینه عالی برای مدیریت ماشین‌های مجازی ویندوزی و دیگر چیزهای مورد نیاز آنها است. در اصل یکی از نیازهای اصلی پروژه که در دسترس بودن دائمی و مقاومت در برابر خطا است که به کمک k8s رفع می‌شود. نکته حائز اهمیت این است که k8s به تنهایی تمام کارها را انجام نمی‌دهد. در واقع این ابزار فقط بستر را برای حامل‌ها فراهم می‌سازد. به دلیل امکان تعریف منابع دلخواه و در نتیجه آن، قابلیت توسعه پذیری و بالای این ابزار، افزونه^{۱۳}‌های بسیار زیادی برای انجام کارهای مختلف وجود دارد. برخی از این افزونه‌ها مربوط به سیستم داخلی خود k8s است مانند افزونه‌های شبکه که در ادامه معرفی می‌شوند و یا افزونه‌هایی که یک سرویس کامل را اجرا میکنند مانند kubevirt که در ادامه پروژه به آن نیاز داشتیم.

۴.۳ جزئیات نصب k8s

در هنگام نصب k8s نکات زیادی باید رعایت شوند. مثلاً بسته به نوع اینکه گره اصلی^{۱۴} است یا کارگر^{۱۵} دستورات متفاوتی وجود دارد. اما همه گره‌ها پیش‌نیازهای مورد نیاز یکسانی را دارند. مثلاً حداقل منابع **RAM** و **CPU** و **Disk** و بعد از آن یک شبکه که در آن گره‌ها بتوانند بهم متصل شوند، همچنین خاموش بودن **Swap** و بارگذاری یکسری کرنل ماژول‌ها^{۱۶} و باینری‌های مورد نیاز از جمله این پیش‌نیازها هستند. بعد از آنکه این پیش‌نیازها فراهم شد، با استفاده از دستورات مربوط به پیوستن گره‌ها گره‌های کارگر را به گره اصلی متصل می‌کنیم.

^{۱۲}Modular

^{۱۳}plugins

^{۱۴}master

^{۱۵}worker

^{۱۶}Kernel modules

username	hostname	NIC(ens18)	NIC(ens19)	role	CPU	RAM	primary disk
master	masternode	192.168.170.190	-	control plane	4	10 GB	64 GB
worker1node	worker1node	192.168.170.191	used for br1	worker	8	32 GB	64 GB
worker2node	worker2node	192.168.170.192	used for br1	worker	8	32 GB	64 GB

جدول ۱۰.۳: مشخصات سرورهای مجازی در ۸۸

جدول بالا ویژگی‌ها و مشخصات هر گره را نشان می‌دهد. در فصل بعدی قسمت شبکه هر نود کاملاً توضیح داده خواهد شد.

فصل ۴

پیاده‌سازی شبکه

بحث شبکه یکی از چالشی ترین قسمت های این پروژه بود. زیرا ما سه شبکه متفاوت را مدیریت می کردیم. شبکه اول، شبکه داخلی و اصلی بین گره ها است. این شبکه بر روی رابط ^۱ens18 هر گره و با آدرس آیپی ^۲192.168.170.19x/24 به صورت آیپی ثابت مشخص شده است. این تنظیمات بر روی netplan هر گره نوشته شده است و توسط گره ها اجرا می شود. شبکه دوم شبکه داخلی k8s است که بر پایه شبکه اول پیاده سازی شده است. مدیریت این شبکه با calico است که یک CNI برای مدیریت شبکه داخلی k8s است. این شبکه برای ارتباط بین ^۳pod ها استفاده می شود. رنج آیپی آن نیز 10.0.0.0/8 است. این شبکه به دلیل داخلی بودن قابل دسترسی از بیرون k8s نیست. شبکه سوم نیز برای ماشین های ویندوز در نظر گرفته شده است. این شبکه به صورت عمومی باشد تا کاربران بتوانند به آن وصل شوند. چون دو شبکه دیگر کاملاً داخلی هستند. این شبکه بر روی رابط ^۴ens19 است و رنج آیپی آن نیز 192.168.172.0/24 است که توسط DHCP سرور آیپی ها به ماشین های ویندوز اختصاص داده می شوند.

۱.۴ Calico

calico یک افزونه شبکه ای برای k8s است که به عنوان یک CNI^۴ عمل می کند. وظیفه اصلی آن فراهم کردن اتصال شبکه ای میان پادهاست، به طوری که همه پادها بتوانند در سراسر خوشه یکدیگر را ببینند و داده تبادل کنند. calico با استفاده از BGP^۵ و مکانیزم های ساده اما کارآمد مسیریابی، به جای تونل زنی پیچیده، مسیرهای ارتباطی را مدیریت می کند. این موضوع باعث می شود که هم کارایی بالاتری داشته باشد و هم عیب یابی شبکه راحت تر انجام شود. علاوه بر این، calico قابلیت پیاده سازی سیاست های امنیتی شبکه (Network Policy)^۶ را هم دارد که به مدیر خوشه اجازه می دهد ترافیک میان سرویس ها را محدود یا مجاز کند.

در این پروژه هم از Calico به عنوان CNI اصلی خوشه به دلیل سرعت بالا در مقیاس بالا برای پادها استفاده شده است. هر پاد در تنظیمات خود یک رابط به نام eth0 دارد که برای این ارتباط با این شبکه است. همچنین هر پاد یک آدرس آیپی از شبکه 10.0.0.0/8 می گیرد.

۲.۴ Multus

Multus یک افزونه CNI است که به عنوان یک رابط استاندارد برای اتصال پادها به شبکه عمل می کند. تفاوت اصلی آن با افزونه های ساده تر این است که به هر پاد اجازه می دهد بیش از یک رابط شبکه داشته باشد. به این ترتیب، یک پاد می تواند هم زمان به چندین شبکه متصل شود؛ مثلاً یک شبکه اصلی برای ارتباطات داخلی خوشه و یک یا چند شبکه دیگر برای دسترسی به سرویس های خاص یا منابع خارجی.

^۱interface

^۲IP Address

^۳یک منبع تعریف شده در k8s که یک یا تعدادی حامل را در خود نگه می دارد. کوچک ترین واحد محاسباتی در k8s

^۴Container Network Interface: رابط استاندارد برای اتصال حامل ها به شبکه

^۵Border Gateway Protocol: پروتکل مسیریابی بین شبکه ها

^۶Network Policy: مجموعه ای از قوانین امنیتی برای کنترل ترافیک بین پادها

این قابلیت به‌ویژه در سناریوهای پیشرفته، مانند شبکه‌های مخابراتی (NFV) یا محیط‌های چندمستأجری^۷، کاربرد زیادی دارد. Multus همچنین می‌تواند با سایر افزونه‌های CNI (مانند Calico یا Flannel) ترکیب شود و امکان مدیریت انعطاف‌پذیرتر و پیشرفته‌تر شبکه‌ها را در یک خوشه فراهم کند.

همانطور که قبلاً گفته شد، نیاز داریم تا یک شبکه عمومی برای ماشین‌های ویندوز در نظر بگیریم تا مجزا از شبکه‌های داخلی باشند و امکان دسترسی از خارج نیز وجود داشته باشد. پس بنابراین نیاز داشتیم تا به صورتی یک رابط دیگر نیز برای این شبکه ایجاد کنیم. برای اینکه بتوانیم دو رابط داشته باشیم ناچار هستیم از این CNI استفاده کنیم.

۳.۴ Linux bridge

پل در لینوکس یا Linux Bridge در واقع یک ابزار نرم‌افزاری در هسته‌ی لینوکس است که رفتار یک سویچ لایه‌ی دوم^۸ را شبیه‌سازی می‌کند. این پل می‌تواند چندین واسط شبکه (چه فیزیکی و چه مجازی) را به یکدیگر متصل کند و آن‌ها را در یک دامنه‌ی پخش^۹ مشترک قرار دهد. در نتیجه، بسته‌های داده بر اساس آدرس مقصد MAC بین واسط‌ها جابه‌جا می‌شوند، درست مانند اینکه همه‌ی آن‌ها به یک سویچ واقعی متصل شده باشند.

ما برای اینکه ماشین‌های مجازی ویندوز را به شبکه عمومی متصل کنیم، نیاز داریم تا به طریقی آن‌ها را به آن شبکه از طریق رابط ens19 متصل کنیم. در حالت عادی ای این رابط فقط بسته‌هایی را ارسال و دریافت می‌کند که با آدرس MAC آن همخوانی داشته باشند. بنابراین قادر به انتقال بسته‌های ماشین‌های ویندوز که MAC متفاوتی دارند نیست. در نتیجه ما این رابط را به یک پل تبدیل می‌کنیم که ماشین‌های ویندوز نیز از طریق آن بتوانند بسته‌های خود را جابه‌جا کنند. برای استفاده از این پل ها کافی است آن‌ها را هم در netplan تعریف کنیم. پس از اجرای آن تنظیمات، یک پل در لیست رابط‌ها ایجاد می‌شود. در سمت k8s نیز باید یک رابط شبکه (NAD)^{۱۰} به این پل ایجاد کنیم تا پادها بتوانند با آن ارتباط برقرار کنند. ما میتوانیم به ens19 آییی بدهیم. اما ترجیح دادیم به آن هیچ آدرس آییی اختصاص ندهیم تا کاملاً همانند یک پل واقعی عمل کند.

۴.۴ kubemacpool

یکی از مشکلاتی که بعداً ایجاد شد، این بود که پس از هر بار خاموش روشن کردن ماشین‌های ویندوز، یک آدرس MAC رندوم به ماشین مجازی اختصاص داده میشد و DHCP سرور مجبور بود که هر بار یک آدرس IP دیگر به ماشین ویندوز بدهد. در این حالت یک ماشین ویندوز پس از هر بار خاموش روشن شدن یک آدرس IP جدید داشت که مطلوب نبود. kubemacpool یک افزونه است که دقیقاً برای حل همین مشکل ایجاد شده است. ابتدا یک رنج آدرس MAC برای آن تعریف می‌کنیم. سپس این افزونه خودش به هر ماشین یا پاد یک آدرس MAC از آن رنج مشخص شده می‌دهد و این آدرس ثابت می‌ماند. بنابراین DHCP سرور نیز آدرس IP قبلی ماشین را به آن اختصاص می‌دهد. همچنین مدیریت اینکه آدرس‌های MAC در شبکه یکتا بمانند نیز توسط همین افزونه انجام می‌شود.

^۷Multi-tenant

^۸Switch در لایه‌ی پیوند داده کار می‌کند و بر اساس آدرس‌های MAC تصمیم به ارسال بسته‌ها می‌گیرد.

^۹Broadcast Domain: محدوده‌ای که پیام‌های پخش‌شده به همه‌ی دستگاه‌ها می‌رسد.

^{۱۰}Network Attachment definition

فصل ۵

بستر مجازی سازی در k8s

۱.۵ Kubevirt

۲.۵ منابع ماشین ها

۳.۵ انتقال زنده

۴.۵ تخصیص منابع به صورت زنده

در این فصل، ما به بررسی نحوه اجرای ماشین های مجازی در کلاسترهای k8s می پردازیم. در ابتدا، ما به معرفی Kubevirt می پردازیم که یک پلتفرم برای اجرای ماشین های مجازی در کلاسترهای k8s است. سپس، ما به بررسی نحوه تخصیص منابع به ماشین های مجازی می پردازیم. در ادامه، ما به بررسی نحوه انتقال زنده ماشین های مجازی می پردازیم. در نهایت، ما به بررسی نحوه تخصیص منابع به صورت زنده می پردازیم.

فصل ۶

Windows image

۱.۶ گرفتن اطلاعات

۲.۶ تنظیمات اضافی

00000000 000 00000, 0000000000 000 00000 000000000, 0000000000 000000000 00000 00000000, 0000000
 000000000000 000000, 000000000 0, 000000000000000 000000, 00000000 000000000 00000, 00000 00000 00000
 0000000000 00000000000, 000000000000000 00000 000000 00000000 000 000000000000000 0000000, 00,
 00000000 000 00000 00 000 0000000000 000000, 00000000000 00 00000, 00000 000 00000 00000000000
 0000000000 00 00000000 000000 00000 00000000000000 00000, 0000000000 000000000 000000000000 000000000,
 000 00 00000 00000 000 0000000000000000 000000; 000000000 000000000 0000000000 00 0000000 0000
 000000 00000 000000000, 00000000000 000 00000 00000000000, 00000000 000000000000000, 0000000000
 000000, 00 00000 000000000000 000000000000 000000, 0 000000 00000 000000000 00 00000000, 0000000000
 000000000 0000000000, 00000000000 00000 00 00000 00 000000 0000000, 0000000 00000000000, 000000000000
 000000000000000000 00000, 000000000000000 00000000 00000 00000000, 0000000000 000000000000 00000 00000000,
 0000000, 000 000000 0000000, 000 000000 000000 000000000000

فصل ٧

kubevirt-manager

فصل ۸

نظارت

Prometheus ۱.۸

Graphana ۲.۸

در این فصل، ما به بررسی ابزارهای نظارت می‌پردازیم. Prometheus یک سیستم نظارت و هشدار است که به شما امکان می‌دهد تا داده‌های متریک را جمع‌آوری و ذخیره کنید. Graphana یک ابزار بصری‌سازی داده است که به شما امکان می‌دهد تا نمودارهای زیادهای از داده‌های Prometheus را ایجاد کنید. ما به شما خواهیم آموخت که چگونه این ابزارها را نصب و پیکربندی کنید و چگونه از آنها برای نظارت بر سیستم‌های خود استفاده کنید.

پیوست آ

اطلاعات اضافی

معمولاً، در صورتی که یک شرکت یا سازمان به شما اطلاع دهد که شما را به یک وبسایت یا یک برنامه دعوت کرده است، این یک نشانه است که شما را به یک وبسایت یا یک برنامه دعوت کرده است. این می تواند به شما کمک کند تا با شرکت یا سازمانی که شما را دعوت کرده است، ارتباط برقرار کنید. این می تواند به شما کمک کند تا با شرکت یا سازمانی که شما را دعوت کرده است، ارتباط برقرار کنید. این می تواند به شما کمک کند تا با شرکت یا سازمانی که شما را دعوت کرده است، ارتباط برقرار کنید.