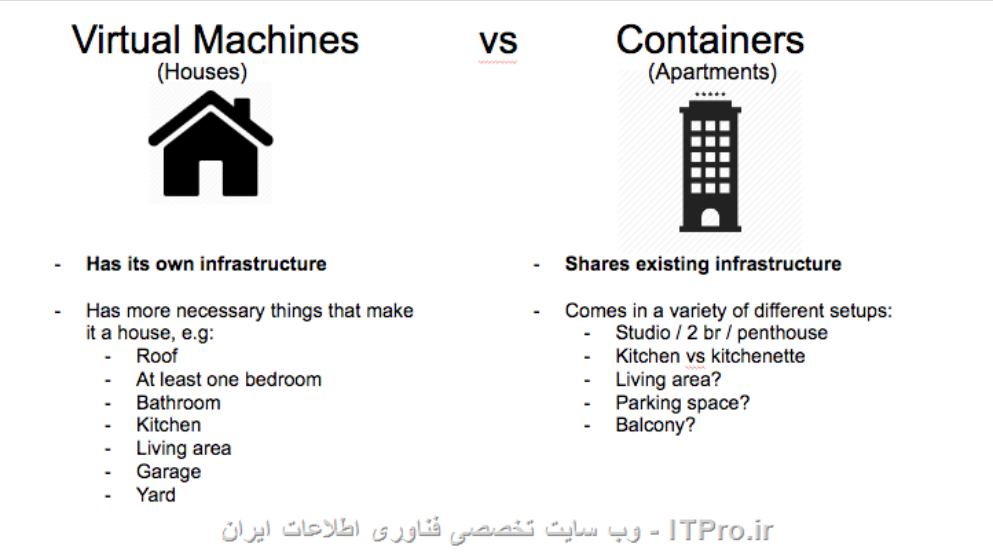
آموزش داکر (Docker) قسمت ۱ : مقایسه Container و VM

Docker یکی از موفق ترین پروژه های متن باز در تاریخ فناوری اطلاعات است. سازمان ها همواره در تلاش برای افزودن قابلیت حمل به برنامه های کاربردی خود از طریق Container ها هستند ، اما اولین قدم برای رسیدن به این هدف ، درک container ها و مزایای کلیدی آنهاست.اغلب افرادی که برای اولین بار با Docker Container کار می کنند به آن "ماشین مجازی سبک وزن" می گویند! پس به آسانی قابل فهم است که این دو تکنولوژی مشخصات مشابهی دارند. اما مشخصات مشابه چه مواردی هستند؟!! هر دو طراحی شده اند تا بتوانند:

- محیطی ایزوله برای برنامه های کاربردی فراهم کنند.
- به راحتی بین میزبان ها (Hosts) جابجا شوند.

در واقع معماری ماشین مجازی و Container از پایه با هم متفاوت است! جهت درک بهتر موضوع میشه این مثال و مقایسه رو ارائه داد:

- خانه ها ====> [ماشین های مجازی]
- آپارتمان ها ====> [Docker Containers]



خانه ها [VMs] کاملا مستقل هستند و قابلیت جلوگیری از ورود مهمان های ناخواسته را برای محافظت از خود ارائه می کنند.آنها همچنین زیرساخت های خود را دارند - لوله کشی، گرمایش، برق، و غیره ، علاوه بر این، در بیشتر موارد، خانه ها دارای حداقل یک اتاق خواب،هال و پذیرایی، حمام و آشپزخانه هستند.بسیار دشوار است که یک "خانه استودیویی" پیدا کنید - حتی اگر شخصی یکی از کوچکترین خانه هایی که می تواند پیدا کند را بخرد ، ممکن است بیشتر از آنچه نیاز دارد خرید کند و به بعضی از قسمت های خانه اصلا نیازی نداشته باشد. زیرا خانه ها فقط ساخته می شوند.(همه ی آن ها سفارشی ساخته نمی شوند - مثل سخت افزار سرور های مختلف که شرکت های

سخت افزاری آن هارا برای استفاده عمومی (general-purpose) می سازندو ممکن است از همه ی منابع استفاده نکنید.)

آپارتمان ها (Docker Containers) نیز در برابر مهمانان های ناخواسته از خود محافظت می کنند، اما آنها بر روی زیرساخت مشترک ساخته شده اند.ساختمان آپارتمان یا Docker Host نیز همانند ماشین مجازی زیرساخت خود را دارد لوله کشی، گرمایش، برق، و غیره به هر آپارتمان.(سروری که سرویس شبه داکر(Docker-Daemon) در آن در حال اجراست به عنوان Docker Host یا میزبان داکر شناخته می شود.)علاوه بر این، آپارتمان ها در اندازه های مختلف ارائه می شوند - از استودیو تا پنت هاوس چند خوابه و شما تنها دقیقا همان چیزی را که نیاز دارید اجاره می کنید.Docker Container ها از منابع سخت افزاری به اشتراک گذاشته شده Docker Host استفاده می کنند. به علاوه، توسعه دهندگان Docker Image می سازند و Docker Image دقیقا همان چیزی است که آنها برای اجرای برنامه خود نیاز دارند: توسعه دهندگان قادرند تا به سرعت به برنامه اولیه ویژگی های جدید را اضافه کنند.)ماشین های مجازی در جهت مخالف ساخته شده اند. آنها با یک سیستم عامل شروع به کار می کنند و بسته به برنامه کاربردی، توسعه دهندگان ممکن است قادر نباشند اجزای ناخواسته را از بین ببرند یا با کندی میتوانند ویژگی جدید را اضافه کنند.برای بسیاری از افراد این مفاهیم به راحتی قابل درک هستند. با این حال، حتی زمانی که با تفاوت معماری Docker containers و ماشین های مجازی آشنا می شوند اغلب در تلاش هستند تا هرآنچه درباره ی VM می دانند با Container انطباق دهند مثلا:

۱. چطور میتوانم از Container پشتیبان بگیرم؟
۲. از کدام استراتژی مدیریتی برای اعمال patch در container های در حال اجرا استفاده کنم؟
۳. Application server کجا در حال اجراست؟

البته در پایان متوجه خواهند شد که Docker تکنولوژی مجازی سازی نیست، بلکه یک تکنولوژی تحویل برنامه است (Application Delivery Technology) اگر VM را به صورت واحدی انتزاعی و یکپارچه (Monolithic) در نظر بگیریم در دنیای ماشین های مجازی نه تنها کد برنامه ذخیره می شود، بلکه داده ها (stateful data) نیز ممکن است به همراه آن در VM ذخیره شوند.ماشین مجازی منابع دریافتی از سرور فیزیکی را گرفته و به صورت Binary بسته بندی می کند ، بنابریان می تواند آن را براحتی انتقال دهد.(انتقال OS+APP :معمولا حجم انتقال بالا و بسیار زمان بر خواهد بود)در Container ها واحد انتزاعی برنامه کاربردی است، در بیان دقیق تر سرویسی است که کمک می کند تا برنامه کاربردی ساخته شود. در معماری Micro-service ، بسیاری از سرویس های کوچک (که هرکدام به عنوان یک Docker Container نمایان می شوند) یک برنامه را می سازند.اکنون برنامه ها می توانند به اجزای بسیار کوچکتر شکسته شوند، این امر توسعه و مدیریت محصول را از پایه و اساس تغییر می دهد.

بنابراین، در پاسخ به این سوال که "یک sysadmin چگونه از Docker Container پشتیبان می گیرد؟" می توان گفت که او نیازی به انجام این کار ندارد. داده تولید شده توسط برنامه کاربردی در Container وجود ندارد بلکه داده ها در Volume ای وجود دارند که بین Container ها از طریق معماری نرم افزار تعیین شده توسعه دهندگان به اشتراک

گذاشته می شود و Sysadmin ها تنها از Volume ها پشتیبان می گیرند و Container هارا فراموش می کنند زیرا Container ها Stateless و غیر قابل تغییر (Immutable) بوده و داده ای را درخود ذخیره نمی کنند.مطمئنا اعمال patch هنوز هم بخشی از دنیای مدیران سیستم است، اما این امر روی Container های در حال اجرا انجام نمی گردد.در حقیقت اگر کسی یک Container در حال اجرا را پتچ کند و سپس Container جدیدی را بر اساس یک Docker Image پتچ نشده اجرا کند، همه چیز به هم خواهد ریخت.به جای اینکار مدیران سیستم Docker Image موجود خود را به روز کرده سپس Container های در حال اجرا خود را متوقف کرده و Container های جدید را اجرا می کنند. از آنجا که یک Container می تواند در کسری از یک ثانیه متوقف و اجرا شود،بنابریان این به روز رسانی ها بسیار سریع تر از ماشین های مجازی انجام می شوند.در پاسخ به سوال سوم می توان گفت که Application Server نیز به سرویسی درون Container تبدیل خواهد شد.مطمئنا ممکن است مواردی وجود داشته باشد که برنامه های مبتنی بر معماری Micro-service نیاز به اتصال به سرویس غیر کانتینری داشته باشند، اما عمدتا برای اغلب سرورهای مستقل که کد برنامه در آن اجرا می شود، می توان از یک یا چند کانتینر برای آن عملکرد مشابه استفاده کرد که این رویکرد دو مزیت برایمان خواهد داشت:

۱-کاهش سربار

۲-افزایش مقیاس پذیری افقی برنامه ها (لطفا جهت درک بهتر مقیاس پذیری افقی شکل زیر را مشاهده کنید)

نویسنده : محمد فعال فرد

منبع : انجمن تخصصی فناوری اطلاعات ایران