علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۲۱، شماره ۲، ۱۴۰۲
صفحات ۲۹-۳۸
مقاله پژوهشی

مطالعه‌ای تطبیقی از مجازی‌سازی

: مجازی‌سازی سنگین‌وزن یا سبک‌وزن؟

حسین آقاسی زاده[۱]، احسان عطائی[۲*]، مصطفی بستام[۲]

*نویسنده مسئول، دریافت: ۰۲/۰۶/۱۸، بازنگری: ۰۲/۰۹/۱۱، پذیرش: ۰۲/۱۰/۱۰

[۱] دانشجوی کارشناسی ارشد، گروه مهندسی کامپیوتر، دانشکده مهندسی و فناوری، دانشگاه مازندران، بابلسر، ایران
[۲] استادیار، گروه مهندسی کامپیوتر، دانشکده مهندسی و فناوری، دانشگاه مازندران، بابلسر، ایران

چکیده

سال‌هاست که مجازی‌سازی به‌عنوان روشی برای به اشتراک‌گذاری منابع سخت‌افزاری یا نرم‌افزاری بین کاربران مطرح بوده است. اما اهمیت مضاعف مجازی‌سازی ازآن‌رو است که در اکثر پارادایم‌های محاسباتی که در دو دهه اخیر پدید آمده‌اند، جایگاه ویژه و نقش غیرقابل‌انکاری داشته است. در این مقاله، مجازی‌سازی سنگین‌وزن (مبتنی بر ماشین‌های مجازی) و سبک‌وزن (مبتنی بر کانتینرها) از منظر استفاده از منابع مختلف سخت‌افزاری سیستم به‌صورت تجربی با یکدیگر مقایسه شده‌اند. بررسی مجازی‌سازی سنگین‌وزن در این مقاله، پارا مجازی‌سازی و مجازی‌سازی کامل را در برمی‌گیرد؛ همچنین، مجازی‌سازیِ نوع یک و نوع دو پوشش داده‌شده است. در این راستا، دو ناظر ماشین مجازی Xen و KVM موردمطالعه قرارگرفته‌اند. در بررسی مجازی‌سازی سبک‌وزن نیز سه پلتفرم اصلی و نوظهور کانتینر سازی، شامل Docker، Podman و LXC موردمطالعه قرارگرفته‌اند. همچنین از ماشین بومی (فاقد مجازی‌سازی) به‌عنوان پایه ارزیابی استفاده‌شده است. آنگاه، عملکرد همه پیکربندی‌ها در استفاده از منابع سخت‌افزاری اصلی شامل پردازنده، حافظه اصلی، دیسک و شبکه در محیط آزمایش واقعی توسط اجرای پرکاربردترین بنچمارک‌های رایج مورداندازه‌گیری، مقایسه و تحلیل قرارگرفته است.

کلمات کلیدی: ارزیابی کارایی، بنچمارک، کانتینر، ماشین مجازی، مجازی‌سازی.

۱- مقدمه

طی سال‌های اخیر، واژه مجازی‌سازی بیش‌ازپیش شنیده می‌شود و به موضوعی جذاب و پرکاربرد برای محققان و شرکت‌های بزرگ تبدیل‌شده است. پارادایم رایانش ابری ذاتاً بر پایه فناوری‌های مجازی‌سازی بنانهاده شده است. اگرچه مفهوم ماشین مجازی حدود دو دهه است که در این زمینه مطرح بوده است، اخیراً فناوری‌های نوظهور سبک‌وزن‌تری مانند کانتینرها، به‌طور فزاینده‌ای محبوب و در رایانش ابری ادغام‌شده‌اند [۱]. ارائه‌دهندگان سرویس‌های ابری، مجازی‌سازی را به‌منظور بهبود بهره‌وری و مدیریت منابع، صرفه‌جویی در هزینه، افزایش قابلیت اطمینان، قابل‌حمل بودن و همچنین کاهش مصرف انرژی و هزینه مراکز داده استفاده می‌کنند [۲]. به لطف این محبوبیت، سال‌های زیادی است که مجازی‌سازی، موضوعی جذاب برای محققان بوده است. درواقع از اوایل سال ۱۹۹۰، تغییر پارادایم از واژه "واقعی" به "مجازی"، منجر به ایجاد علایق تحقیقاتی متنوعی در این زمینه شده است. به‌طورکلی می‌توان گفت دو شیوه اصلی مجازی‌سازی وجود دارد: مجازی‌سازی مبتنی بر سخت‌افزار و مجازی‌سازی مبتنی بر سیستم‌عامل [۳]. مجازی‌سازی مبتنی بر سخت‌افزار روشی است که در آن از ماشین‌های مجازی برای ایجاد نسخه‌های مجازیِ محیط فیزیکی استفاده می‌شود. در مدل مبتنی بر سیستم‌عامل، کانتینرها، کتابخانه‌ها و درایورها یا باینری‌های وابسته به هسته سیستم‌عامل را به اشتراک می‌گذارند. مجازی‌سازی مبتنی بر سیستم‌عامل به این ویژگی اشاره می‌کند که در هسته، وجود فضای نمونه مجزای مختلف از فضای نمونه کاربر را امکان‌پذیر می‌کند.

بنابراین، با روش سنتی ماشین مجازی متفاوت است. در حالت کلی دو نوع ناظر ماشین مجازی یا هایپروایزر وجود دارد: نوع یک و نوع دو. هایپروایزر زمانی که مستقیماً روی سخت‌افزار نصب می‌شود و شکلی از سیستم‌عامل یا هسته را ارائه می‌دهد، نوع یک است؛ و زمانی که به‌عنوان یک برنامه روی سیستم‌عامل نصب می‌شود، نوع دو محسوب می‌گردد. جدول ۱ درک بهتری از تفاوت‌های هایپروایزر نوع یک و دو را نشان می‌دهد.

جدول ۱- تفاوت ماشین مجازی نوع ۱ و نوع ۲

معیار	نوع ۱	نوع ۲
روش اجرا	نصب روی سخت‌افزار	نصب روی سیستم‌عامل
بوت شدن	بوت شدن قبل از سیستم‌عامل	بوت شدن بعد از سیستم‌عامل
بهره‌وری	بالاتر	پایین‌تر
دسترسی	بالاتر	پایین‌تر
کارایی	بالاتر	پایین‌تر
امنیت	بالاتر	پایین‌تر
استفاده معمول	در مراکز داده	در ماشین‌های کاربران

در کنار این‌ها، فنونی برای مجازی‌سازی وجود دارد که معروف‌ترین آن‌ها مجازی‌سازی کامل[1] یا ماشین مجازی سخت‌افزاری[2] (HVM) و پارامجازی‌سازی[3] (PV) هستند. مجازی‌سازی کامل، یک شبیه‌سازی کامل از سخت‌افزار است که می‌توان مستقیماً به سخت‌افزار مورداستفاده هایپروایزر دسترسی داشت، درصورتی‌که پارامجازی‌سازی توسط رابط برنامه‌نویسی کاربردی اجرا می‌شود که به یک سیستم‌عامل اصلاح‌شده نیاز دارد [۴]. از مزایای فن مجازی‌سازی کامل می‌توان به قابلیت حمل، سازگارتر بودن و پشتیبانی از سیستم‌عامل‌های مهمان بدون نیاز به تغییر آن‌ها اشاره کرد. درحالی‌که در پارامجازی‌سازی به‌جای تلاش برای شبیه‌سازی کل محیط سخت‌افزاری، یک سیستم‌عامل مهمان قبل از نصب در داخل ماشین مجازی تغییر داده می‌شود. بدین شکل، به همه سیستم‌عامل‌های مهمان در سیستم اجازه داده می‌شود تا منابع را به اشتراک بگذارند و با یکدیگر همکاری کنند. نرم‌افزارهای ناظر ماشین مجازی یا هایپروایزر متعددی در دنیای امروز رایج هستند که از معروف‌ترین آن‌ها می‌توان KVM [۵] و Xen [۶] را نام برد. تفاوت‌های این دو ناظر ماشین مجازی در جدول ۲ موردبررسی قرارگرفته است. در حالیکه KVM از فن مجازی‌سازی کامل استفاده می‌نماید، Xen فنی دیگر به نام PVHVM را معرفی کرده است که به‌نوعی ترکیب پارامجازی‌سازی و مجازی‌سازی سخت‌افزار محسوب می‌شود. هدف اصلی فن PVHVM، افزایش کارایی مهمانان HVM کاملاً مجازی شده از طریق استفاده از برخی درایورهای فن پارامجازی‌سازی است. لازم به ذکر است که درواقع PVHVM، درایورهای PV بر روی HVM یعنی به عبارتی درایورهای PV بهینه‌شده برای محیط‌های HVM هستند که شبیه‌سازی دیسک و شبکه را دور می‌زنند.

کانتینر یک واحد استاندارد نرم‌افزار است که کد و تمام وابستگی‌های آن را بسته‌بندی می‌کند تا برنامه به‌سرعت، با اطمینان و باقابلیت حمل بالا در محیط دلخواه اجرا شود. از معروف‌ترین بسترهای ایجاد و مدیریت کانتینرها می‌توان به Docker [۷]، Podman [۸] و LXC [۹] اشاره کرد. LXC و Docker پلتفرم‌های شناخته‌شده‌ای هستند؛ ولی Podman نسبت به آن‌ها جدیدتر است. Podman یک موتور کانتینری بدون نیاز به دسترسی سامانه‌ای است که توسط شرکت RedHat [۱۰] توسعه‌یافته و به‌عنوان جایگزینی برای پلتفرم معروفِ Docker ارائه‌شده است.

جدول ۲- تفاوت ماشین‌مجازی KVM و Xen

معیار	KVM	Xen
مجازی‌سازی	HVM	HVM, PV, PVHVM
سیستم‌عامل	Linux	Linux, UNIX
پایداری	پایین‌تر	بالاتر
کارایی	پایین‌تر	بالاتر

رویکرد عدم نیاز به دسترسی سامانه‌ای، امکان اجرای کانتینر توسط کاربر را مهیا می‌سازد [۱۱]. این عامل باعث شده تا Podman از امنیت بیشتری نسبت به Docker برخوردار شود و طبق گفته خودشان بتواند جایگزین مناسبی برای Docker باشد. البته ناگفته نماند که برای کانتینرهای بدون نیاز به دسترسی سامانه‌ای، همانند Podman حالت شبکه پیش‌فرض slirp4netns است. slirp4netns یک پشته شبکه مجازی فراهم می‌کند و به کانتینرها اجازه می‌دهد آدرس IP خود را داشته باشند و به اینترنت و سایر کانتینرها متصل شوند. این شبکه کانتینر را بدون نیاز به دسترسی سامانه‌ای ساده می‌کند. جدول ۳ تفاوت‌های عمده بین این پلتفرم‌های کانتینر سازی را نمایش می‌دهد.

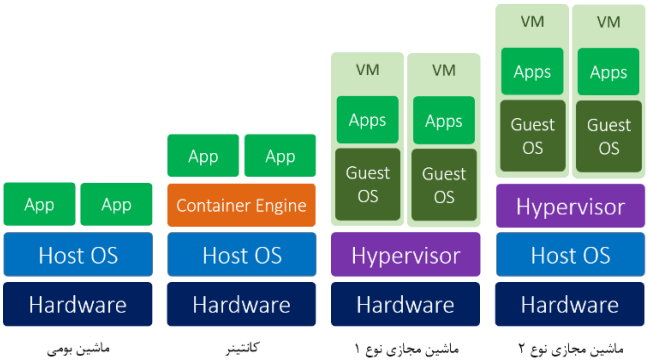
جدول ۳- تفاوت کانتینرهای مختلف

معیار	Podman	LXC	Docker
پلتفرم‌های پشتیبانی شده	Linux, macOS, Windows	Linux	Linux, macOS, Windows
امکان بازیابی اطلاعات	بله	خیر	بله
نیاز به مجوز سامانه‌ای	خیر	خیر	بله
مجازی‌سازی	Application virtualization	Operating system virtualization	Application virtualization
سرعت استقرار	سریع	کند	سریع

این مقاله، کارایی ماشین‌های مجازی بر روی Xen -که به‌صورت نوع یک- و KVM -که به‌صورت نوع دو اجراشده است- را از جوانب مختلف بررسی می‌کند؛ سپس این عمل برای کانتینرهای مختلف مبتنی بر Docker، LXC و Podman انجام می‌شود تا هر دو مدلِ اصلی مجازی‌سازی -مبتنی بر ماشین مجازی و مبتنی بر کانتینر- بررسی گردد. برای این منظور، با استفاده از بنچمارک‌های شناخته‌شده و رایج، کارایی استفاده از پردازنده، حافظه اصلی، دیسک و شبکه موردبررسی و ارزیابی قرار می‌گیرد. همچنین کارایی پیکربندی‌های مورداشاره، در اجرای یک سیستم مدیریت پایگاه داده، بررسی و تحلیل می‌شود. نتیجه این کار، دید وسیع و کاملی را برای محققین و طراحان مراکز داده ابری فراهم می‌نماید تا بهترین پیکربندی ممکن برای محیط عملیاتی‌شان را طبق نیاز خود انتخاب کنند. شکل ۱ معماری ماشین بومی، انواع ماشین مجازی و کانتینر را نشان می‌دهد. همین‌طور جدول ۴ تفاوت‌های اصلی کانتینر و ماشین مجازی را به‌اختصار نشان می‌دهد.

ادامه مقاله به‌صورت زیر سازمان‌دهی شده است: بخش ۲ مروری بر تحقیقات مرتبط درزمینه موضوع پژوهش ارائه می‌دهد. در بخش ۳، نرم‌افزارها و سخت‌افزارهای مورداستفاده، بنچمارک‌ها و همچنین روش‌های پیکربندی مورداستفاده در این مقاله بررسی می‌شود. بخش ۴ پیکربندی‌های مختلف را با یکدیگر مقایسه می‌کند و نتایج ارزیابی کارایی را ارائه می‌نماید. نهایتاً در بخش ۵، جمع‌بندی و پیشنهادهایی برای ادامه کار آمده است.

[1] Full Virtualization
[2] Hardware Virtual Machine
[3] Paravirtualization



شکل ۱- معماری ماشین بومی، کانتینر و ماشین‌های مجازی نوع ۱ و ۲

جدول ۴- تفاوت کانتینر و ماشین مجازی

معیار	ماشین مجازی	کانتینر
سیستم‌عامل	سیستم‌عامل خود را دارد	هسته را به اشتراک می‌گذارد
زمان بوت	در چند دقیقه	در چند ثانیه
کارایی	پایین‌تر	بالاتر
امنیت	بالا	متوسط
مجازی‌سازی	توسط سخت‌افزار	توسط سیستم‌عامل
حجم	زیاد	کم
بارِ کاری	سنگین‌وزن	سبک‌وزن

۲- پیشینه پژوهش

با مرور ادبیات موضوعی مشخص می‌شود که فناوری‌های مجازی‌سازی موضوعی بااهمیت و جذاب برای بسیاری از محققین بوده است. برای درک بهتر، بررسی ادبیات به سه دسته تقسیم می‌شود. دسته اول مطالعاتی را که در مورد ماشین‌های مجازیِ راه‌اندازی شده بر بستر هایپروایزرهای مختلف است، مورد تجزیه‌وتحلیل قرار می‌دهد. در دسته دوم، مطالعاتی که در مورد مجازی‌سازیِ مبتنی بر کانتینر است بررسی می‌شود. درنهایت، در دسته سوم مطالعاتی که مقایسه‌ای بین ماشین‌های مجازی و کانتینر انجام داده‌اند گنجانده می‌شود. تلاش شده است که مرور ادبیات به‌صورت مختصر و با تأکید و تمرکز بر مطالعات پژوهشیِ مرتبط با این مقاله باشد.

۲-۱- ماشین مجازی

در [۱۲] هایپروایزرهای Hyper-V و ESXi موردبررسی قرار گرفت. میزبان Hyper-V در بنچمارک‌های فایل سیستم که با سرور ویندوز ۲۰۱۶ به‌عنوان سیستم‌عامل مهمان اجرا می‌شود کمی بهتر عمل می‌کند و ESXi زمانی که سه ماشین مجازی هم‌زمان کار می‌کنند کمی‌کندتر عمل می‌کند. تفاوت بین ESXi و Hyper-V در اینجاست که ESXi بیش از ۵۰٪ بهتر عمل می‌کند، مخصوصاً زمانی که سیستم‌عامل مهمان Centos باشد. همان نویسندگان، در [۱۳] عملکرد هایپروایزر VirtualBox را موردمطالعه قرارگرفته است. نتایج مشابهی در خصوص ماشین‌های مجازی VirtualBox به دست آمد. در [۱۴] سه هایپروایزر رایج یعنی VMware ESXi5، Microsoft Hyper-V2008R2 و Citrix Xen Server 6.0.2 موردبررسی قرار گرفتند. نویسندگان نشان دادند که برای کارایی SQL، ماشین مجازی VMware ESXi5 عملکرد قابل‌توجه و مقبولی را در خصوص پردازنده و دیسک نشان داده است و به ترتیب نزولی Hyper-V2008R2 و Citrix Xen Server 6.0.2 در پله‌های بعدی قرار گرفتند. اما در شرایط بار کاری سبک، Citrix Xen 6.0.2 نسبت به Hyper-V2008R2 و VMware ESXi5 عملکرد بهتری را نشان دادند؛ در ادامه و با افزایش استفاده از منابع، Citrix Xen Server 6.0.2 از دور رقابت خارج شد. این به دلیل آن است که رسیدن مصرف دیسک به ۹۸٪ موجب کند شدن پاسخ سرور شده بود؛ اگرچه میزان استفاده از پردازنده آن بسیار پایین بود. هدف [۱۵]، ارزیابی عملکرد VMWare ESXi Server، XenServer و KVMدر محیط ابر خصوصی است. نتایج عملکرد به‌طور قانع‌کننده‌ای نشان داد که XenServer و ESXi Server هر دو تقریباً به یک اندازه خوب عمل می‌کنند و عملکرد آن‌ها تقریباً در همه آزمایش‌ها به عملکرد ماشین بومی نزدیک است بدون اینکه نشانه‌هایی از هرگونه سربار مجازی‌سازی پیدا باشد. در آزمون‌های پردازنده، ESXi از XenServer بهتر عمل نمود و در آزمون‌های حافظه اصلی و I/O، XenServer از ESXi عملکرد بهتری داشت. در [۱۶] چهار هایپروایزر معروف یعنی KVM، ESXi، Xen و Hyper-V از منظر فایل سیستم موردبررسی قرار گرفتند. نویسندگان به این نتیجه رسیدند که ماشین بومی در تمام معیارهای فایل سیستم، بسیار بهتر عمل نمود و این عمدتاً به این دلیل است که ماشین بومی از حافظه نهان استفاده می‌کند درحالی‌که KVM این کار را نمی‌کند. نتایج نشان داد که افت عملکرد KVM، برای یک ماشین مجازی بیش از ۵۰ درصد و درزمانی که بیش از ۲ و یا ۳ ماشین مجازی هم‌زمان کار کنند، بسیار قابل‌توجه است.

۲-۲- کانتینر

در [۱۷] مقایسه‌ای بین دو پلتفرم پرکاربرد کانتینری یعنی LXC و Docker ارائه شد. از دیدگاه این محققین و بر اساس بررسی‌های انجام‌شده، هر دو پلتفرم از منظر عملکرد تقریباً معادل هم بودند. اما تأخیر در زمان رفت‌وبرگشت درخواست مشتری Docker نسبت به LXC کمی بیشتر بود. بااین‌حال، آن‌ها می‌گویند که هر فناوری برای موارد مختلف، کاربرد خاص خود را دارد. محققان در [۱۸] به این نتیجه رسیدند که Podman اساساً برای استفاده در HPC[1] مناسب است؛ همچنین آن‌ها متوجه شدند که کانتینرها سربار عملکرد کمی تولید می‌کنند. به عقیده نویسندگان این مقاله، Podman تعدادی ویژگی سودمند ارائه می‌دهد که این‌ها شامل قابلیت کانتینرهای بدون نیاز به دسترسی سامانه‌ای برای HPC، ایجاد تصاویر[2] Podman فقط با امتیازات کاربر، پشتیبانی از پادها[3] و ایزوله سازی قوی‌تر است. در [۱۹] عملکرد به اشتراک‌گذاری داده با استفاده از Docker Volume و NFS[4] نشان داده‌شده است. آن‌ها به این نتیجه رسیدند که Docker پیکربندی ساده‌تر و پویاتری نسبت به NFS دارد. در حالیکه برای ایجاد یک Docker Volume تنها لازم است از پرچم‌ها[5] و دستورات مربوط به آن استفاده کنید، برای NFS، باید بسته‌های اضافی هم‌روی میزبان و هم‌روی کانتینرها نصب شود و یک فایل پیکربندی برای انجام نگاشت اشتراک‌گذاری ویرایش گردد. در مورد عملکرد، NFS ناهم‌زمان عملکرد بهتری را در تمام آزمایش‌های انجام‌شده برای عملیات نوشتن و خواندن روی دیسک‌های HDD[6] و SSD[7] نشان داد. بااین‌حال، غیرفعال کردن مکانیسم همگام‌سازی تأثیر بسیار مثبتی بر عملکرد دارد، اما خطر از دست رفتن اطلاعات در صورت بازنشانی سخت یا خرابی سیستم‌عامل را افزایش می‌دهد. در [۲۰] کانتینر Docker موردبررسی قرارگرفته است. این مقاله عملکرد Docker و اهمیت آن را در ماشین بومی برجسته می‌کند. اندازه‌گیری‌ها نشان داد که کانتینر بر روی ماشین بومی عملکرد خوبی را از خود نشان می‌دهد و با توجه به نتایج این پژوهش، برای به دست آوردن نتایج بهتر، چیدمان کانتینر Docker در ماشین بومی ضروری است.

[1] High Performance Computing
[2] Images
[3] Pods
[4] Network File System
[5] Flags
[6] Hard disk drive
[7] Solid State Drive

۳-۲- کانتینر و ماشین مجازی

در [۲۱] مطالعه‌ای تجربی عمیق بین ماشین مجازی و کانتینر Docker در محیط کلان داده Spark ارائه شد. نویسندگان به سه نتیجه جالب دست یافتند؛ اول، کانتینرهای Docker برای مدیران سیستم هم در مرحله استقرار و هم در مرحله راه‌اندازی راحت‌تر از ماشین‌های مجازی هستند. دوم، با حجم‌های کاری مختلف داده‌های بزرگ، کانتینر Docker مقیاس‌پذیری بسیار بهتری نسبت به ماشین‌های مجازی نشان می‌دهد. درنهایت با حجم کاری یکسان، کانتینر Docker به بهره‌وری پردازنده و دیسک بالاتری دست می‌یابد. در [۲۲] مجازی‌سازی سنگین‌وزن و سبک‌وزن با یکدیگر مقایسه شدند؛ نویسندگان به این نتیجه رسیدند که برخلاف ماشین‌های مجازی که محدودیت‌های منابع سفت و سختی دارند، کانتینرها محدودیت‌های نرمی را نیز مجاز می‌دارند که در سناریوهای تعهد بیش‌ازحد یا همان overcommitment مفید هستند، زیرا ممکن است از منابع کمتر استفاده‌شده تخصیص داده‌شده به سایر کانتینرها استفاده کنند. ایده اصلی پشت تعهد بیش‌ازحد این است که منابع بیشتری نسبت به سیستم موجود تخصیص دهیم، با این فرض که همه منابع به‌طور هم‌زمان توسط همه ماشین‌های مجازی یا کانتینرها استفاده نخواهند شد. نویسندگان دریافتند که استفاده Docker از فایل سیستم لایه‌ای کپی-روی-نوشتن و کنترل نسخه، موجب آسان‌تر شدن تحویل مداوم و یکپارچه‌سازی بهتر شده است. بنا بر مطالعه این نویسندگان، اگرچه ماشین مجازی‌های سبک مانند Clear Linux باهدف ارائه ویژگی‌های کانتینر محبوب هستند، اما با ویژگی‌های جداسازی ماشین‌های مجازی، به نظر می‌رسد رویکردهای مجازی‌سازیِ ترکیبی یک‌راه امیدوارکننده باشد. در [۲۳] کارایی دو رویکرد مبتنی بر کانتینر و مبتنی بر هایپروایزور در محیط اینترنت اشیاء و دستگاه‌های لبه شبکه با یکدیگر مقایسه شد. برای این منظور، از بنچمارک‌های مصنوعی بر روی یک پلتفرم Cubieboard2 SoC استفاده شد. نتایج روی راه‌حل مبتنی بر هایپروایزور، سربار قابل‌توجهی را نشان می‌داد که به‌راحتی قابل کاهش نبود؛ اما نتایج برای پلتفرم Docker امیدوارکننده بود و سربار تا حد زیادی کاهش داده شد.

درنهایت، مهم‌ترین و مرتبط‌ترین تحقیقات و مطالعات بررسی‌شده در این مقاله، از منظر نوع هایپروایزرها و کانتینرهای موردبررسی و منابع سخت‌افزاری که کارایی آن‌ها مورد ارزیابی قرارگرفته‌اند، در جدول ۵ خلاصه‌شده‌اند. همان‌طور که در این جدول ملاحظه می‌شود، مطالعه حاضر نسبت به تحقیقات دیگر، دیدگاه مقایسه‌ای بهتر و منصفانه‌تری را نسبت به کارایی پلتفرم‌های مطرح کانتینری (با توجه به مقایسه هم‌زمان هر سه پلتفرم مطرح و لحاظ نمودن پلتفرم نوظهور Podman) و مقایسه آن با کارایی ماشین‌های مجازی مبتنی بر دو هایپروایزر متداول اما کمتر بررسی‌شده (Xen و KVM) و با بررسی هم‌زمان عملکرد چهار منبع سخت‌افزاری مهم و رایج به دست می‌دهد.

جدول ۵- مقایسه سایر مطالعات مرتبط با این پژوهش

شماره	عنوان	هایپروایزر	پلتفرم کانتینر	منابع			
				پردازنده	حافظه اصلی	دیسک	شبکه
۱	VMware ESXi and Microsoft Hyper-V Hypervisor Performance Comparison [12]	ESXi, Hyper-V	-			✓	
۲	Performance comparison of native host and hyper-based virtualization VirtualBox [13]	Virtual Box	-			✓	
۳	Performance evaluation and comparison of the top market virtualization hypervisors [14]	ESXi, Xen, Hyper-V	-			✓	
۴	Performance Comparison of Hypervisors in the Private Cloud [15]	KVM, Xen, ESXi	-	✓	✓	✓	✓
۵	Performance comparison of native host and hyper-based KVM virtualization [16]	KVM	-			✓	
۶	Comparison of LXC and Docker Technologies [17]	-	Docker, LXC			✓	
۷	Rootless Containers with Podman for HPC [18]	-	Podman	✓	✓		
۸	Performance Evaluation of Container's Shared Volumes [19]	-	Docker		✓		
۹	Analysis of Docker Performance in Cloud Environment [20]	-	Docker				
۱۰	A Comparative Study of Containers and Virtual Machines in Big Data Environment [21]	KVM	Docker	✓	✓	✓	✓
۱۱	Containers and Virtual Machines at Scale: A Comparative Study [22]	KVM	Docker	✓	✓	✓	✓
۱۲	Virtualization at the Network Edge: A Performance Comparison [23]	KVM	Docker	✓	✓	✓	✓
۱۳	مطالعه جاری	KVM, Xen	Docker, LXC, Podman	✓	✓	✓	✓

۳- روش تحقیق

این بخش، به بررسی اطلاعات کلی اعم از نرم‌افزارها و سخت‌افزارها، بنچمارک‌ها،

حالت‌های پیکربندی و روش اجرای آن‌ها می‌پردازد.

۳-۱- پیکربندی نرم‌افزار و سخت‌افزار

محیط راه‌اندازی و همه اندازه‌گیری‌های این پژوهش بر روی ماشینی با پردازنده Intel® Core™ i7-7700 @ 4.2 GHz، مادربرد ASUS PRIME H270-PLUS، با تراشه Intel® Xeon® E3-1200، هشت گیگابایت حافظه اصلی، دیسک 1000GB Western Digital WD10EZEX-75W و کارت شبکه Realtek RTL8111/8186/8411 انجام‌شده است. سیستم‌عامل مورداستفاده، اوبونتو سرور نسخه ۲۰.۰۴ بوده است.

سرور Xen به‌صورت نوع یک و با استفاده از XCP-ng نسخه ۸.۲.۱ راه‌اندازی و اجرا می‌شود. XCP-ng یک توزیع لینوکس از پروژه Xen است که بر روی هایپروایزر Xen و پروژه Xen API (XAPI) تکیه دارد. این پروژه در سال ۲۰۱۸ و به دنبال پروژه منبع باز XenServerدر حال حاضر (Citrix Hypervisor)معرفی شد. هدف اصلی XCP-ng ارائه یک‌راه آسان برای استقرار یک زیرساخت مبتنی بر ماشین مجازی با استفاده از Xen به‌عنوان هایپروایزر است. همچنین در این مطالعه، برنامه Xen Orchestra بر روی Docker اجرا و به کمک آن ماشین‌های مجازی روی سرور Xen مدیریت می‌شود. Xen Orchestra یک برنامه مبتنی بر وب است که برای مدیریت XenServer، پشتیبان گیری و آغازگر ابر طراحی‌شده است. این نرم‌افزار یک رابط واحد فراهم می‌کند که در بسیاری از دستگاه‌ها برای کنترل مخازن[1]، میزبان‌ها و ماشین‌های مجازی کار می‌کند و این امکان را می‌دهد که زیرساخت XenServer با حداکثر کارایی ایجاد، اصلاح و مدیریت شود. همچنین سرور Xen با فن ترکیبی PVHVM -یعنی ترکیب مجازی‌سازی سخت‌افزاری به همراه درایورهای کارت شبکه و دیسک در حالت پارامجازی‌سازی- اجرا می‌شود.

هایپروایزر دومی که در پژوهش حاضر بررسی‌شده است، KVM نسخه ۶.۲ است که به‌صورت نوع دو و در حالت HVM اجرا می‌شود. همچنین از حالت Bridge برای شبکه ماشین مجازی KVM استفاده می‌شود. هنگامی‌که یک Bridge لینوکس در KVM استفاده می‌شود، به ماشین مجازی اجازه می‌دهد تا به شبکه‌ها و سرویس‌های خارجیِ بیرون از محیط مجازی دسترسی داشته باشد. Bridged Networking یک کارت شبکه اختصاصی برای یک ماشین مجازی است که مسیر ارتباطی را به شبکه‌های بیرونی فراهم می‌کند.

برای ساخت کانتینرها از Dockerنسخه ۲۰.۱۰.۱۸، Podman نسخه ۳.۴.۴ و LXC نسخه ۵.۵ استفاده می‌شود. همچنین برای یکنواخت شدن شیوه اندازه‌گیری، به‌تمامی ماشین‌های مجازی، حداکثر منابع قابل‌استفاده اختصاص داده می‌شود.

۳-۲- بنچ‌مارک‌ها

در پژوهش حاضر، کارایی چهار منبع سخت‌افزاری اصلی یعنی پردازنده، حافظه اصلی، دیسک و شبکه موردبررسی قرارگرفته است. معمولاً برای ارزیابی و تحلیل کارایی، از بسته‌های نرم‌افزاری شناخته‌شده‌ای که تحت عنوان بنچ‌مارک توسعه‌یافته‌اند، استفاده می‌گردد. اگرچه بنچ‌مارک‌های زیادی در طی سالیان گذشته برای محک عملکرد برنامه‌ها پیاده‌سازی شده‌اند، در پژوهش حاضر از متداول‌ترین آن‌ها شامل بنچ‌مارک‌های Compress-7zip [۲۴] نسخه ۱.۱۰.۰ برای پردازنده، STREAM [۲۵] نسخه ۱.۳.۳ برای حافظه، IOzone [۲۶] نسخه ۱.۹.۶ برای دیسک و Netperf [۲۷] نسخه ۲.۷.۰ برای شبکه استفاده می‌شود. Compress-7zip یک برنامه آزمون فشرده‌سازی و خارج کردن از فشردگی است که کارایی پردازنده را اندازه‌گیری می‌کند. روش کار به این صورت است که فایلی با حجم ۱۲۸ مگابایت با سایز دیکشنری ۳۲ مگابایت را فشرده‌سازی می‌کند و سپس از حالت فشرده خارج می‌کند. این بنچ‌مارک درنهایت تعداد *میلیون دستورالعمل در ثانیه*[2] (MIPS)را برای فشرده‌سازی و از حالت فشرده خارج کردن برمی‌گرداند. طبیعتاً هرچه عدد MIPS بیشتر باشد، کارایی نیز بهتر است.

STREAM یک برنامه بنچ‌مارک ساده و مصنوعی است که پهنای باند حالت پایدار حافظه اصلی را در واحد *مگابایت بر ثانیه* اندازه‌گیری می‌کند. قانون کلی برای اجرای STREAM این است که هر آرایه باید حداقل ۴ برابر اندازه مجموع تمام حافظه‌های نهان سطح آخرِ استفاده‌شده در اجرا یا یک‌میلیون عنصر (هرکدام که بزرگ‌تر است) باشد. STREAM از چهار نوع عملیات برای تجزیه‌وتحلیل استفاده می‌کند:

- "Copy" نرخ انتقال را در صورت عدم وجود محاسبات اندازه‌گیری می‌کند.
- "Scale" عملیات ساده حسابی اضافه می‌کند.
- "Add" عملوند سوم را اضافه می‌کند تا امکان آزمایش چندین پورت بارگذاری/ذخیره در ماشین‌های برداری را فراهم کند.
- "Triad" اجازه عملیات زنجیره‌ای/همپوشانی/تجمیع ضرب/افزودن را می‌دهد.

مجموعه عملیات STREAM در جدول شماره ۶ نشان داده‌شده است.

جدول ۶- عملیات استریم

نوع عملیات	هسته
کپی	a(i)=b(i)
مقیاس	a(i)=q*b(i)
اضافه	a(i)=b(i)+c(i)
سه‌گانه	a(i)=b(i)+q*c(i)

بسته نرم‌افزاری IOzone یک ابزار محک فایل‌سیستم است. این بنچ‌مارک، انواع عملیات فایل را تولید و کارایی دیسک را اندازه‌گیری می‌نماید. IOzoneعملکرد ورودی/خروجی فایل را برای این عملیات آزمایش می‌کند. برای این منظور، خواندن و نوشتن بر روی دیسک به کمک IOzone بررسی‌شده است. برای این آزمایش، یک فایل ۵۱۲ مگابایتی و رکورد با سایز دو مگابایت موردبررسی قرارگرفته است.

نهایتاً Netperf یک ابزار بنچ‌مارک شبکه است که کارایی شبکه را از جنبه‌های مختلف محک می‌زند و پهنای باند شبکه را بین دو میزبان به دست می‌آورد. همچنین برای به دست آوردن تأخیر بین دو میزبان‌هم کاربرد دارد. Netperf آزمون‌های مختلفی را ارائه می‌دهد که در این کار از TCP_STREAM برای اندازه‌گیری پهنای باند پروتکل TCP استفاده می‌شود. روش کار به این صورت است که مقداری داده از سیستم در حال اجرای Netperf به سیستم در حال اجرای Netserver انتقال داده می‌شود. همچنین آزمون TCP_RR و UDP_RR نیز در این مطالعه انجام می‌شود. یک آزمون TCP_RR را می‌توان به‌عنوان پینگ فضای کاربر در نظر گرفت. نرخ تراکنش، تعداد تراکنش‌های کامل مبادله شده، تقسیم‌بر مدت‌زمانی است که برای انجام آن تراکنش‌ها صرف شده است. آزمون UPD_RR بسیار شبیه به آزمون TCP_RR است به‌جز اینکه از UDP به‌جای TCP استفاده می‌شود. UDP امکان ارسال مجدد دیتاگرام‌های UDP ازدست‌رفته را فراهم نمی‌کند و Netperf نیز چیزی برای آن اضافه نمی‌کند؛ به این معنی که اگر هر درخواست یا پاسخی از بین برود، تبادل درخواست‌ها و پاسخ‌ها از آن نقطه تا پایان زمان‌سنج آزمون متوقف می‌شود. Netperf نمی‌داند که این اتفاق افتاده است یا خیر؛ تنها علامت آن نرخ پایین تراکنش در ثانیه خواهد بود.

هرکدام از بنچ‌مارک‌ها ۵ بار انجام شد و سپس انحراف آن بررسی و نتیجه آن طبق میانگین نوشته شد. اگر انحراف یا اختلاف در نتایج زیاد بود، آزمایش تا ۱۵ بار انجام می‌شد تا به حد ثابتی برسد و سپس میانگین آن در نظر گرفته می‌شد. افزایش

[1] Repository
[2] Million Instructions Per Second

تعداد تکرارها به جمع‌آوری نقاط داده بیشتر و کاهش تأثیر تغییرات پرت یا تصادفی کمک می‌کند.

Sysbench یک ابزارِ بنچمارکِ منبع‌باز و چندمنظوره است که ارزیابی عملکرد را برای منابع و نرم‌افزارهای شناخته‌شده همانند پردازنده، رم، دیسک و پایگاه داده MySQL انجام می‌دهد. یکی از مزایای مهم Sysbench این است که ویژگی‌هایِ معیار چند رشته‌ای را برای لینوکس ارائه می‌دهد. در این کار از Sysbench همراه با آزمون OLTP برایِ بررسیِ عملکردِ MySQL استفاده می‌شود. OLTP یا پردازش تراکنش آنلاین، تغییر، درج، حذف یا پرس‌وجو از داده‌ها در پایگاه داده را انجام می‌دهد که مهم‌ترین خروجیِ آن تعداد عملیات در ثانیه (transaction per second) است. این آزمون، جستجوهایِ کلید اصلی را آزمایش می‌کند. برای این کار، تعداد رشته‌ها تا صد رشته افزایش می‌یابد و در هر مرحله تعداد عملیات بررسی می‌شود. روشِ کار به این صورت است که در ابتدا یک کاربر ایجاد می‌شود، سپس تمامِ دسترسی‌ها به آن اعطا می‌شود. در مرحله بعد بیست جدول با سایزِ ده هزار ساخته می‌شود. درنهایت، در هر مرحله، کارایی آن با رشته‌های مختلف که از یک تا صد رشته است مورد ارزیابی و بررسی قرار می‌گیرد.

۳-۳- پیکربندی‌های مورد ارزیابی

برای انجام این کار، شش پیکربندی مختلف به‌منظور بررسی عملکرد محاسباتی، مورداستفاده قرارگرفته است. این پیکربندی‌ها عبارت‌اند از:

۱- ماشین بومی (Native)
۲- ماشین‌مجازی مبتنی بر هایپروایزر KVM
۳- ماشین‌مجازی مبتنی بر هایپروایزر Xen
۴- کانتینر مبتنی بر Docker
۵- کانتینر مبتنی بر Podman
۶- کانتینر مبتنی بر LXC

این پیکربندی‌ها، به دلیل پتانسیل آن‌ها در بهبود مقیاس‌پذیری نرم‌افزار، ارائه محیط‌های محاسباتی ایزوله برای کاربران ابری و مدیریت زمان‌بندی برنامه‌ها انتخاب شدند. ابتدا، کارایی استفاده از منابع (پردازنده، حافظه اصلی، دیسک و شبکه) بر روی ماشین بومی به‌عنوان یک پلتفرم پایه مورد ارزیابی قرار گرفت. آنگاه، این ارزیابی برای سایر پیکربندی‌های دیگر نیز انجام شد. باهدف انجام ارزیابی منصفانه و در شرایط برابر، هر یک از پیکربندی‌ها به‌صورت ایزوله مورد ارزیابی قرارگرفته‌اند. همچنین، در همه سناریوهای ۲ تا ۶، تنها یک ماشین مجازی بر روی یکی از هایپروایزرهای هدف یا یک کانتینر بر روی یکی از پلتفرم‌های ساخت کانتینرِ هدف پیکربندی و اجراشده است.

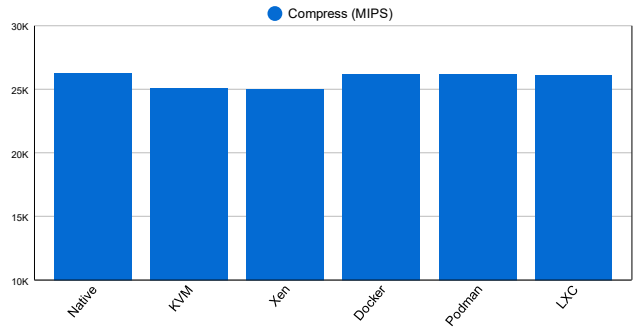
۴- نتایج

در این بخش کارایی چهار منبع اصلی، یعنی پردازنده، رم، دیسک و شبکه اندازه‌گیری و تمامی نتایج بررسی و تحلیل می‌شوند؛ درنهایت در قالب یک جدول، بهترین پیکربندی برای هر منبع نیز نشان داده می‌شود.

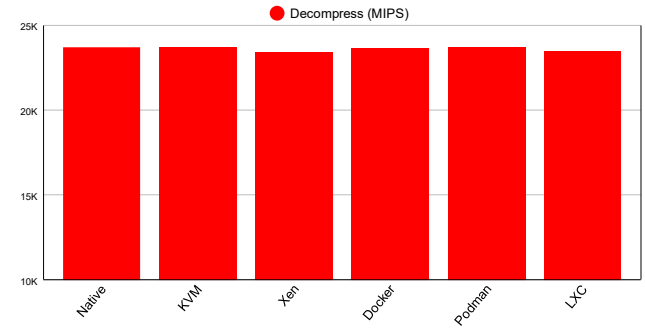
۴-۱- پردازنده

در این ارزیابی عملکرد پردازنده موردبررسی قرار گرفت. بنچمارک Compress-7zip در ماشین بومی، ماشین مجازی و کانتینر اجرا شد. شکل ۲ و ۳ عملکرد اجرای این بنچ مارک به‌منظور فشرده‌سازی و خارج کردن از فشردگی را نشان می‌دهند. بعدازانجام آزمایش مشاهده شد که در فرآیند این اندازه‌گیری، برایِ عملیات فشرده‌سازی، ماشینِ بومی بالاترین کارایی را دارد. بعدازآن، با اختلاف کمی به ترتیب کانتینرهایِ Podman، Docker و سپس LXC قرار گرفتند. در عملیات خارج کردن از فشردگی، همانند فشرده‌سازی، ماشین بومی بهترین کارایی را دارد و بعدازآن تقریباً تمامی روش‌های پیاده‌سازی با اختلافی ناچیز در یک سطح قرار می‌گیرند.

در حالت کلی، یعنی در نظر گرفتن هم‌زمان فشرده‌سازی و سپس خارج کردن از فشردگی، ماشین بومی بالاترین جایگاه را به دست آورد. بعدازآن به ترتیب کانتینرهای Podman، Docker و LXC قرار گرفتند. در کل، کانتینرها با اختلاف ناچیزی بهتر از ماشین‌های مجازی رفتار نمودند. این کارایی بهتر، به خاطر ترجمه نکردن دستورات سامانه‌ای، ایزوله نبودن کامل، اجرای مستقیم کاربرد بر روی کانتینر و سربار پایین‌تر آن‌ها است. همچنین، ماشین مجازی Xen ضعیف‌ترین کارایی را در خصوص عملکرد پردازنده از خود نشان داد.



شکل ۲- کارایی فشرده‌سازی

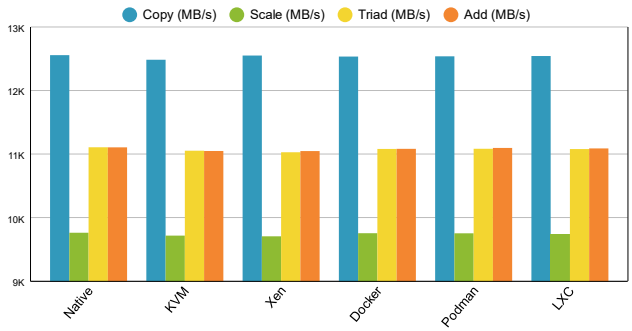


شکل ۳- کارایی خارج کردن از فشردگی

۴-۲- حافظه اصلی

عملکرد حافظه توسط بنچمارک STREAM همراه با چهار عملیات Copy، Scale، Triad و Add اندازه‌گیری شد. شکل ۴، مقایسه عملکرد حافظه اصلی در پیکربندی‌های مختلف را نشان می‌دهد. در نگاه اول می‌توان گفت که تمامی روش‌های پیکربندی، با توجه به وجود لایه‌های مجازی‌سازی مختص خود، کارایی بسیار نزدیکی با یکدیگر دارند. اما برای توضیح جزئی‌تر باید گفت، همانند کارایی پردازنده، ماشین بومی بالاترین کارایی حافظه را در بین تمامی روش‌های پیکربندی نشان داد. کانتینر Podman و بعدازآن به ترتیب کانتینر Docker و LXC با اختلاف ناچیزی در مراحل بعدی قرار گرفتند.

در بعضی حالات، برای عملیات مختلف، کانتینر کارایی برابر یا بهتری نسبت به ماشین مجازی دارد که می‌توان به دلیل سربار کمتر، اشتراک هسته و منابع و بهینه‌سازی کانتینر باشد. همانند بررسی کارایی پردازنده، ماشین مجازی Xen و KVM کمترین کارایی را برای حافظه اصلی نسبت به سایر حالات پیکربندی دارند. دلیل آن‌هم سربار بیشتر ماشین‌های مجازی نسبت به کانتینر به دلیل معماری پیچیده‌تر و اشتراک‌گذاری منابع سیستم بین ماشین‌های مجازی است.



شکل ۴- کارایی حافظه اصلی

۴-۳- دیسک

ارزیابی عملکرد دیسک توسط بنچمارک IOzone صورت پذیرفت. طبق شکل‌های ۵ و ۶ برای خواندن از دیسک، برخلاف منابع قبلی، ماشین بومی در صدر قرار نداشت؛ بلکه ماشین مجازی Xen بیشترین کارایی را داشت. سپس با اختلاف، ماشین بومی قرار گرفت. درنهایت، بقیه حالات تقریباً در یک سطح قرار گرفتند. این‌طور مشاهده شد که ماشین مجازی Xen در تمامی روش‌های پیکربندی بیشترین کارایی را برای خواندن از دیسک دارد که قاعدتاً یکی از دلایل اصلی آن استفاده از فن PVHVM بوده است. در فن PVHVM که در بخش ۲ موردبررسی قرار گرفت، دیسک، برخلاف پردازنده و حافظه از درایورهای پارامجازی‌سازی استفاده می‌کند. این قابلیت باعث می‌شود تا کارایی و سرعت بسیار بهتری نسبت به ماشین بومی و ماشین مجازی KVM و کانتینرها داشته باشد.

برای نوشتن بر روی دیسک، همانند خواندن از دیسک، ماشین مجازی Xen بالاترین کارایی را داشت. در مرحله بعد با اختلاف نسبتاً زیادی، ماشین بومی قرار گرفت. سپس، با اختلاف نسبتاً زیادی، تمامی حالات تقریباً در یک سطح قرار گرفتند؛ برخلاف حالت خواندن از دیسک، کانتینر Podman نتوانست کارایی مطلوبی ارائه نماید و در رقابت با کانتینرهای دیگر شکست خورد و کمترین کارایی را در کل حالات داشت. در کل می‌توان گفت ماشین مجازی Xen بازهم توانست کارایی خوبی را از خود نشان دهد که دلیل آن می‌تواند چند مورد باشد:

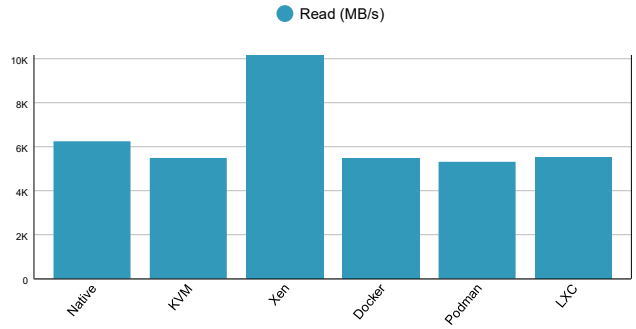
۱. پارا مجازی‌سازی ارتباطات مهمان-هایپروایزر را بهینه می‌کند، سربار اضافی را کاهش می‌دهد و عملکرد را بهبود می‌بخشد.

۲. Xen تخصیص دقیق پردازنده، حافظه اصلی و منابع دیسک را به مهمانان امکان‌پذیر می‌کند، اختلاف را به حداقل می‌رساند و عملکرد خواندن و نوشتن را افزایش می‌دهد.

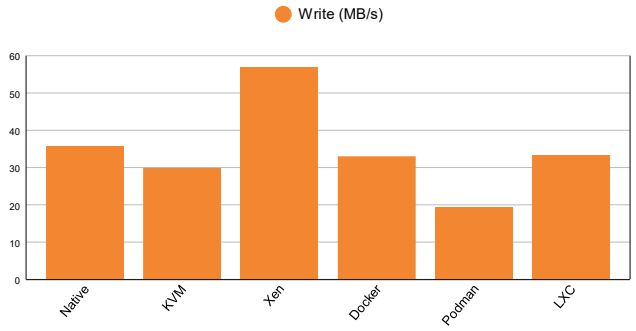
۳. Xen با فن PVHVM به مهمانان امکان دسترسی مستقیم به منابع سخت‌افزاری را می‌دهد و از تنگناهای احتمالی در میزبان یا کانتینرها برای خواندن یا نوشتن سریع‌تر عبور می‌کند.

۴-۴- شبکه

در این خصوص، به‌منظور ارزیابی عملکرد شبکه بنچمارک Netperf مورداستفاده قرار گرفت. به کمک این بنچمارک آزمایش‌های TCP-Stream، TCP-Request Response و UDP-Request Response صورت پذیرفت و کارایی عملکرد شبکه از جهات مختلف اندازه‌گیری و بررسی شد. شکل ۷ مقدار گذردهی پهنای باند پروتکل TCP را نشان می‌دهد. طی اندازه‌گیری کارایی شبکه تحت آزمایش TCP_Stream، تقریباً تمامی حالات به‌جز حالت مربوط به کانتینر Podman در یک سطح قرار گرفتند. درواقع کانتینر Podman کمترین کارایی را در بین رقبای خود داشت. مشخص‌شده که این عملکرد به این دلیل است که کاربران غیرمجاز نمی‌توانند رابط‌های شبکه را روی میزبان ایجاد کنند؛ بنابراین، برای کانتینرهای بدون نیاز به دسترسی سامانه‌ای، حالت شبکه پیش‌فرض slirp4netns است. به دلیل مجوزهای محدود، slirp4netns فاقد برخی از ویژگی‌های شبکه در مقایسه با عملکرد شبکه Podman در محیط اجرا با دسترسی سامانه‌ای است.



شکل ۵- کارایی خواندن از دیسک

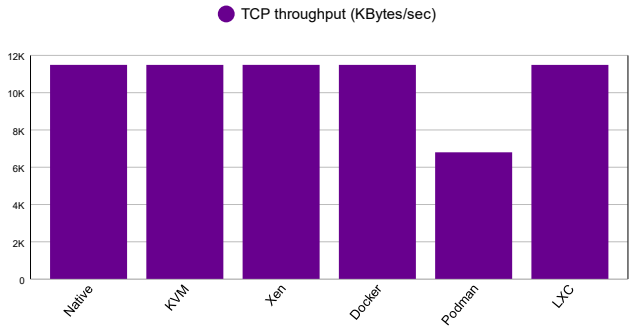


شکل ۶- کارایی نوشتن بر روی دیسک

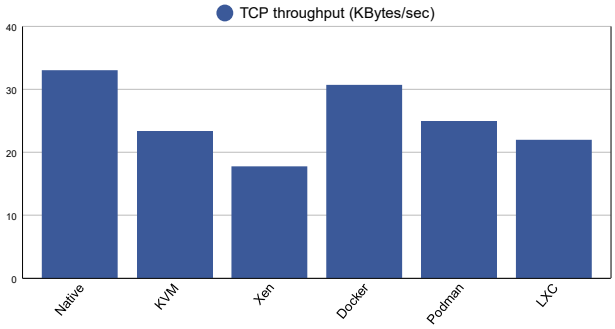
همچنین شکل‌های ۸ و ۹ مقدار گذردهی دریافت پاسخ را برای پروتکل UDP و TCP نشان می‌دهد. برای آزمایش دریافت پاسخ، برای پروتکل‌های TCP و UDP، به ترتیب ماشین بومی و کانتینر Docker در صدر قرار گرفتند. کانتینر Podman در رده بعدی واقع‌شده است، کانتینر LXC و ماشین مجازی KVM، تقریباً در یک سطح قرار دارند. اما نکته جالب، ماشین مجازی Xenبود که نتوانست عملکرد خوبی را از خود نشان دهد و ضعیف‌ترین پیکربندی در کل حالات بود.

۴-۵- سیستم مدیریت پایگاه داده MySQL

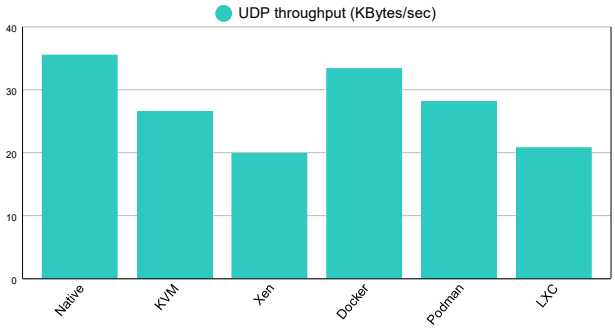
هدف از این بخش، پیکربندی، اجرا و سپس ارزیابی کارایی یک نرم‌افزار مدیریت پایگاه داده تحت پیکربندی‌های مختلف است. البته هرکدام از پیکربندی‌ها روشِ اجرای مختص خود رادارند. به‌عنوان‌مثال Docker، کانتینر نرم‌افزار مربوطه را مستقیماً اجرا می‌کند؛ ولی LXC باید ابتدا سیستم‌عاملی را فراخوانی و اجرا نماید و سپس بر روی آن نرم‌افزار مدیریت پایگاه داده نصب و اجرا می‌شود. بدین منظور، از نرم‌افزار منبع باز MySQL استفاده‌شده است. شکل ۱۰ نتایج ارزیابی را نشان می‌دهد. با توجه به این‌که عملکرد نرم‌افزار MySQL رابطه مستقیمی با دیسک دارد، همانند ماشینِ مجازیِ Xen که کارایی بهتری را در بخش I/O داشت، در اینجا هم Xen، به‌عنوان ماشینِ مجازی کارایی خوبی را از خود نشان داد. در ابتدا، برایِ یک‌رشته، ماشینِ مجازیِ Xen با اختلاف زیاد بهترین کارایی را داشت و به دنبالِ آن با اختلاف زیادی بقیه حالات تقریباً کارایی مشابهی داشتند. ماشین بومی و سپس کانتینرها به ترتیب LXC، Docker و سپس Podman کارایی بهتری را در بینِ پیکربندی‌ها داشتند؛ درنهایت، KVM در آخرین جایگاه قرار گرفت.



شکل ۷- کارایی گذردهی شبکه (TCP-Stream)



شکل ۸- کارایی گذردهی شبکه (درخواست و پاسخ TCP)

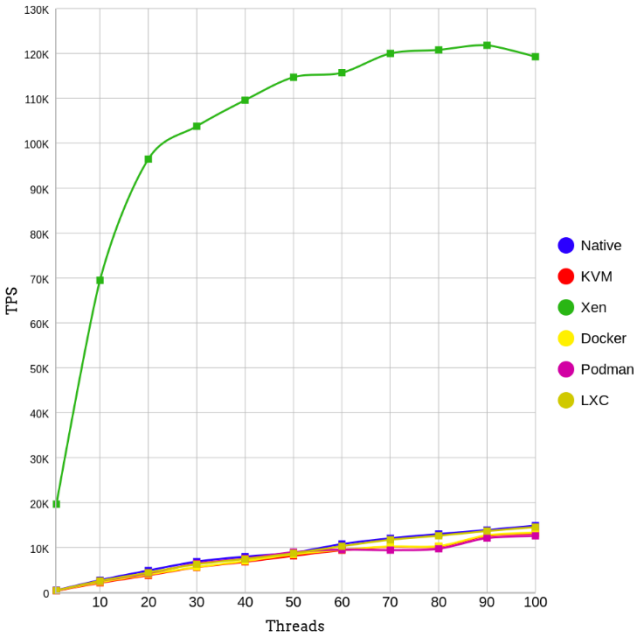


شکل ۹- کارایی گذردهی شبکه (درخواست و پاسخ UDP)

با افزایشِ تعداد رشته‌های موازی، شاهدِ افزایشِ کارایی ناگهانی برای Xen بودیم؛ درصورتی‌که بقیه حالات با افزایش رشته‌ها، رشد کمی را در تعداد تراکنش‌ها در حالت کلی تجربه کردند. اما نکته جالب در مورد پیکربندی Xen اینجاست که با افزایش رشته تا پنجاه عدد، تعداد تراکنش بر ثانیه به‌سرعت افزایش می‌یابد؛ ولی در ادامه و با افزایش رشته‌ها از پنجاه به صد رشته، افزایش تراکنش چشم‌گیری وجود ندارد.

۶-۴- بحث

جدول ۷، جمع‌بندی کلی ارزیابی عملکرد سناریوهای مختلف را نشان می‌دهد. همان‌طور که مشاهده می‌شود، برای منابع پردازنده، حافظه اصلی و شبکه، ماشین بومی بهترین عملکرد را از خود نشان داد و به دنبال آن کانتینرها قرار گرفتند؛ البته که این رقابت بسیار نزدیک و تنگاتنگ بود. اما برای دیسک، این ماشین‌مجازی Xen بود که بهترین عملکرد را ارائه نمود و بعدازآن ماشین بومی و کانتینر LXC بهترین عملکرد را داشتند.



شکل ۱۰- کارایی پایگاه داده MySQL

با توجه به نتایج و رتبه‌بندی به‌دست‌آمده می‌توان گفت برخی از سرویس‌ها یا برنامه‌ها به‌شدت به دیسک وابسته‌اند و به همین دلیل استفاده از Xen چه به‌صورت مجزا و چه لایه زیرین می‌تواند مفید باشد. این برنامه‌ها عبارت‌اند:

۱. سرورهای پایگاه داده: پایگاه‌های داده MySQL، PostgreSQL و Oracle اغلب به عملیات خواندن و نوشتن دیسک مکرر نیاز دارند. عملکرد ورودی/خروجی دیسک بهبودیافته Xen عملکرد کلی پایگاه داده را افزایش می‌دهد.

۲. سرورهای فایل: خدماتی مانند اشتراک فایل و ذخیره‌سازی متصل به شبکه، انتقال فایل‌های بزرگ را انجام می‌دهند. ورودی/خروجی دیسک کارآمد Xen عملکرد سرور فایل را بهبود می‌بخشد، عملیات خواندن/نوشتن سریع‌تر و دسترسی بهتر به داده‌ها را تضمین می‌کند.

۳. شبکه‌های تحویل محتوا: این شبکه‌ها محتوای وب را در مکان‌های مختلف، با تکیه‌بر ورودی/خروجی دیسک برای ذخیره‌سازی، بازیابی و توزیع، ارائه می‌دهند. ورودی/خروجی دیسک پیشرفته Xen تحویل محتوا را تسریع می‌کند و عملکرد شبکه را بهبود می‌بخشد.

۴. سامانه‌های ثبت و تجزیه‌وتحلیل داده‌ها: برنامه‌هایی که حجم زیادی از گزارش‌ها تولید می‌کنند یا مجموعه‌های داده گسترده را پردازش می‌کنند از ورودی/خروجی دیسک کارآمد بهره می‌برند. این شامل ابزارهای تجزیه‌وتحلیل گزارش، چارچوب‌های کلان داده مانند Hadoop و Spark و پلتفرم‌های جریان داده در زمان واقعی است. ورودی/خروجی دیسک برتر Xen پردازش و تجزیه‌وتحلیل سریع‌تر داده را امکان‌پذیر می‌کند.

۵. زیرساخت دسکتاپ مجازی: این محیط‌ها چندین نمونه دسکتاپ مجازی را روی یک سرور متمرکز اجرا می‌کنند. ورودی/خروجی دیسک برای ذخیره پروفایل‌های کاربر و تصاویر ماشین مجازی بسیار مهم است. عملکرد دیسک بهینه Xen تجربه کاربری روان و مدیریت کارآمد دسکتاپ را تضمین می‌کند.

همچنین برای برخی از سرویس‌ها و برنامه‌هایی که بیشتر به کارایی شبکه وابسته‌اند Docker می‌تواند پیکربندی مناسبی باشد. این سرویس‌ها عبارت‌اند از:

۱. وب سرورها که اتصالات شبکه سریع زمان پاسخگویی سریع و بهبود تجربه کاربر را تضمین می‌کند.

۲. خدمات پخش ویدئو که عملکرد بالای شبکه امکان پخش یکپارچه و به حداقل رساندن بافر را فراهم می‌کند.

۳. پلتفرم‌های رایانش ابری که اتصال شبکه قابل‌اعتماد، انتقال کارآمد داده و ارتباط روان بین منابع ابری را تسهیل می‌کند.

۴. شبکه‌های تحویل محتوا که عملکرد بهینه شبکه، تحویل سریع محتوا و تأخیر کم را برای کاربران نهایی تضمین می‌کند.

۵. سرورهای بازی آنلاین که اتصالات سریع و پایدار شبکه تأخیر را به حداقل می‌رساند و یک تجربه بازی چند نفره روان را ارائه می‌دهد.

۶. تجارت با فرکانس بالا که اتصالات شبکه با تأخیر بسیار کم برای اجرای سریع تجارت و کاهش خطرات بسیار مهم هستند.

جدول ۷- جدول رتبه‌بندی عملکرد

رتبه	پردازنده	حافظه	دیسک	شبکه
۱	Native	Native	Xen	Native
۲	Podman	Podman	Native	Docker
۳	Docker	LXC	LXC	KVM

۵- جمع‌بندی و کارهای آینده

در این تحقیق، عملکرد دو فناوری مجازی‌سازی محبوب و پرکاربرد (ماشین‌مجازی و کانتینر) ازلحاظ میزان استفاده از چهار منبع اصلی (پردازنده، حافظه اصلی، دیسک و شبکه) به‌طور مجزا مورد ارزیابی قرار گرفت؛ اندازه‌گیری میزان مصرف منابع بر روی ماشین بومی و بدون استفاده از مجازی‌سازی نیز به‌صورت جداگانه صورت پذیرفت و با آزمایش‌های متعدد در شرایط مختلف ارزیابی شد. بررسی‌ها در محیط واقعی نشان داد که کانتینر سربار کمتری دارد و در اکثر حالات کارایی قابل قبولی از خود نشان می‌دهد. به‌گونه‌ای که در مورد عملکرد پردازنده و حافظه اصلی، کانتینر Podman و در مورد عملکرد شبکه، کانتینر Docker بهترین کارایی را داشتند. البته در مورد ماشین‌مجازی Xen استثنائا کارایی بسیار خوبی برای دیسک مشاهده شد که به دلیل استفاده از حالت PVHVM این ناظر ماشین مجازی بوده است.

تحلیل‌های انجام‌شده در این مطالعه به ارائه‌دهندگان سرویس‌های مبتنی بر مجازی‌سازی و ازجمله طراحان مراکز داده ابری نشان می‌دهد که چه پیکربندی‌ای برای چه شرایط، محیط و کاربردی مناسب‌ترین کارایی را به دنبال خواهد داشت؛ درنتیجه، محققین می‌توانند به درک بهتری از نقاط قوت و ضعف پیکربندی‌های مختلف دست یابند و فراهم‌کنندگان این سرویس‌ها قادر خواهند بود که مطابق نیاز خود، پیکربندی مناسب را انتخاب نمایند.

یافته‌های این مطالعه به‌عنوان پایه‌ای برای کارهای آینده خواهد بود. یک موضوع جالب تحقیقاتی برای آینده، بررسی و تحلیل اجرای ترکیبی کانتینرها و ماشین‌های مجازی است؛ بدین ترتیب که کانتینرهای مبتنی بر فناوری‌های مختلف بر روی ماشین‌های مجازی مبتنی بر هایپروایزرهای مختلف اجرا و ارزیابی شوند. به نظر می‌رسد که با این کار از مزایای هر دو فناوری مجازی‌سازی به‌صورت هم‌زمان می‌توان بهره برد. همچنین، بررسی ماشین‌های مجازی سبک‌وزن به‌عنوان جایگزینی برای کانتینرها در مطالعات آتی موردنظر خواهد بود.

۶- مآخذ

[1] T. Combe, A. Martin, and R. Di Pietro, "To Docker or Not to Docker: A Security Perspective," *IEEE Cloud Computing,* vol. 3, no. 5, pp. 54-62, 2016.

[2] A. Randal, "The Ideal Versus the Real: Revisiting the History of Virtual Machines and Containers," *ACM Computing Surveys (CSUR),* vol. 53, no. 1, pp. 1-31, 2020.

[3] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and Linux containers," in *2015 IEEE international symposium on performance analysis of systems and software (ISPASS)*, 2015: IEEE, pp. 171-172.

[4] I. Mavridis and H. Karatza, "Performance and overhead study of containers running on top of virtual machines," in *2017 IEEE 19th Conference on Business Informatics (CBI)*, 2017, vol. 2: IEEE, pp. 32-38.

[5] *KVM. (n.d.b). https://www.linux-kvm.org/page/Main_Page*

[6] *XCP-ng - XenServer Based, Community Powered. (n.d.). https://xcp-ng.org/*

[7] Ratliff, J. (2023, January 23). Docker: *Accelerated, Containerized Application Development. Docker.* https://www.docker.com/

[8] *Podman. (n.d.). https://podman.io/*

[9] *Linux Containers. (n.d.). https://linuxcontainers.org/*

[10] *Red Hat - We make open source technologies for the enterprise.* (n.d.). https://www.redhat.com/

[11] J. Higgins, V. Holmes, and C. Venters, "Securing user defined containers for scientific computing," in *2016 International Conference on High Performance Computing & Simulation (HPCS)*, 2016: IEEE, pp. 449-453.

[12] B. Đorđević, R. Furtula, and V. Timčenko, "VMware ESXi and Microsoft Hyper-V hypervisor performance comparison," in 2020 28th Telecommunications Forum (TELFOR), 2020: IEEE, pp. 1-4.

[13] B. Đorđević, V. Timčenko, O. Pavlović, and N. Davidović, "Performance comparison of native host and hyper-based virtualization VirtualBox," in *2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2021: IEEE, pp. 1-4.

[14] A. Elsayed and N. Abdelbaki, "Performance evaluation and comparison of the top market virtualization hypervisors," in *2013 8th International Conference on Computer Engineering & Systems (ICCES)*, 2013: IEEE, pp. 45-50.

[15] P. Reddy and L. Rajamani, "Performance comparison of hypervisors in the private cloud," in *Advanced Computing, Networking and Informatics-Volume 2*: Springer, 2014, pp. 393-402.

[16] B. Đorđevic, M. Marjanović, and V. Timčenko, "Performance comparison of native host and hyper-based KVM virtualization," in *2020 28th Telecommunications Forum (TELFOR)*, 2020: IEEE, pp. 1-4.

[17] M. Moravcik, P. Segec, M. Kontsek, J. Uramova, and J. Papan, "Comparison of LXC and Docker technologies," in *2020 18th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 2020: IEEE, pp. 481-486.

[18] H. Gantikow, S. Walter, and C. Reich, "Rootless Containers with Podman for HPC," in *International Conference on High Performance Computing*, 2020: Springer, pp. 343-354.

[19] N. G. Bachiega, P. S. de Souza, S. M. Bruschi, and S. d. R. de Souza, "Performance evaluation of container's shared volumes," in *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2020: IEEE, pp. 114-123.

[20] D. Saxena and N. Sharma, "Analysis of Docker Performance in Cloud Environment," in *Advances in Information Communication Technology and Computing*: Springer, 2021, pp. 9-18.

[21] Q. Zhang, L. Liu, C. Pu, Q. Dou, L. Wu, and W. Zhou, "A comparative study of containers and virtual machines in big data environment," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2018: IEEE, pp. 178-185.

[22] P. Sharma, L. Chaufournier, P. Shenoy, and Y. Tay, "Containers and virtual machines at scale: A comparative study," in *Proceedings of the 17th international middleware conference*, 2016, pp. 1-13.

[23] F. Ramalho and A. Neto, "Virtualization at the network edge: A performance comparison," in 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016: IEEE, pp. 1-6.

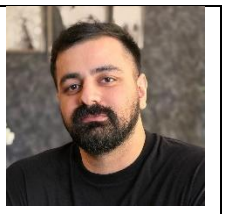
[24] 7-Zip LZMA Benchmark. (n.d.). https://www.7-cpu.com/

[25] MEMORY BANDWIDTH: STREAM BENCHMARK PERFORMANCE RESULTS. (n.d.). https://www.cs.virginia.edu/stream/

[26] IOzone Filesystem Benchmark. (n.d.). https://www.iozone.org/

[27] The Netperf Homepage. (n.d.). https://hewlettpackard.github.io/netperf/

معرفی نویسندگان:



حسین آقاسی زاده مدرک کارشناسی خود را در رشته مهندسی کامپیوتر از دانشگاه آزاد اسلامی در سال ۱۳۹۸ و مدرک کارشناسی ارشد خود را در همین رشته از دانشگاه مازندران در سال ۱۴۰۲ دریافت کرده است. پژوهش کارشناسی ارشد او بر بهبود عملکرد خوشه‌های کوبرنتیز و پلتفرم‌های سرورلس از طریق یکپارچه‌سازی ماشین‌های مجازی و کانتینرها متمرکز بوده است. حوزه‌های پژوهشی او شامل رایانش ابری، سرورلس و اینترنت اشیا است. در حال حاضر، او دانشجوی دکترای دانشگاه ترنتو و بنیاد برونو کسلر (FBK) در واحد تحقیقاتی OpenIoT است.

ایمیل: haqasizade@fbk.eu

احسان عطائی تحصیلات خود را در مقاطع کارشناسی، کارشناسی ارشد و دکتری رشته مهندسی کامپیوتر، به ترتیب در سال‌های ۱۳۸۱، ۱۳۸۳ و ۱۳۹۶ از دانشگاه صنعتی شریف تهران اخذ نموده و هم‌اکنون استادیار گروه مهندسی کامپیوتر دانشکده مهندسی و فناوری دانشگاه مازندران است. زمینه‌های پژوهشی موردعلاقه ایشان عبارت‌اند از: رایانش ابری، سامانه‌های توزیع‌شده، مدل‌سازی کارایی و اتکاپذیری و یادگیری ماشینی.

ایمیل: ataie@umz.ac.ir

مصطفی بستام استادیار دانشکده مهندسی کامپیوتر دانشگاه مازندران است. او در سال ۱۳۹۷ دکترای خود را در رشته شبکه‌های کامپیوتری از دانشگاه صنعتی امیرکبیر (پلی‌تکنیک تهران) دریافت کرد. مدرک کارشناسی خود را در رشته مهندسی کامپیوتر از دانشگاه شهید باهنر کرمان در سال ۱۳۸۵ و مدرک کارشناسی ارشد خود را در رشته شبکه‌های کامپیوتری از دانشگاه صنعتی امیرکبیر در سال ۱۳۸۸ اخذ کرده است. علایق پژوهشی او شامل اینترنت اشیاء (IoT)، رایانش ابری و مه، شبکه‌های نرم‌افزار محور(SDN) ، بلاک چین و یادگیری ماشین است.

ایمیل: bastam@umz.ac.ir