دانشکده مهندسی کامپیوتر

بررسی چالش‌ها و راه‌حل‌های مهاجرت زنده‌ی واحدهای اجرایی

سمینار کارشناسی ارشد

در رشته مهندسی کامپیوتر گرایش نرم‌افزار

(تمرکز سیستم‌های کامپیوتری)

ارائه‌دهنده:

هومن به‌نژاد فرد

استاد راهنما:

دکتر محسن شریفی

آبان ۱۳۹۶


بسم الله الرحمن الرحيم

چکیده

با ظهور فناوری‌هایی نظیر مجازی‌سازی و کانتینرها، پردازه‌ها در سیستم‌عامل تنها واحدهای اجرایی محسوب نمی‌شوند. با فناوری مجازی‌سازی می‌توان ماشین‌های مجازی را در قالب یک یا چند پردازه به سیستم‌عامل معرفی کرد و به آن‌ها منابع را تخصیص داد. با ظهور محاسبات ابری و محاسبات توان بالا، زمان پاسخ و بهره‌وری منابع بسیار اهمیت پیداکرده است. در این نوع سامانه‌ها معمولا از فناوری مجازی-سازی استفاده می‌کنند و برای بالا بردن بهره‌وری منابع فیزیکی نظیر پردازنده، حافظه و ورودی/خروجی از مهاجرت زنده ماشین‌های مجازی استفاده می‌کنند. مهاجرت زنده فرآیندی است که یک ماشین مجازی را از یک ابرناظر به یک ابرناظر دیگر بدون متوقف کردن آن، انتقال می‌دهد. این روند همواره در مقابل زمان پاسخ و کارایی برنامه کاربردی درون ماشین مجازی بوده و سربارهای پردازشی و ارتباطی دارد. مطالعات زیادی روی فرآیند مهاجرت زنده انجام شده است که هر یک به‌منظور کم کردن زمان پاسخ و بالا بردن بهره‌وری، یک یا چند معیار از مهاجرت زنده را در ابرناظرهای مختلف بهبود بخشیده‌اند. در این گزارش به بررسی کامل چالش‌ها و راه‌حل‌های آن‌ها و بازگو کردن نقاط مثبت و منفی هر یک پرداخته‌شده است.

واژه‌های کلیدی: واحدهای اجرایی، مجازی‌سازی، ماشین مجازی، مهاجرت زنده.

فهرست مطالب

عنوان **شماره‌ی صفحه**

فهرست شکل‌ها

عنوان **شماره‌ی صفحه**

فهرست جدول‌ها

عنوان **شماره‌ی صفحه**

فصل اول: مقدمه

۱-۱- شرح مسئله

امروزه در سیستم‌های عامل، پردازه‌ها تنها واحدهای اجرایی[۱] محسوب نمی‌شوند، ماشین‌های مجازی و کانتینرها[۲] که در سیستم‌عامل نیز به صورت پردازه ظاهر می‌شوند و مثل آن رفتار می‌کنند. مهاجرت پردازه یک روش انتقال پردازه بین دو ماشین است که با این کار می‌توان بارکاری، انعطاف‌پذیری در برابر خطا، مدیریت سامانه و دسترسی به داده را در سیستم‌های توزیعی بالا برد. در مهاجرت ماشین‌های مجازی، حفاظت از محتوای حالت ماشین و داده‌ی آن دارای اهمیت بالایی است و روش‌های استفاده‌شده برای مهاجرت موظف‌اند حالت ماشین یا به طور کلی پردازه را در طول انتقال ثابت نگه دارند و از انسجام آن پس از اتمام مهاجرت اطمینان حاصل نمایند. در روش مهاجرت زنده، پردازه به روند اجرایی خود ادامه می‌دهد و حالت آن تغییر می‌کند، این تغییرات به صورت زنده در حال انتقال است و چالش‌هایی نظیر تغییر در انسجام حافظه کاری آن، تغییر در انسجام داده‌های آن و تاثیر پاسخ‌دهی پردازه را فراهم می‌کند.

زمان پاسخ نیز جزء چالش‌هایی است که هم در مهاجرت زنده ماشین‌های مجازی، پردازه‌ها و کانتینرها مطرح است و این موضوع در مجازی‌سازها بسیار اهمیت دارد و زمان پاسخ نیز در صورتی معنا پیدا می‌کند که پردازه بتواند با اطلاعات درست انتقال‌یافته به روند اجرایی خود ادامه دهد و در تغییر اطلاعات در زمان انتقال می‌تواند در روند اجرایی پردازه‌ها مشکل به وجود بیاورد. فرآیندهای مهاجرت واحدهای اجرایی، سربار محاسباتی را بر روی پردازنده دارد و ترافیک زیادی را روی شبکه کامپیوتری اعمال می‌کند. این موضوع باعث مصرف انرژی و خود چالشی محسوب می‌شود. هر تغییر که باعث بهبود روند انتقال شود، می‌تواند مصرف انرژی را کاهش دهد.

۱-۲- معرفی حوزه سمینار

عوامل کاهش‌دهنده‌ی سطح کیفیت روند مهاجرت واحدهای اجرایی دارای گوناگونی و تعداد زیادی است. در این گزارش تلاش شده است که مروری کلی بر رویکردهای مختلف مهاجرت انواع واحدهای اجرایی نظیر پردازه‌ها و ماشین‌های مجازی انجام شود و چالش‌های هریک مورد بررسی قرار گیرد. سپس در مورد

[۱] Proccessing Units

[۲] Containers

معیارهای کارایی و ارزیابی‌های انجام‌شده بحث می‌شود. در بین انواع مهاجرت به مهاجرت زنده اهمیت بیشتری داده می‌شود زیرا از این روش برای توزیع بارکاری و بالا بردن بهره‌وری منابع در زمان اجرا استفاده می‌کنند. سپس مسئله را هم از دیدگاه امنیتی و هم از دیدگاه کارایی مورد بررسی قرار می‌گیرید. معیارهایی نظیر الف. مدت زمان خاموش بودن ب. مدت زمان مهاجرت پ. زمان آمادگی[1] و میزان داده-های انتقال‌یافته در طول مهاجرت زنده مطرح می‌شود و همچنین نقاط مثبت و منفی و کاربردهای هر یک را در پیاده‌سازی‌های انجام شده بیان می‌گردد.

۱-۳- ساختار گزارش

ساختار ادامه این گزارش به‌قرار زیر است: مفاهیم کلی پردازه و مهاجرت پردازه‌ها به همراه خصوصیات و اصول مهاجرت آن‌ها، روش‌های مهاجرت واحدهای اجرایی و معیارهای ارزیابی کارایی روش‌های مهاجرت پردازه در فصل ۲ ارائه می‌شود. در فصل ۳، رویکردها مختلف مهاجرت واحدهای اجرایی، روش‌های مشتق شده از راه‌کارهای فعلی مهاجرت و ارزیابی‌های انجام شده برای هر یک بیان می‌شود. در فصل ۴، نتیجه‌گیری و کارهای آینده در زمینه مهاجرت زنده واحدهای اجرایی مورد بحث قرار می‌گیرد.

[1] Preparation Time

فصل دوم: ادبیات موضوع

۲-۱- مقدمه

در این فصل قبل از بررسی رویکردها و چالش‌های مهاجرت زنده واحدهای اجرایی بخصوص ماشین‌های مجازی به مفاهیم و ادبیات مربوط در این زمینه پرداخته می‌شود. ابتدا مفهوم واحد اجرایی سپس با انواع مختلف آن که پردازه‌ها، ماشین‌های مجازی و کانتینرها می‌شوند، بیان می‌شود. سپس مبحث مجازی-سازی و ابرناظرها[۱] و روش‌های مختلف مجازی‌سازی بیان می‌شود. در ادامه، موضوع مهاجرت، گونه‌های مهاجرت و روش‌های پرکاربرد آن بیان‌شده و معیارهای کارایی هر یک بررسی می‌گردد.

۲-۲- واحدهای اجرایی

امروزه پردازه‌ها دیگر تنها واحدهای اجرایی در سیستم‌عامل شناخته نمی‌شوند. کانتینرها و ماشین‌های مجازی نیز نوعی پردازه و واحد اجرایی محسوب می‌شوند. این‌گونه واحدها خود به صورت یک یا چند پردازه در سیستم‌عامل معرفی می‌گردند و از منابع سیستم استفاده می‌کنند. پردازه یک مفهوم اصلی در سیستم‌عامل است که باوجود آن سیستم‌عامل معنا پیدا می‌کند. هر واحد اجرایی دارای منابع پایه‌ای مثل پشته، حافظه، داده و فضای آدرس مخصوص به خود است که در آن فضا اجرا می‌شود [1].

ماشین‌های مجازی نیز، به صورت پردازه‌های مجزا[۲] از هم در ابرناظرها ظاهر می‌شود، لذا به آن‌ها نیز واحد اجرایی گفته می‌شود. کانتینرها که وظیفه مجزا کردن پردازه‌ها را بر عهده دارند نیز در اصل پردازه شناخته می‌شوند. در ادامه به تعریف و مفاهیم هر یک از آن‌ها پرداخته می‌شود.

۲-۲-۱- پردازه

پردازه به یک نمونه از برنامه در حال اجرا گفته می‌شود که دارای پشته، حافظه و فضای آدرس مخصوص به خود است. هر پردازه یک پردازنده مجازی مخصوص به خود را دارد. برای درک بهتر، پردازنده را به چند پردازنده مجازی تقسیم می‌کنیم که هر پردازنده مجازی مسئول اجرا کردن یک پردازه می‌باشد [1]. در حالتی که فقط یک پردازنده واقعی روی ماشین قرار داشته باشد، زمان‌بند[۳] سیستم‌عامل پردازنده‌های

---

[۱] Hypervisor

[۲] Isolate

[۳] Scheduler

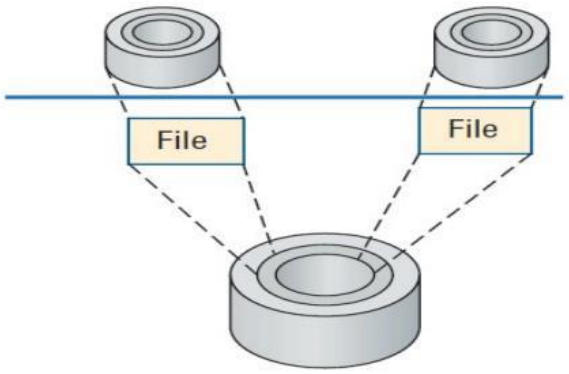
مجازی را زمان‌بندی می‌کند و به هر یک، بازه‌ی مشخص زمانی را برای اجرا تخصیص می‌دهد. این روند در اصل برای پردازه‌ها انجام می‌شود ولی پردازنده‌ها نیز می‌تواند مجازی‌سازی شوند که مفصل در بخش بعدی به آن پرداخته می‌شود.

۲-۲-۲- فناوری ماشین مجازی

فناوری مجازی‌سازی یک سامانه کامپیوتری را به تعدادی ماشین که به صورت جزئی یا کاملا از یکدیگر مجزا هستند تقسیم می‌کند [2]. به این ماشین‌ها، ماشین مجازی یا ماشین مهمان گفته می‌شوند. فناوری مجازی‌سازی که امروزه وجود دارد، جدید نمی‌باشد و حاصل تلاش‌های گذشته می‌باشد. این فناوری اولین بار توسط IBM در سال ۱۹۶۰ مطرح گردید [3]. IBM برای به اشتراک گذاشتن قدرت پردازشی و ذخیره‌سازی[۱] سامانه‌های کامپیوتری غول‌پیکر خود از این فناوری استفاده کرد. ولی این فناوری به دلیل کاهش کارایی، سربار منابع و همچنین ظهور کامپیوترهای شخصی برای چندین دهه منسوخ‌شده بود. در سال ۱۹۹۳ یک رشد ناگهانی در تعداد خدمت‌دهنده[۲] دیده شد [3]. هر چند تعداد خدمت‌دهنده‌ها درحال افزایش بود ولی تعداد زیادی بدون استفاده بودند چون امکان اجرای بیشتر از یک برنامه کاربردی بر روی یک خدمت‌دهنده نبود، حتی اگر این برنامه تنها بخشی از منابع خدمت‌دهنده را استفاده می‌کرد. در نتیجه استفاده از مجازی‌سازی با پیشرفت‌های نرم‌افزاری و قابلیت‌های سخت‌افزاری به ویژه توسعه و به‌کارگیری فناوری‌های خاص تعبیه‌شده در سخت‌افزار مانند مجازی‌سازی اینتل که موجب بهبود کارایی و کاهش سربار شده، دوباره مورد توجه قرار گرفت. هم‌اکنون این فناوری در بسیاری از سامانه‌های محاسباتی از دستگاه‌های موبایل تا کارگزارها باقدرت پردازشی بالا به کار می‌رود. شاید به نظر برسد که مجازی‌سازی و انتزاع[۳] یک مفهوم را بیان می‌کنند. اما در حقیقت این دو مفهوم از یکدیگر متمایز هستند. هدف از انتزاع، ساده‌سازی و حذف جزئیات است، اما هدف از مجازی‌سازی لزوماً ساده کردن و حذف جزئیات نیست. به عنوان مثال، فایل یک انتزاع از دیسک است که بسیاری از جزئیات و پیچیدگی‌های دیسک را ندارد. از طرف دیگر در بسیاری از کاربردها، نیاز است که با استفاده از فناوری مجازی‌سازی یک دیسک حقیقی را به چندین دیسک مجازی تبدیل کنیم. نرم‌افزار مجازی‌سازی، نگاشتی بین محتویات هر

[۱] Storage
[۲] Servers
[۳] Abstraction

یک از این دیسک‌های مجازی با محتویات دیسک واقعی فراهم می‌کند. تمام جزئیات و پیچیدگی‌های دیسک حقیقی در هر یک از دیسک‌های مجازی تقلید می‌شود. بنابراین، سطح جزئیات در واسط دیسک مجازی نظیر جزئیات آدرس‌بندی شیارها[1] و قطاع[2] مشابه با دیسک حقیقی باقی می‌ماند، بدون اینکه هیچ انتزاعی روی دهد. دستور نوشتن بر روی هر یک از این دیسک‌های مجازی تبدیل به دستور نوشتن بر روی فایل و سپس تبدیل به دستور نوشتن بر روی دیسک حقیقی می‌شود [4]. همان‌طور که در شکل (۲-۱) نشان داده شده است، مجازی‌سازی از انتزاع فایل به عنوان یک گام میانی برای ایجاد نگاشت بین دیسک حقیقی و دیسک مجازی استفاده می‌کند.

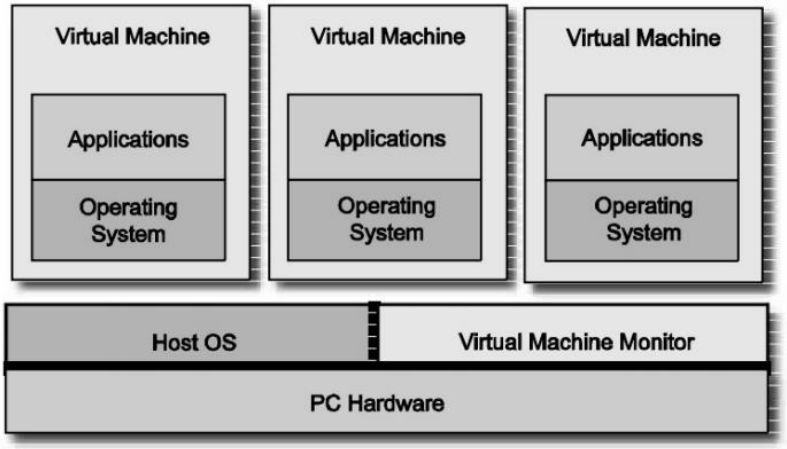


شکل (۲-۱) پیاده‌سازی دیسک‌های مجازی [4]

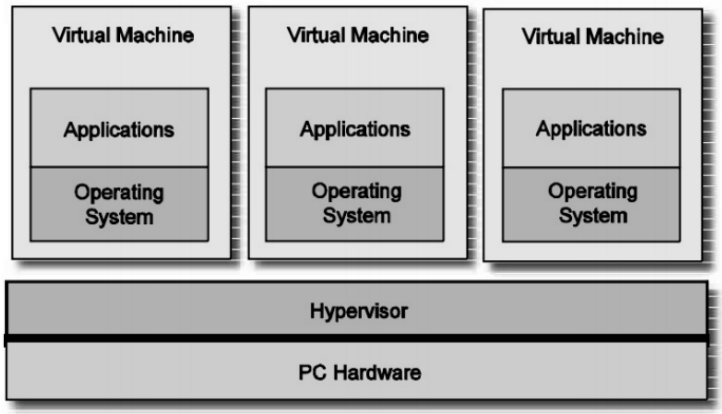
مفهوم مجازی‌سازی نه‌تنها در زیرسامانه‌هایی مثل دیسک بلکه در یک ماشین کامل نیز می‌تواند استفاده شود. یک ماشین مجازی با اضافه کردن لایه‌ای نرم‌افزاری به ماشین حقیقی پیاده‌سازی می‌شود. این لایه نرم‌افزاری، معماری مطلوب ماشین مجازی را صرف‌نظر از سخت‌افزار زیرین فراهم می‌کند. کاربران مختلف با استفاده از ماشین‌های مجازی می‌توانند برنامه‌های خود را با هر نوع سیستم‌عامل و پیکربندی موردنظر خود را بر روی هر ماشین حقیقی به اجرا درآورند. علاوه بر این، مجازی‌سازی این قابلیت را فراهم می‌کند که یک کامپیوتر شخصی یا کارگزار بتواند به طور هم‌زمان چندین سیستم‌عامل مختلف یا چندین برنامه از یک سیستم‌عامل یکسان را بر روی منابع سخت‌افزاری خود اجرا کند. بنابراین، کاربران می‌توانند به‌جای این‌که هر یک از برنامه‌های خود را بر روی چندین ماشین مجزا اجرا کنند، همه

[1] Tracks
[2] Sector

آن‌ها را فقط بر روی یک ماشین فیزیکی اجرا کنند. این امر مشکل محدودیت منابع سخت‌افزاری و درصد استفاده پایین از منابع را در مراکز داده از بین خواهد برد. هر ماشین حقیقی را می‌توان با نصب یک لایه نرم‌افزاری بر روی آن مجازی کرد. این لایه نرم‌افزاری به عنوان ناظر ماشین مجازی یا ناظر شناخته می-شود که مسئولیت ایجاد، حذف، توقف و ازسرگیری ماشین‌های مجازی و مهم‌تر از همه، مدیریت منابع بین ماشین‌های مجازی را بر عهده دارد. همان‌طور که در شکل (۲-۲) نشان داده شده است، اگر این لایه نرم‌افزاری در کنار سیستم‌عامل میزبان اجرا شود و منابعی که باید مجازی و یا تخصیص داده شوند مشخص کند به آن ناظر ماشین مجازی گفته می‌شود. اگر این لایه نرم‌افزاری به طور مستقیم بر روی سخت‌افزار اجرا شود به آن ناظر گفته می‌شود که در شکل (۲-۳) نشان داده شده است.



شکل (۲-۲) پیاده‌سازی ناظر ماشین مجازی [3]



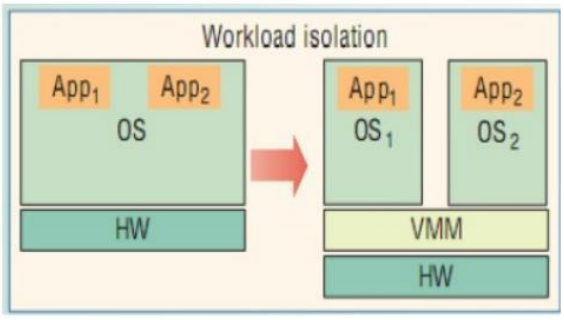
شکل (۲-۳) پیاده‌سازی ناظر [3]

◘ مزایای مجازی‌سازی

فناوری مجازی‌سازی مزایای متعددی دارد که ازجمله آن‌ها می‌توان به مجزاسازی، متمرکزسازی و مهاجرت اشاره کرد [5, 6, 7]. هر یک از این مزایا در ادامه به طور مختصر شرح داده شده است.

مجزاسازی

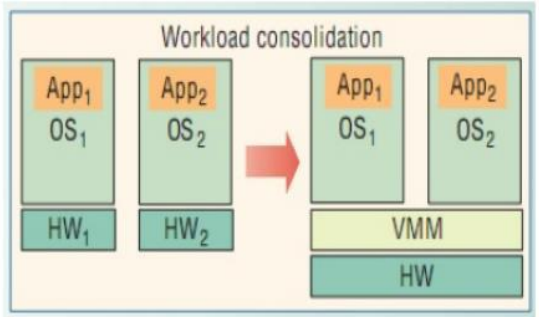
این قابلیت مجازی‌سازی سبب می‌شود ماشین‌های مجازی به صورت مستقل اجرا شوند که این امر موجب بهبود امنیت می‌شود. همچنین این قابلیت موجب افزایش قابلیت اطمینان می‌شود زیرا خرابی یک ماشین مجازی بر ماشین دیگری اثر نمی‌گذارد. بعلاوه، یکی از مزایای دیگر کنترل بهتر کارایی است. این فرایند در شکل (۲-۴) نشان داده شده است.



شکل (۲-۴) مجزاسازی ماشین‌های مجازی [5]

متمرکزسازی

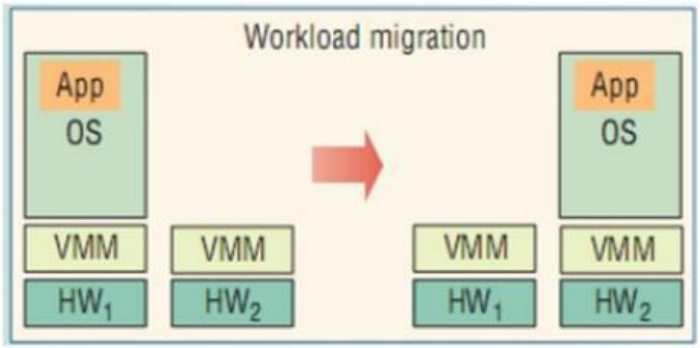
این قابلیت مجازی‌سازی سبب می‌شود که بارهای کاری بجای اینکه بر روی سخت‌افزارهای مجزا اجرا شوند بر روی یک سخت‌افزار اشتراکی اجرا شوند. بنابراین این قابلیت موجب افزایش بهره‌وری منابع و کاهش انرژی مصرفی می‌شود. این فرایند در شکل (۲-۵) نشان داده شده است.



شکل (۲-۵) متمرکزسازی ماشین‌های مجازی [5]

مهاجرت

این قابلیت مجازی‌سازی موجب تسهیل در نگهداری سخت‌افزار، توازن بار، مدیریت انرژی، افزایش کیفیت سرویس و تحمل‌پذیری در برابر خرابی می‌شود. این عمل با کپسوله‌سازی وضعیت سیستم‌عامل مهمان داخل ماشین مجازی و سپس انتقال آن به مکانی دیگر انجام می‌گیرد. این فرآیند در شکل (۲-۶) نشان داده شده است .



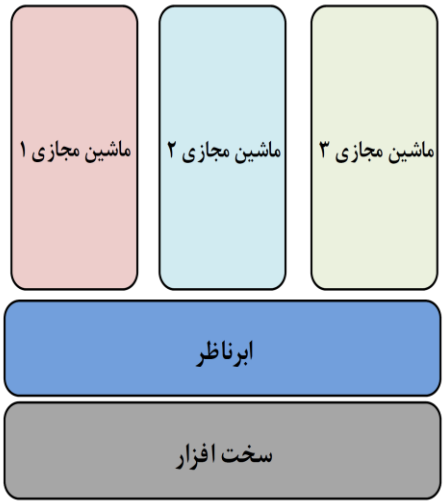
شکل (۲-۶) مهاجرت ماشین‌های مجازی [5]

۲-۲-۳- انواع ابرناظر

ابرناظرها به دو دسته تقسیم می‌شوند [8, 9]:

۱. نوع ۱

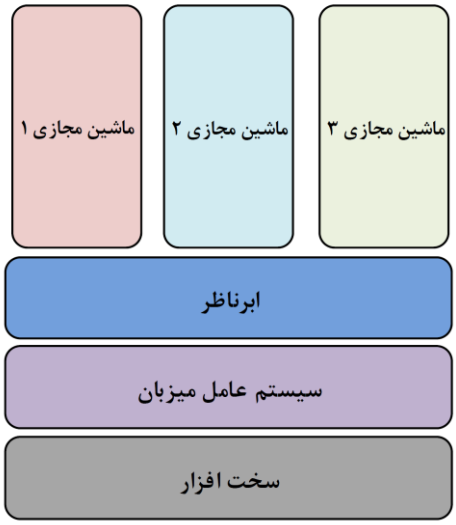
این نوع را Native یا Bare-metal نیز می‌گویند. در این مدل، ابرناظر نقش و وظایف سیستم-عامل را ایفا کرده و عملیات مدیریت منابع، عاملیت و فراهم کردن امکان فراخوانی‌های سیستمی را بر عهده دارد. ازاین‌رو مستقیماً روی سخت‌افزار فیزیکی نصب و اجرا می‌شود. برای مثال می-توان Xen، Hyper-V و VMWare-ESX را نام برد. معماری آن در شکل (۲-۷) آمده است.



شکل (۲-۷) محیط مجازی‌سازی شده مبتنی بر ابرناظر نوع ۱ [10]

۲. نوع ۲

در این مدل خود ابرناظر، به عنوان یک برنامه، روی یک سیستم‌عامل میزبان اجرا می‌شود. نوع دوم از فراخوانی‌های سیستمی سیستم‌عامل میزبان استفاده می‌کند و فقط وظیفه نظارت بر ماشین‌های مجازی را بر عهده دارد. ازاین‌رو ماشین‌های مجازی دو سطح با سخت‌افزار واقعی فاصله دارند. برای مثال می‌توان Oracle Virtualbox و VMWare Workstation نام برد. معماری آن در شکل (۲-۸) آمده است.



شکل (۲-۸) محیط مجازی‌سازی شده مبتنی بر ابرناظر نوع ۲ [10]

۲-۲-۴- روش‌های مجازی‌سازی

روش‌های مختلفی برای مجازی‌سازی وجود دارند که در ادامه به معرفی آن‌ها پرداخته می‌شود:

۱. مجازی‌سازی کامل

در این نوع از مجازی‌سازی، سخت‌افزار مجازی موردنظر به طور کامل برای ماشین مجازی فراهم می‌شود. هر سیستم‌عاملی می‌تواند، بدون نیاز به تغییر در ماشین مجازی است. همچنین سیستم‌عامل میهمان، از مجازی‌سازی و حضور ابرناظر خبر ندارد در این روش، هر دستور که سیستم‌عامل میهمان به روی سخت‌افزار مجازی اجرا می‌کند، توسط ابرناظر به دستور معادل آن روی سخت‌افزار حقیقی ترجمه می‌شود. البته در همین حالت هم امکان ترجمه خط به خط[۱] و همچنین ترجمه یک دسته از دستورات وجود دارد. روش اول کندتر بوده و روش دوم سریع‌تر است [11].

۲. مجازی‌سازی Para

در این روش هم ابرناظر وجود دارد. سیستم‌عامل میهمان کمی تغییر می‌کنند، به‌طوری‌که از حضور ابرناظر خبر دارند و می‌توانند مستقیماً با خود ابرناظر ارتباط داشته باشند. در این روش دیگر سربار ترجمه و تبدیل دستورات وجود ندارد. ازاین‌رو سرعت این روش نسبت به روش مجازی‌سازی کامل سریع‌تر است؛ اما محدودیت اصلی این روش این است که نمی‌توان از هر سیستم‌عاملی به عنوان سیستم‌عامل میهمان استفاده کرد [11].

۳. مجازی‌سازی در سطح سیستم‌عامل

در این روش، هسته[۲] سیستم‌عامل، امکان مجازی‌سازی را فراهم می‌کند. در این روش هسته می‌تواند مجازی‌سازی را در سطح کاربر ایجاد کند؛ اما این مجازی‌سازی به صورت کامل نیست؛ یعنی فقط چند محیط اجرایی به صورت مجزا در سطح کاربر ایجاد می‌شوند و همگی از یک هسته مشترک استفاده می‌کنند [11].

---

[۱] Inerpret

[۲] Kernel

◘ مجازی‌سازی پردازنده

پردازنده به عنوان مغز سیستم، مهم‌ترین منبع فیزیکی است که ابرناظر باید آن را مجازی‌سازی کند. امروزه، معماری x86 معماری غالب را تشکیل می‌دهد. در تمامی ابرناظرها، پردازنده‌های مجازی در غالب چند پردازه قرار می‌گیرد و زمان‌بند اصلی ابرناظر آن‌ها را زمان‌بندی می‌کند. در برخی از ابرناظرها، هر پردازنده مجازی، یک پردازه محسوب می‌شود و ورودی/خروجی ماشین مجازی آن در یک پردازه دیگر به زمان‌بند معرفی می‌گردد.

هر یک از پردازنده‌های مجازی یک سطح محافظتی دارند که ابرناظر یا سیستم‌عامل آن‌ها را در سطح مشخصی برای اجرا قرار می‌دهد. مدل حفاظتی پردازنده‌های مبتنی بر معماری x86 از چهار حلقه تشکیل شده است: حلقه ۰ تا حلقه ۳. همان‌گونه که در شکل (۲-۹) نشان داده شده است، در یک محیط بومی (مجازی‌سازی شده) سیستم‌عامل در حلقه ۰ و برنامه‌های کاربردی در حلقه ۳ اجرا می‌گردند (حلقه‌های ۱ و ۲ مورداستفاده قرار نمی‌گیرند). وقتی سیستم‌عامل کنترل پردازنده را به دست می‌گیرد، وضعیت اجرایی آن را حلقه ۰ قرار می‌دهد، درنتیجه سیستم‌عامل می‌تواند هم دستورالعمل‌های ممتاز و هم دستورالعمل‌های غیر ممتاز را اجرا کند. درصورتی‌که سیستم‌عامل بخواهد کنترل اجرا را به برنامه کاربردی برگرداند، وضعیت اجرایی پردازنده را حلقه ۳ قرار می‌دهد و درنتیجه برنامه کاربردی بخواهد دستورالعمل ممتاز را اجرا کند، تله[1] اتفاق می‌افتد و کنترل اجرا به سیستم‌عامل برگردانده می‌شود.

شکل (۲-۹) معماری یک محیط مجازی‌سازی نشده [12]

[1] Trap

حال این سوال مطرح می‌گردد که در یک محیط مجازی‌سازی شده، ابرناظر، سیستم‌عامل میهمان (که بر روی ماشین مجازی اجرا می‌گردد) و برنامه‌های کاربردی (که بر روی سیستم‌عامل میهمان اجرا می گردند) در چه حلقه‌هایی باید اجرا گردند و ارتباط بین سیستم‌عامل میهمان و ابرناظر چگونه باید صورت گیرد. بر این اساس، سه روش مختلف برای مجازی‌سازی پردازنده وجود دارد که در ادامه به بررسی آن‌ها می‌پردازیم: مجازی‌سازی کامل[۱]، با استفاده از ترجمه دودوئی[۲]، مجازی‌سازی جزئی[۳]، مجازی‌سازی به کمک سخت‌افزار.

◘ مجازی‌سازی کامل با استفاده از ترجمه دودوئی

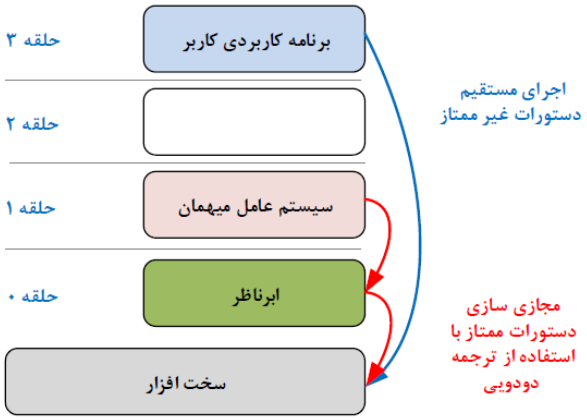
در این روش، ابرناظر در حلقه ۰، سیستم‌عامل میهمان در حلقه ۱ و برنامه‌های کاربردی در حلقه ۴ اجرا می‌گردند. دستورالعمل‌های غیر ممتاز موجود در کد سیستم‌عامل میهمان به‌صورت مستقیم بر روی پردازنده اجرا می‌گردند اما دستورالعمل‌های ممتاز با استفاده از روشی موسوم به ترجمه دودویی توسط ابرناظر مجازی‌سازی شده، سپس کد باینری حاصل از مجازی‌سازی بر روی پردازنده اجرا می‌گردد. مزیت این روش آن است که نیازی به تغییر کد سیستم‌عامل میهمان نیست و عیب آن این است که باید کد سیستم‌عامل میهمان قبل از اجرا پایش شده، دستورالعمل‌های ممتاز تشخیص داده شده و بعد از مجازی-سازی توسط ابرناظر بر روی پردازنده اجرا گردند. این کار سربار اجرایی تولید می‌کند و درصورتی‌که تعداد اجراهای دستورالعمل‌های ممتاز در سیستم‌عامل میهمان زیاد باشد، کارایی به طرز چشم‌گیری افت می-کند. معماری یک محیط مجازی‌سازی شده مبتنی بر این روش در شکل (۲-۱۰) نشان داده شده است.

---

[۱] Full Virtualization

[۲] Binary Translation

[۳] Paravirtualization

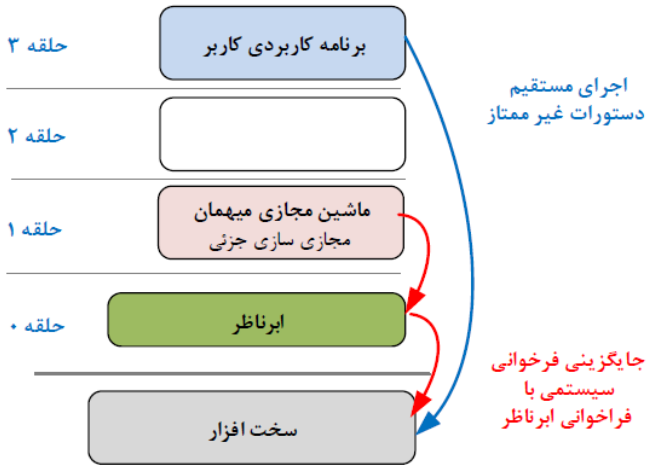


شکل (۲-۱۰) مجازی‌سازی کامل با استفاده از ترجمه دودویی [12]

◘ مجازی‌سازی جزئی

در روش مجازی‌سازی جزئی نیز ابرناظر در حلقه ۰، سیستم‌عامل میهمان در حلقه ۱ و برنامه‌های کاربردی در حلقه ۳ اجرا می‌شوند و کد سیستم‌عامل میهمان به‌گونه‌ای تغییر یافته است که در آن فراخوانی‌های سیستم[۱] به فراخوانی‌های ابرناظر[۲] تبدیل شده است. درصورتی‌که یک فراخوانی سیستم (به‌عنوان‌مثال، read و یا write) در سیستم‌عامل مهمان انجام گیرد، کنترل اجرا به ابرناظر منتقل می‌گردد و ابرناظر فراخوانی سیستمی مذکور را مجازی‌سازی کرده و دوباره کنترل را به سیستم‌عامل میهمان برمی‌گرداند مزیت این روش این است که نیازی به پایش کد سیستم‌عامل میهمان قبل از اجرا نیست، درنتیجه این روش سربار اجرایی کمتری نسبت به روش مجازی‌سازی کامل دارد. عیب این روش آن است که کد سیستم‌عامل میهمان باید تغییر داده شود و درصورتی‌که سیستم‌عامل میهمان کد بسته باشد (مانند سیستم‌عامل‌های ویندوز)، اجرای آن با استفاده از این روش مجازی‌سازی امکان‌پذیر نیست. معماری محیط مجازی‌سازی شده در شکل (۲-۱۱) نشان داده شده است.

[۱] System Call
[۲] Hypercall



شکل (۲-۱۱) مجازی‌سازی جزئی [12]

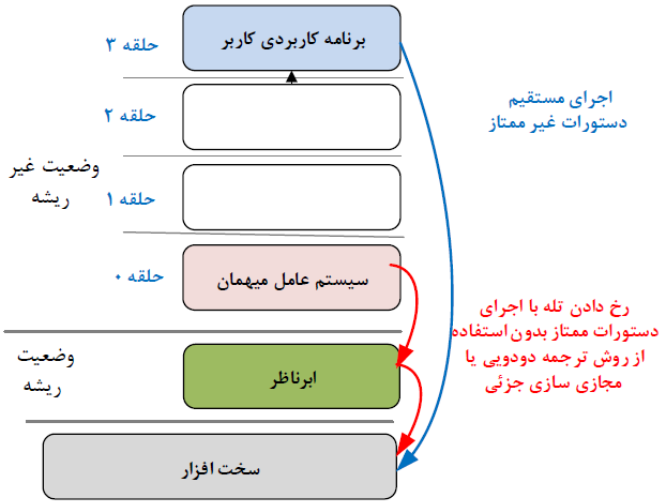
◘ مجازی‌سازی به کمک سخت‌افزار

در مجازی‌سازی کامل، کد سیستم‌عامل میهمان قبل از اجرا باید پایش شود و این امر سربار اجرایی تولید می‌کند. در مجازی‌سازی جزئی نیز کد سیستم‌عامل میهمان باید تغییر داده شود و برای سیستم‌عامل‌های کد بسته این امر امکان‌پذیر نیست. روش مجازی‌سازی به کم سخت‌افزار معایب مذکور را ندارد. این روش مبتنی بر فناوری Intel-VT یا AMD-VT می‌باشد.

در هر دو فناوری، افزونه‌ای[۱] به معماری x86 اضافه شده و دو وضعیت اجرایی جدید یعنی وضعیت ریشه[۲] و وضعیت غیر ریشه[۳] به معماری پردازنده اضافه شده است. پردازنده در هرلحظه در یکی از دو وضعیت ریشه یا غیر ریشه و در یکی از چهار حلقه‌ی ۰ تا ۴ در حال اجرا است. وقتی ابرناظر کنترل اجرا را به دست می‌گیرد، وضعیت پردازنده را در وضعیت ریشه و حلقه اجرایی را حلقه ۰ قرار می‌دهد و درنتیجه می‌تواند همه‌ی دستورالعمل‌های ممتاز را اجرا کند. وقتی ابرناظر می‌خواهد کنترل اجرا را به سیستم‌عامل میهمان برگرداند، وضعیت پردازنده را وضعیت غیر ریشه قرار می‌دهد. درصورتی‌که در هنگام اجرا شدن ماشین مجازی، فراخوانی سیستم انجام بگیرد با توجه به اینکه وضعیت پردازنده در وضعیت غیر ریشه قرار دارد، کنترل اجرا به ابرناظر برمی‌گردد. ابرناظر وضعیت پردازنده را در وضعیت ریشه قرار داده، فراخوانی سیستمی مذکور را مجازی‌سازی کرده، وضعیت پردازنده را وضعیت غیر ریشه قرار داده و کنترل اجرا را دوباره به سیستم‌عامل میهمان برمی‌گرداند. لازم به ذکر است که در وضعیت اجرایی غیر

[۱] Extension
[۲] Root Mode
[۳] Non-root Mode

ریشه، سیستم‌عامل میهمان در حلقه ۰ و برنامه‌های کاربردی در حلقه ۳ اجرا می‌گردند. معماری مذکور در شکل (۲-۱۲) نشان داده شده است.



شکل (۲-۱۲) مجازی‌سازی به کمک سخت‌افزار [12]

یک ابرناظر می‌تواند از یک یا چند روش فوق برای مجازی‌سازی پردازنده استفاده کند. برای نمونه، ابرناظر KVM از روش مجازی‌سازی به کمک سخت‌افزار و VMWare ESX از روش مجازی‌سازی کامل با استفاده از ترجمه دودویی برای مجازی‌سازی پردازنده استفاده می‌کند. ابرناظر Xen که در ادامه به بررسی آن خواهیم پرداخت، می‌تواند به‌صورت هم‌زمان از هر دو روش مجازی‌سازی جزئی و مجازی‌سازی به کمک سخت‌افزار پردازنده استفاده کند.

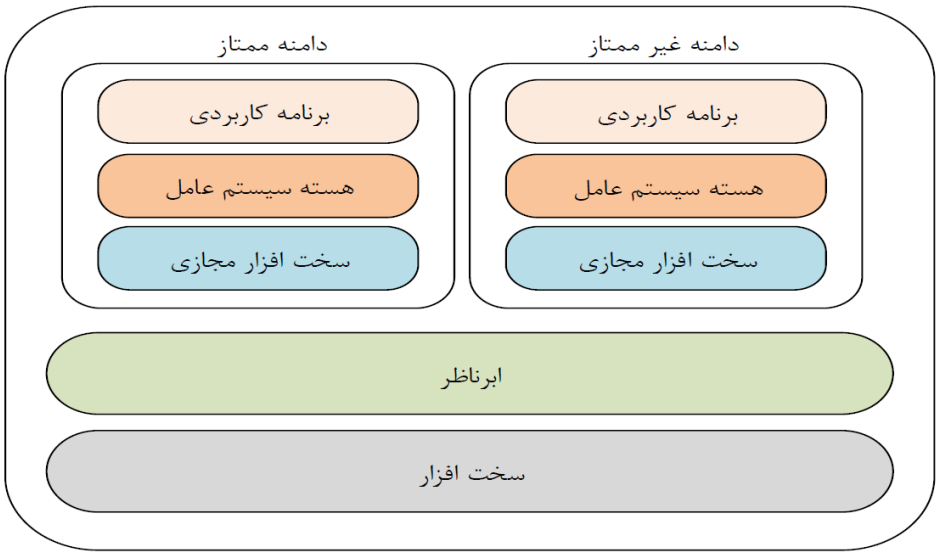
۲-۲-۵- ابرناظر Xen

پروژه Xen به صورت یک پروژه تحقیقاتی در دانشگاه کمبریج آغاز شد. هدف این پروژه ارائه یک زیرساختار عمومی برای محاسبات توزیع‌شده بود. پروژه مذکور در سال ۲۰۰۳ به نتیجه رسید و ابرناظر Xen را به عنوان یک فناوری کلیدی به جهانیان عرضه کرد. Xen به مرور زمان رشد و گسترش پیدا کرد و به عنوان یک فناوری کلیدی به جهانیان عرضه کرد. Xen یکی از متداول‌ترین و پرکاربردترین ابرناظرها در حوزه صنعت و آموزش تبدیل شد [13].

Xen یک ابرناظر نوع ۱ بوده و مستقیماً بر روی سخت‌افزار اجرا می‌گردد و می‌تواند از هر دو روش مجازی‌سازی جزئی و مجازی‌سازی به کمک سخت‌افزار برای مجازی‌سازی پردازنده استفاده کند. درنتیجه،

Xen می‌تواند یک ماشین مجازی را در دو وضعیت PVM[1] و HVM[2] نصب و اجرا نماید. در وضعیت PVM، نسخه تغییریافته سیستم‌عامل بر روی ماشین مجازی نصب می‌گردد و از روش مجازی‌سازی جزئی برای مجازی‌سازی پردازنده استفاده می‌گردد. در وضعیت HVM، نیازی به تغییر کد سیستم‌عامل نیست و روش مجازی‌سازی به کمک سخت‌افزار برای مجازی‌سازی پردازنده مورد استفاده قرار می‌گیرد.

Xen به صورت مستقیم با سخت‌افزار در ارتباط بوده و مسئولیت مدیریت پردازنده، حافظه اصلی و دستگاه‌های ورودی/خروجی بین ماشین‌های مجازی را بر عهده دارد.



شکل (۲-۱۳) معماری یک محیط مجازی‌سازی شده مبتنی بر ابرناظر Xen [13]

همان‌طور که در شکل (۲-۱۳) دیده می‌شود، ماشین‌های مجازی در ابرناظر Xen دامنه نام می‌گیرند و همواره در حالت PVM یک ماشین مجازی به نام دامنه ممتاز یا دامنه ۰ یا DOM0 وجود دارد. دامنه ممتاز هسته‌ی تغییریافته لینوکس است که وظایف خاصی را بر عهده دارد. این دامنه قبل از اجراشدن بقیه دامنه به وجود می‌آید و دو نقش اساسی را بر عهده دارد:

۱. راه‌انداز[3] دستگاه‌های ورودی/خروجی را در خود جای داده است و دسترسی مستقیم به دستگاه‌های ورودی/خروجی دارد. عملیات ورودی/خروجی سایر دامنه‌ها از طریق Dom0 انجام می‌گیرد.
۲. ابزارهای کنترلی و مدیریتی برای مدیریت سایر دامنه‌ها در Dom0 نصب و اجرا می‌گردد.

منظور از مدیریت سایر دامنه‌ها، ایجاد، توقف، حذف و مهاجرت سایر ماشین‌های مجازی می‌باشد.

[1] Paravirtualized Virtual Machine
[2] Hardware Assisted Virtual Machine
[3] Driver

برای مدیریت و کنترل دامنه‌های غیر ممتاز سرویس xend در Dom0 نصب و اجرا می‌گردد. کاربر درخواست ایجاد، توقف یا مهاجرت دامنه‌های غیر ممتاز را از طریق ابزارهای مدیریتی مانند xm برای xend می‌فرستند (ارتباط بین xm و xend از طریق رابط XML-RPC صورت می‌گیرد). سرویس Xend از طریق کتابخانه libxenctrl با ابرناظر Xen ارتباط برقرار کرده و درخواست‌ها را برای Xen ارسال می‌کند.

۲-۲-۶- ابرناظر KVM

KVM یک ابرناظر نوع ۲ است و به صورت یک ماژول هسته[۱] به هسته سیستم‌عامل لینوکس اضافه می‌شود. این ابرناظر برای زمان‌بندی کردن پردازنده‌های ماشین مجازی از زمان‌بند[۲] سیستم‌عامل لینوکس که CFS نام دارد، استفاده می‌کند. این ماژول می‌تواند مجازی‌سازی از نوع جزئی را با استفاده از Virtio فراهم آورد. Virtio یک نوع ورودی/خروجی مجازی است که هسته سیستم‌عامل مهمان برای ارتباط داشتن با ورودی/خروجی از آن استفاده می‌کند.

KVM می‌تواند این ویژگی را به سیستم‌عامل مهمان ندهد و مجازی‌سازی را به صورت مجازی‌سازی کامل انجام دهد. این کار توسط Qemu که یک شبیه‌ساز[۳] است انجام می‌شود. با استفاده از Virtio کارایی ورودی/خروجی سیستم‌عامل مهمان بهتر می‌شود زیرا به ازای هر درخواست ورودی/خروجی سیستم‌عامل مهمان که به صورت پردازه‌ای در سیستم‌عامل میزبان است، عملیات تله[۴] رخ می‌دهد و چون تعداد درخواست‌ها زیاد می‌باشد، کارایی ماشین مجازی پایین می‌آید. Virtio روشی است که سربار این تله را کم می‌کند و بخشی از کارایی ازدست‌رفته را برگرداند [14]. برای درک بهتر شکل (۳-۱۴) روند پاسخ‌دهی به ورودی/خروجی دیسک را در KVM و Virtio نشان می‌دهد.

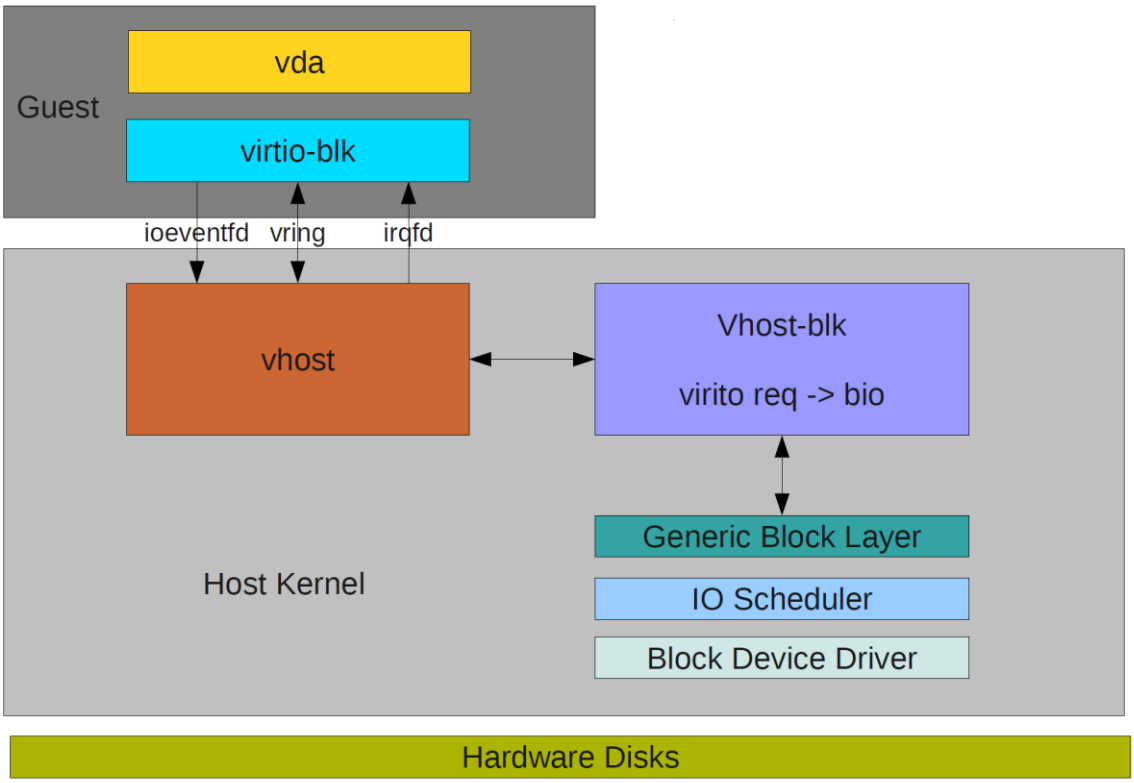
---

[۱] Kernel Module

[۲] Scheduler

[۳] Emulate

[۴] Trap



شکل (۲-۱۴) نحوه پاسخ‌دهی ورودی/خروجی دیسک با استفاده از Virtio

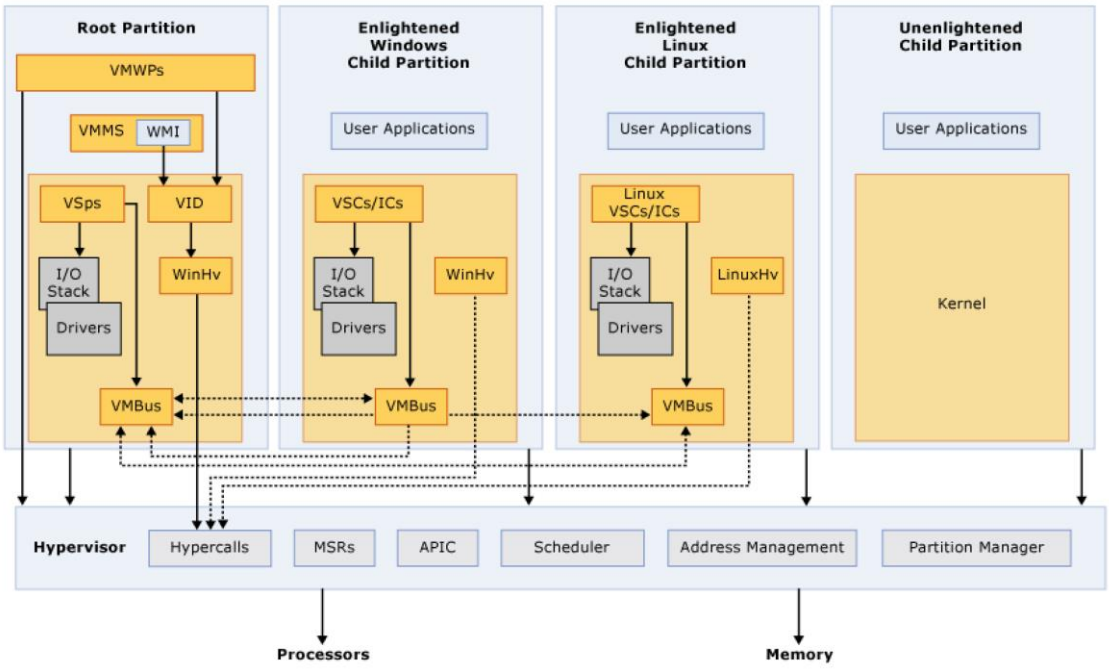
۲-۲-۷- ابرناظر Hyper-V

Hyper-V یک ابرناظر نوع ۱ مانند ابرناظر Xen است و توسط شرکت ماکروسافت تولید و پشتیبانی می‌شود. این میان‌افزار در سیستم‌عامل‌های Windows Server 2008 به بالا قابل‌اجرا است. البته در نسخه‌های غیر Server نیز قابل‌اجرا است. این میان‌افزار به صورت یک نقش[۱] در سیستم‌عامل‌های Server شرکت ماکروسافت ظاهر می‌شود. در اول شاید به نظر برسد که این محصول یک ابرناظر نوع ۲ باشد و در کنار سیستم‌عامل میزبان نصب می‌شود ولی معماری آن به‌گونه‌ای است که وقتی نقش آن فعال می‌شود، بعد از بارگذاری دوباره سیستم‌عامل، خود به‌جای Windows Server روی سخت‌افزار بارگذاری می‌شود و سیستم‌عامل را به عنوان یک ماشین مجازی روی خود بالا می‌آورد [15].

این ابرناظر همانند Xen از دو نوع مجازی‌سازی جزئی و مجازی‌سازی کامل پشتیبانی می‌کند. در این

[۱] Role

ابرناظر برای ایجاد قابلیت مجزاسازی از تکنیک قسمت‌بندی[1] استفاده شده است که حتما یک قسمت آن به عنوان ریشه شناخته می‌شود و سیستم‌عامل آن یکی از نسخه‌های ۶۴ بیتی Windows Server ویرایش سال ۲۰۰۸ یا سال‌های بالاتر است. ماشین‌های مجازی توسط قسمت ریشه کنترل و ایجاد می-گردند و این روند با استفاده از تکنیک hypercall به ابرناظر اعمال می‌گردد. ماشین‌های مجازی مهمان منابع ماشین را به صورت دستگاه‌های‌مجازی می‌بینند و درخواست‌های خود را یا از طریق VBus یا از طریق ابرناظر به قسمت ریشه ارسال می‌کنند. شکل (۲-۱۵) معماری کلی از Hyper-V را نشان می‌دهد.



شکل (۲-۱۵) نمای کلی از ابرناظر Hyper-V [15]

۲-۲-۸- کانتینرها

در چند سال اخیر یک مفهوم جدیدی با نام کانتینر معرفی شده است. کانتینرها یک نوع مجازی‌سازی سبک‌وزن[2] هستند. این نوع مجازی‌سازی سربار کمتری نسبت به مجازی‌سازی‌های قبلی دارد و به همین دلیل است که آن‌ها را سبک‌وزن می‌نامند [16]. در این نوع مجازی‌سازی هسته سیستم‌عامل تغییر پیدا نمی‌کند و پردازه‌هایی که قرار است روی ماشین‌های مجازی مجزا اجرا شوند، روی یک سیستم‌عامل به اجرا درمی‌آیند.

[1] Partitioning
[2] Light Weight

کانتینرها وظیفه‌ی مجزا کردن کامل پردازه‌های درون خود را نسبت به بقیه پردازه‌ها دارند. همانند ماشین‌های مجازی که وظیفه تفکیک پردازه‌ها و اثر آن‌ها روی پردازه‌ها در یک ماشین مجازی دیگر دارند، کانتینرها نیز موظف‌اند حافظه، فضای آدرس و هر چیزی که پردازه برای خود دارد را دیگر پردازه‌ها جدا کند با این تفاوت که کانتینرها نمی‌توانند سیستم‌عامل را برای هر یک از پردازه‌های خود مجازی-سازی کند. در کانتینرها هسته سیستم‌عامل را بین پردازه‌های درون خود به اشتراک می‌گذارد. بنابراین، اگر یک پردازه بخواهد از فراخوانی سیستمی خاصی استفاده کند، نمی‌تواند روی هر ماشین دلخواه به اجرا در بیاید. دلیل سبک‌وزن بودن آن‌ها نیز به خاطر به اشتراک‌گذاری هسته سیستم‌عامل است.

دلیل دیگر استفاده از کانتینرها این است که دیگر لازم نیست تعلقات[1] مختلف برنامه در ماشین نصب شود و فقط کافی است برنامه‌های جانبی دیگر در کانتینرهای مختلف اجرا و به کانتینری که برنامه اصلی در آن اجرا می‌شود متصل شود. برای درک بهتر این مفهوم برنامه‌ای را در نظر بگیرید که برای اجرا شده و خدمت‌رسانی به کاربران نیاز به پایگاه‌داده Oracle دارد. حال این برنامه می‌خواهد در ماشینی اجرا شود که این پایگاه‌داده در آن نصب نیست. در این حالت صرفا کافی است کانتینری ایجاد شود و پایگاه‌داده Oracle در آن باشد و ارتباط بین این دو کانتینر برقرار شود. این ارتباط می‌تواند از طریق سوکت و یا شبکه باشد و در حالتی خاص، می‌تواند از طریق فایل باشد این حالت تنها در سیستم‌عامل‌های مبتنی بر لینوکس امکان‌پذیر است.

کانتینرها کتابخانه‌های موردنیاز برای اجرا شدن برنامه را به همراه برنامه در خود جای می‌دهند و اجرا شدن پردازه‌های درون خود را مستقل از کتابخانه‌های نصب شده درون سیستم‌عامل می‌کند. کانتینرها نیز می‌توانند محدودیت‌های خاصی شبیه به ماشین‌های مجازی را بر روی پردازه‌های درون خود اعمال کنند. این محدودیت‌ها روی حجم مصرفی حافظه، نرخ خواندن/نوشتن روی دیسک و کنترل پهنای باند شبکه می‌باشد.

تفاوت کانتینرها با ابرناظرها در فراهم آوری سطح انتزاع است. کانتینرها در نمی‌تواند یک پردازنده خاص و یا نوع گذرگاه‌های ماشین سخت‌افزاری را مجازی‌سازی کنند. این دلیلی است که آن‌ها را سبک‌وزن می‌کند.

Docker، LXC و LXD از کانتینرهای معروف در دنیا هستند که بر روی سیستم‌عامل لینوکس نصب می‌شوند. البته، هسته سیستم‌عامل لینوکس محیطی برای اجرا شدن آن‌ها در خود فراهم آورده است. و

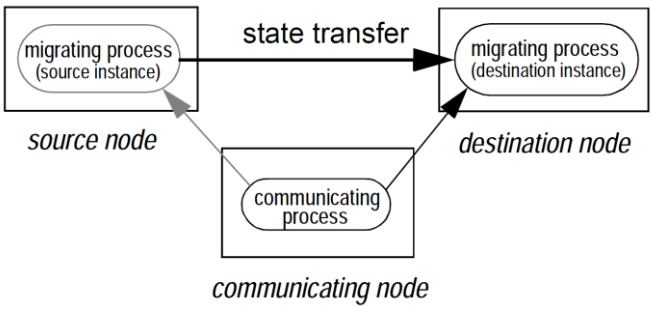
[1] Dependency

این کانتینرها با استفاده از قابلیت cgroup مجزاسازی بین پردازه‌ها را انجام می‌دهند.

۲-۳- مهاجرت

مهاجرت پردازه، عمل انتقال پردازه از یک گره پردازشی به گره پردازشی دیگر در زمان اجرای پردازه است [17]. در روند مهاجرت پردازه، وضعیت پردازه به ماشین دیگر انتقال پیدا می‌کند. این وضعیت شامل فضای آدرس پردازه، وضعیت ارتباطی[۱] (مثل فایل‌های باز شده و کانال‌های پیام[۲]) می‌باشد که برای درک بهتر آن، شکل (۲-۱۶) روند مهاجرت را در سطح بالا نشان می‌دهد.

در روند مهاجرت ابتدا، یک پردازه در سیستم‌عامل ماشین مقصد ایجاد می‌شود و سپس وضعیت پردازه در ماشین مبدا به ماشین مقصد انتقال پیدا می‌کند و بعد از آن، حافظه و فضای آدرس پردازه به ماشین مقصد انتقال پیدا می‌کند. در این روند، بعد از ایجاد پردازه در سیستم‌عامل ماشین مقصد، روند اجرایی پردازه در ماشین مبدا متوقف می‌شود و سپس بقیه مراحل انجام می‌شوند.



شکل (۲-۱۶) روند مهاجرت از دیدگاه سطح بالا [17]

عمل مهاجرت صرفا برای پردازه‌ها صورت نمی‌گیرد، بلکه برای ماشین‌های مجازی نیز انجام می‌شود. مهاجرت ماشین‌های مجازی به دو دسته تقسیم می‌شود: مهاجرت زنده[۳] (مهاجرت پویا یا مهاجرت گرم نیز نامیده می‌شود) و مهاجرت سرد[۴] (مهاجرت ایستا نیز نامیده می‌شود). در مهاجرت سرد وضعیت

[۱] Communication State
[۲] Message Tunnels
[۳] Live Migration
[۴] Cold Migration

ماشین مجازی از دست می‌رود و کاربران در سرویس خود متوجه اختلال خواهند شد. درحالی‌که در مهاجرت زنده ماشین مجازی در هنگام مهاجرت به کار خود ادامه می‌دهد و وضعیت خود را از دست نمی‌دهد و کاربران متوجه هیچ‌گونه اختلالی در سرویس نمی‌شوند.

در روند مهاجرت زنده، وضعیت یک ماشین مجازی در طول مهاجرت انتقال داده می‌شود. این وضعیت شامل محتوای حافظه، وضعیت پردازنده و فایل‌های سیستمی آن می‌شود. البته در برخی از مهاجرت‌ها فایل‌های سیستمی نظیر دیسکی که سیستم‌عامل درون آن نصب شده است، انتقال پیدا نمی‌کنند. وضعیت پردازنده ماشین مجازی متشکل از مقادیر ثبات‌های[1] کنترلی و محاسباتی پردازنده، حالت پردازنده و جدول صفحات حافظه است.

در مراکز داده بیشتر از حافظه‌های پایداری[2] نظیر NAS[3] و یا SAN[4] استفاده می‌کنند. این نوع حافظه‌ها بر روی بورد سخت‌افزاری ماشین‌های میزبان نصب نمی‌شوند و درون یک سیستم کاملا مجزا قرار می‌گیرند و امکان ذخیره‌سازی اطلاعات را بر روی شبکه محلی[5] و یا شبکه‌های خاص‌منظوره برای تبادل اطلاعات مثل iSCSI[6] فراهم می‌کند. این امر در مهاجرت پویا نقش اساسی ایفا می‌کند زیرا زمان انتقال دیسک ماشین مجازی مهمان که اغلب دارای حجم بیشتری نسبت به فضای حافظه ماشین مجازی است، بسیار زیاد است. لذا، در مهاجرت زنده فقط حافظه ماشین مجازی به ماشین میزبان دیگری انتقال پیدا می‌کند و اطلاعات دیسک آن از طریق شبکه قابل‌دسترسی خواهد بود. این امر تنها در مهاجرت‌هایی امکان دارد که ماشین میزبان هدف درون مرکز داده قرار داشته باشد و اگر این امر میسر نباشد لازم است اطلاعات دیسک ماشین مجازی نیز به مرکز داده هدف انتقال یابد. با این کار مدت زمان بسیاری صرف انتقال اطلاعات می‌شود و زمان مهاجرت افزایش پیدا می‌کند.

۲-۳-۲- اهداف و مزایای مهاجرت

مهاجرت اهداف و مزایای گوناگونی دارد که بسته به نوع کاربرد و برنامه کاربردی آن متفاوت می‌باشد. در این بخش به اهداف و مزایای مشترک آن‌ها پرداخته می‌شود [17].

---

[1] Register

[2] Persistance

[3] Network Attached Storage

[4] Storage Area Network

[5] Local Aria Network

[6] Internet Small Computer System Interface

- **دست‌یابی به توان پردازشی بیشتر**

زمانی که میزان بارکاری روی یک گره پردازشی بالا رود به صورتی که این گره پردازشی منابع آزادی برای تخصیص دادن به آن نداشته باشد، مهاجرت نقش مهمی ایفا می‌کند. به‌عنوان‌مثال، در حالتی که گره پردازشی و پردازه‌های روی آن در حال استفاده از تمامی هسته‌های پردازنده باشند، پردازه‌ای دیگر در سیستم‌عامل آن به وجود آید، در این حالت اگر گره پردازشی این پردازه را درون خود اجرا کند، باید به آن منابعی اختصاص دهد. این منابع زمانی اختصاص پیدا می‌کنند که میزان بهره‌برداری آن‌ها اشباع‌شده است. با این کار پردازه‌های دیگری که در اجرا بودند، سهم کمتری را از منابع قابل در دسترس خود پیدا می‌کنند. در این حالت، پردازه‌های در حال اجرا تنزل[1] پیدا می‌کنند. برای جلوگیری از تنزل اجرایی پردازه‌ها، میان‌افزاری[2] که روی سیستم‌عامل نصب شده است، پردازه‌ای را با الگوریتم‌های خاص خود و با توجه به میزان استفاده آن از منابع ماشین انتخاب می‌کند و به ماشین دیگری مهاجرت می‌دهد. با این کار بارکاری بین ماشین‌های مختلف توزیع می‌شود.

میان‌افزارهایی نظیر MOSIX و MACH برای این امر وجود دارند.

- **بهره‌برداری از منابع محلی**

حالتی را در نظر بگیرید که پردازه‌ای برای اجرا شدن باید داده‌هایی دسترسی پیدا کند در ماشین دیگری است. برای اجرا شدن آن باید دسترسی به داده‌ها را از طرق شبکه فراهم شود یا داده‌ها در ماشینی که پردازه روی آن است انتقال یابد. این روند می‌تواند زمان زیادی را در برگیرد و این زمان متناسب با حجم داده‌های مورد نظر افزایش یا کاهش پیدا می‌کند. برای اینکه سربار انتقال داده‌ها را کاهش دهیم می‌توانیم پردازه را به ماشین مقصد انتقال دهیم. با این کار زمان دسترسی به داده‌ها و یا منابع مورد نظر را بسیار کاهش داده‌ایم.

منابع مورد استفاده پردازه می‌تواند پردازنده، باند شبکه و یا ورودی/خروجی، حافظه و فایل باشد.

- **به اشتراک‌گذاری منابع**

زمانی مهاجرت با این هدف صورت می‌گیرد که پردازه بخواهد به یک نوع سخت‌افزار خاص یا حافظه

[1] Degradation

[2] Middleware

بیشتر برای مدت کوتاه یا در منبع خاص دیگری دسترسی پیدا کند. معمولا این روش در بالا بردن بهره-وری منابع استفاده می‌شود.

- **انعطاف‌پذیری در برابر خطا**

زمانی یک گره پردازشی به مدت زیادی روشن باشد و پردازه‌های زیادی را روی خود درحال اجرا نگه داشته باشد، احتمال رخ دادن خطا در آن افزایش پیدا می‌کند. بروز خطا در برنامه‌هایی که مدت زمان زیادی را در حال اجرا بودند می‌تواند ... باشد. به همین علت، این‌گونه پردازه‌ها را مهاجرت می‌دهند تا از رخ دادن خطا در آن‌ها جلوگیری کنند.

- **مدیریت سیستم**

زمانی مهاجرت به این منظور صورت می‌گیرد که گره پردازشی نیاز به تعمیر داشته باشد یا در حال خاموش شدن باشد. برای اینکه پردازه‌ها به مشکل برنخورند و در حالت اجرایی خود باقی بمانند، میان-افزار آن‌ها را به ماشین دیگری مهاجرت می‌دهد. با این کار می‌توان پره پردازشی را تعمیر یا جایگزین کرد.

۲-۳-۳- مهاجرت ایستا

مهاجرت ایستا به مهاجرتی گفته می‌شود که پردازه یا ماشین مجازی مورد نظر در حال اجرا نباشد. این‌گونه مهاجرت بیشتر در ماشین‌های مجازی کاربرد دارد. در این روش از واحد اجرایی مورد نظر اسنپشات گرفته می‌شود و از حالت اجرایی خود متوقف می‌شود. سپس اسنپشات به ماشین دیگری انتقال می‌یابد و به حافظه منتقل می‌شود و سیستم‌عامل روند اجرایی آن را ادامه می‌دهد. در این نوع مهاجرت مدت زمان ازکارافتادگی واحد اجرایی بسیار زیاد است و برای پردازه‌هایی که وظیفه خدمت‌رسانی به کاربران را دارند بسیار بد است. ازاین‌رو، برای حل مشکل ازکارافتادگی زیاد واحد اجرایی روش مهاجرت پویا معرفی شده است.

۲-۳-۴- مهاجرت پویا

تفاوتی که مهاجرت پویا با مهاجرت ایستا دارد در مدت زمانی است که پردازه یا ماشین مجازی ازکارافتاده

است و در حال مهاجرت است. در مهاجرت پویا زمان ازکارافتادگی واحد اجرایی نسبت به مهاجرت ایستا بسیار کم است و کاربران متوجه مهاجرت نمی‌شوند. در این روش نیز از وضعیت پردازه اسنپشات گرفته می‌شود و به ماشین مقصد انتقال پیدا می‌کند با این تفاوت که در روند اسنپشات پردازه به کار خود ادامه می‌دهد. وضعیت پردازه و یا ماشین مجازی در کنار روند اجرایی آن به ماشین مقصد ارسال می‌گردد و در یک بازه کوتاه، پردازه از حالت اجرایی متوقف شده و به ماشین مقصد ارسال می‌گردد.

این نوع مهاجرت دارای رویکردهای متفاوتی است، ۳ روش مرسوم آن در ادامه توضیح داده شده است.

۲-۳-۵- روش‌های مهاجرت پویا

انتقال حافظه ماشین مجازی یا پردازه به عنوان مهم‌ترین جنبه در مهاجرت زنده به چندین شیوه قابل‌اجرا است. روش‌های انتقال حافظه از ماشین مبدا به ماشین مقصد بر اساس بارکاری ماشین مجازی و نوع نرم‌افزار درون آن، متفاوت است ولی در بیشتر مراکز داده از سه روش مرسوم زیر استفاده می‌شود که در ادامه به معرفی آن‌ها پرداخته می‌شود [18].

- **روش پیش‌کپی**[۱]

این روش مبتنی بر انتقال تکراری[۲] صفحات حافظه است. در این روش ماشین مجازی درحال اجرا می‌باشد و در کنار آن صفحات حافظه آن از ماشین مبدا به ماشین مقصد انتقال می‌یابد. شکل (۲-۱۷) نشان‌دهنده این روش می‌باشد.

سرعت انتقال صفحات حافظه نسبت به سرعت استفاده شدن آن‌ها توسط پردازنده بسیار پایین است و به همین دلیل بیشتر صفحات حافظه در زمان مهاجرت توسط پردازنده تغییر پیدا می‌کنند و به‌اصطلاح کثیف[۳] می‌شود. این صفحات باید در دوره بعدی، به ماشین مقصد انتقال پیدا کنند. تعداد صفحات تغییریافته در قدم‌های اول بسیار زیاد است و در قدم‌های بعدی به نسبت کاهش پیدا می‌کند. در این روش یک آستانه[۴] تعیین می‌شود که فرآیند انتقال صفحات تغییریافته پایان یابد. بعد از پایان فرآیند، ماشین مجازی که در حال اجرا است، از اجرا می‌ایستد. حال کل صفحات باقی‌مانده به ماشین هدف

---

[۱] Pre-copy

[۲] Iterative

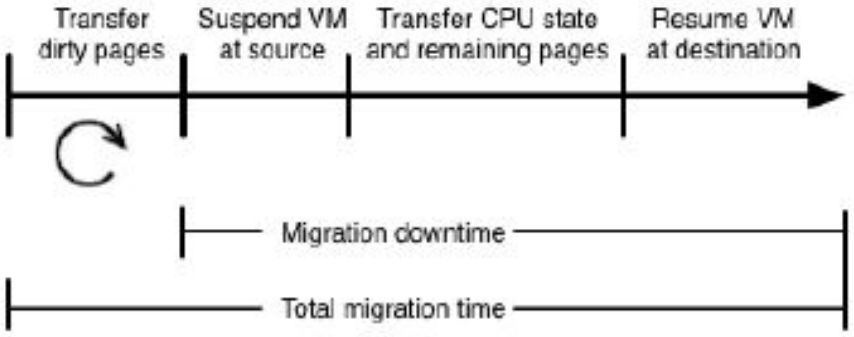
[۳] Dirty

[۴] Threshold

انتقال پیدا می‌کند. بعد از انتقال کامل حافظه ماشین مجازی، وضعیت پردازنده به ماشین هدف انتقال می‌یابد. سپس ماشین مجازی در ماشین مقصد شروع به فعالیت می‌کند.

این روش توسط بیشتر ابرناظرها نظیر VMWare، Xen و Qemu پشتیبانی می‌شود. برای بهبود دادن به این روش از تابع چگالی احتمال استفاده می‌کنند تا بتوانند زودتر به آستانه اتمام انتقال برسند. این روش برای برنامه‌هایی که با حافظه زیاد سروکار دارند، کارایی خوبی ندارد زیرا صفحات حافظه به سرعت تغییر می‌کنند و هیچ‌گاه تعداد صفحات تغییر نکرده به حد آستانه خود نمی‌رسد.

اگر ماشین مجازی در حال خواندن از حافظه باشد و بارکاری قابل‌نوشتن (WWS[1]) آن کم باشد، زمان ازکارافتادگی آن کم می‌شود و مدت زمان کل مهاجرت کاهش می‌یابد. ولی اگر بارکاری قابل‌نوشتن آن زیاد باشد، تعداد صفحات تغییریافته‌ی باقی‌مانده به حد آستانه خود نمی‌رسد و در آخر مدت زمان زیادی باید صرف بشود تا صفحات باقی‌مانده انتقال پیدا کنند و این امر باعث می‌شود ماشین مجازی زمان بیشتری را در حالت ایستاده سپری کند. همچنین، اگر پهنای باند شبکه[2] کم باشد، کارایی بهتری نسبت به روش بعدی از خود نشان می‌دهد.



شکل (۲-۱۷) روش پیش کپی [18]

- **روش پس‌کپی[3]**

این روش نیز مانند روش پیش‌کپی مبتنی بر انتقال تکراری است. تفاوت این روش با روش قبلی در زمان شروع انتقال است. در این روش ابتدا در ماشین میزبان مقصد فضای حافظه‌ای به‌اندازه فضای حافظه

[1] Writable Working Set
[2] Band Width
[3] Post-copy

ماشین مجازی در ماشین مبدا تخصیص داده می‌شود و ماشین مجازی از حالت اجرای خود متوقف می‌شود و وضعیت پردازنده آن به ماشین مقصد انتقال پیدا می‌کند. سپس ماشین مجازی در ماشین مقصد شروع به کار می‌کند. این روند در شکل (۲-۱۸) مقابل مشاهده می‌باشد.

بعد از شروع به کار کردن ماشین مجازی در ماشین مقصد، به دلیل نبودن صفحات حافظه مورد نیاز برای پردازش و خالی بودن فضای حافظه، خطای صفحه رخ[۱] می‌دهد. در هنگام رخ دادن این خطا، صفحه حافظه مورد نظر از ماشین مبدا به ماشین مقصد انتقال می‌یابد. در کنار انتقال این صفحات، صفحات دیگر نیز از حافظه ماشین مبدا به ماشین مقصد انتقال پیدا می‌یابد. این روند تا زمانی انجام می‌شود که تمام فضای حافظه ماشین مجازی در ماشین مبدا به ماشین مقصد منتقل شود.

خطای صفحه در این در این روش به خطای شبکه[۲] نیز معروف است. منظور از خطای شبکه این است که صفحه یا صفحات حافظه مورد نظری که در فضای حافظه فعلی نیستند باید از طریق شبکه به ماشین مقصد انتقال پیدا کند.

در این روش زمان مهاجرت و مدت زمان کل مهاجرت نسبت به روش پیش‌کپی کم است. مدت زمان ازکارافتادگی ماشین مجازی نسبت به روش پیشین نیز کم است زیرا روند اجرایی آن در زمان کمتری بین دو ماشین مبدا و مقصد تعویض می‌شود. در ضمن، برنامه‌های در حال اجرا درون ماشین مجازی تا اتمام فرآیند مهاجرت تنزل پیدا می‌کند و کارایی آن‌ها کاهش می‌یابد زیرا حافظه کاربردی آن به طور کامل منتقل نشده است و تا رسیدن صفحهات مورد نیاز خود از فعالیت می‌ایستند.

این روش برای زمان‌هایی کاربرد دارد که بارکاری ماشین مجازی بیشتر از نوع بارکاری نوشتاری است زیرا فقط یک بار صفحات انتقال پیدا می‌کنند و در ماشین مبدا تغییر پیدا نمی‌کند و تغییرات بعدی آن‌ها در ماشین مقصد انجام می‌شود. همچنین در زمانی که پهنای باند شبکه بالا باشد این روش نسبت به روش پیشین اولویت پیدا می‌کند. از این رویکرد در زمانی که بخواهند ماشین مجازی را از چندین مرکز داده انتقال دهند، نیز کاربرد دارد.

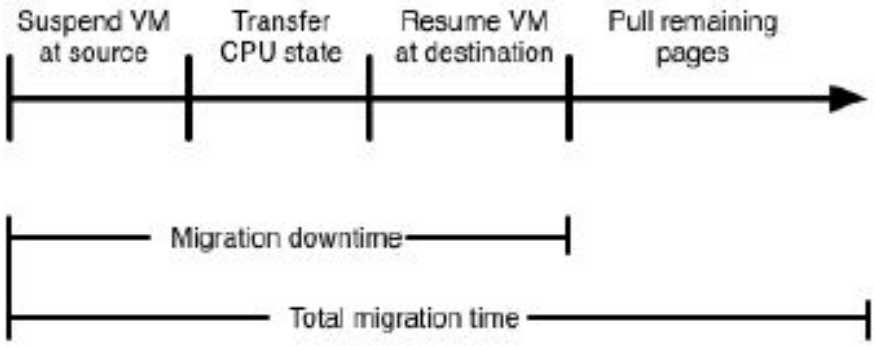
در روش پیش‌کپی، همیشه به‌روزترین وضعیت ماشین مجازی در زمان مهاجرت در ماشین مبدا قرار دارد، درحالی‌که در روش پس‌کپی، وضعیت ماشین مجازی در ماشین مبدا و مقصد توزیع‌شده است.

برای درک بهتر تفاوت زمانی ازکارافتادگی ماشین مجازی و مدت زمان مهاجرت آن در دو روش معرفی شده به شکل (۲-۱۹) مراجعه شود.

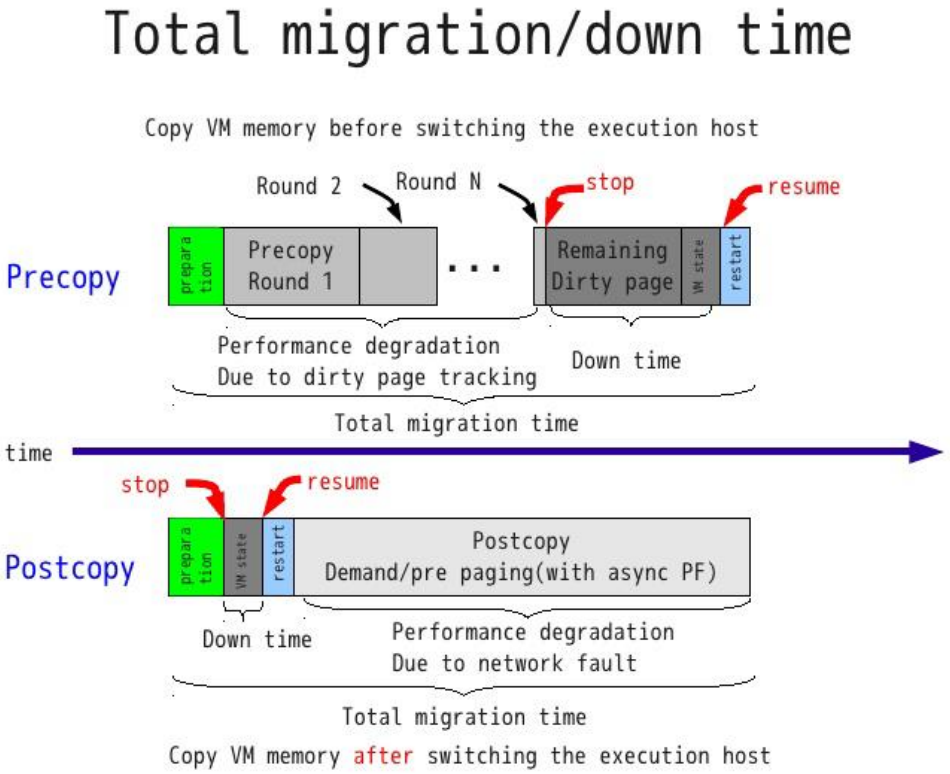
---

[۱] Page Fault

[۲] Network Fault



شکل (۲-۱۸) روش پس‌کپی [18]



شکل (۲-۱۹) تفاوت زمانی ازکارافتادگی ماشین مجازی در دو روش پیش‌کپی و پس‌کپی [18]

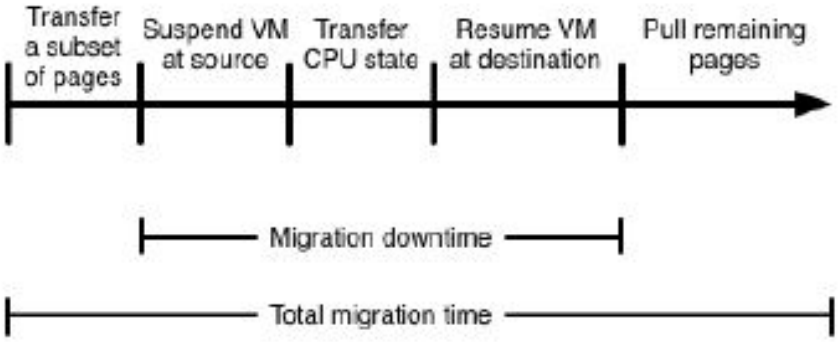
همان‌طور که مشاهده می‌شود هر دو روش مزایا و معایب خود را دارند. میزان خواندن زیاد از حافظه در روش پس‌کپی باعث افزایش خطاهای شبکه و تنزل برنامه‌های درون ماشین مجازی می‌شود و در روش پیش‌کپی میزان نوشتن روی حافظه می‌تواند زمان کل مهاجرت را به بی‌نهایت افزایش دهد. به همین دلیل است که روش ترکیبی[۱] ارائه شد.

[۱] Hybrid

- **روش ترکیبی**

در این روش، علاوه بر وضعیت پردازنده مقداری از حافظه که زیاد استفاده می‌شوند به ماشین مقصد ارسال می‌شوند. اگر این صفحات در زمانی که ماشین مجازی در ماشین مقصد شروع به فعالیت کند، وجود داشته باشد، خطاهای شبکه کاهش پیدا می‌کند.

این رویکرد مزایای هر دو روش قبلی را به نسبت دارد و می‌تواند نوع خاصی از روش پس‌کپی باشد. اگر این روش دقیق بهینه شود، می‌تواند جایگزین خوبی برای دو روش پیشین باشد و در سناریوهایی که دو رویکرد پیشین با شکست مواجه می‌شوند کارایی خوبی داشته باشد. روند کلی این روش در شکل (۲-۲۰) قابل‌مشاهده است.



شکل (۲-۲۰) روش مهاجرت ترکیبی [18]

۲-۳-۶- معیارهای کارایی

معمولا معیارهای زیر برای اندازه‌گیری کارایی مهاجرت زنده مورد استفاده قرار می‌گیرند [19, 20, 21]:

۱. زمان آمادگی[۱]: به زمانی گفته می‌شود که مهاجرت آغاز شده و وضعیت ماشین در حال انتقال به ماشین هدف می‌باشد. در این بازه، ماشین مجازی به کار خود ادامه می‌دهد و صفحات حافظه خود را تغییر می‌دهد.
۲. زمان ازکارافتادگی[۲]: زمانی است که ماشین مجازی متوقف می‌شود. در این حالت وضعیت پردازنده درحال انتقال است.

[۱] Prepration Time
[۲] Down Time

۳. زمان ازسرگیری[۱]: بازه‌ای است که ماشین مجازی کار خود را در ماشین هدف از سر می‌گیرد و عملیات مهاجرت به طور کامل تمام می‌شود. تمام وابستگی‌ها در ماشین مبدا نیز حذف شده است.
۴. زمان کل مهاجرت[۲]: به مجموع مدت زمان‌های بالا، از اول تا اتمام مهاجرت گفته می‌شود. این زمان یکی از مهم‌ترین معیارها است زیرا آزاد شدن منابع را در مبدا و مقصد مورد تاثیر قرار می‌دهد.
۵. تعداد صفحات انتقال‌یافته[۳]: به کل مقدار صفحات حافظه که در فرآیند مهاجرت جابجا می‌شود گفته می‌شود که صفحات تکراری را نیز در خود جای می‌دهد.
۶. مقدار کل داده انتقال‌یافته[۴]: به کل مقدار داده‌ای که جابجا می‌شود تا وضعیت ماشین مجازی همگام[۵] شود، گفته می‌شود.
۷. تنزل برنامه[۶]: به میزان افت کارایی برنامه در طول فرآیند مهاجرت گفته می‌شود.

از بین معیارهای معرفی شده، زمان کل مهاجرت، تعداد صفحات انتقال‌یافته و مدت ازکارافتادگی از معیار مهم تلقی می‌شوند.

۲-۳-۷- امنیت در مهاجرت

همانند معیارهای معرفی شده در مهاجرت، امنیت نیز یک معیاری است که در مراکز داده از اهمیت بالایی برخوردار است. میان‌افزارهایی که در مهاجرت دادن پردازه‌ها استفاده می‌شوند، از روش‌ها رمزنگاری در انتقال داده استفاده نمی‌کنند زیرا عمل رمزنگار سربار پردازشی و سربار شبکه را به همراه خود می‌آورند. امروزه با زیرساخت فعلی و توان بالای پردازش مسیریاب‌های[۷] موجود در سراسر دنیا می‌توان از سربار شبکه چشم‌پوشی کرد ولی در بسیاری از موارد و به دلیل افزایش زمان مهاجرت، ازکارافتادگی ماشین مجازی و تنزل برنامه کاربردی درون آن، از رمزنگاری ارتباط بین ماشین‌ها استفاده نمی‌کنند.

---

[۱] Resume Time
[۲] Total Migration Time
[۳] Pages Transferred
[۴] Total Data Transmitted
[۵] Sync
[۶] Application Degradation
[۷] Routers

امنیت دارای سه بخش اساسی است [22]:

۱. محرمانگی[۱]: محرمانه بودن تقریبا معادل با حریم خصوصی است. اقداماتی که در این زمینه انجام می‌شود برای این است که اطمینان حاصل کنند که اطلاعات را در دسترس افراد مجاز قرار دهند و از دسترسی افراد غیرمجاز به اطلاعات جلوگیری نمایند. یک مثال از اقداماتی که در این زمینه انجام می‌شود، درخواست نام کاربری و گذرواژه از کاربران است. در این روش تنها افرادی قادر به دسترسی به اطلاعات هستند که برای آن‌ها سطح دسترسی به اطلاعات تعریف‌شده باشد.
۲. دسترس‌پذیری[۲]: دسترس‌پذیری اطلاعت به حصول اطمینان از قابل‌دسترس بودن داده‌ها برای افرادی که مجاز به دسترسی به داده‌ها هستند گفته می‌شود. داده زمانی معنا پیدا می‌کنند که بتوان به آن دسترسی پیدا کرد. حملاتی هستند که این ویژگی را نقض می‌کنند مثل DoS[۳] که با هدف خارج کردن اطلاعات از دسترس انجام می‌شود و این جنبه از امینت را هدف قرار می‌دهد.
۳. انسجام[۴]: انسجام شامل حفظ هماهنگی[۵]، صحت[۶] و اطمینان از داده‌ها در طول چرخه زندگی آن است. داده‌ها نباید در انتقالشان تغییر کنند و همچنین نباید توسط افراد غیرمجاز تغییر کنند. از روش‌های مانند کنترل نسخه و یا پشتیبان‌گیری از داده‌ها می‌توان انسجام آن‌ها حفظ کرد.

در طول روند مهاجرت حملاتی وجود دارند که امنیت را در هر یک از جنبه‌های آن، به خطر می‌اندازند. برای مثال، می‌توان ماشینی که قرار است به عنوان میزبان یا ابرناظر جدید ماشین مجازی قرار بگیرد، از دسترس خارج کرد یا با یک ماشین میزبان دیگر جابجا کرد. زمانی که از دسترس خارج شود، مهاجرت صورت نمی‌گیرد و ممکن است مشکلاتی به وجود آورد. اگر مهاجرت به‌منظور تعمیر میزبان بخواهد انجام شود ولی به خاطر حملات DoS انجام نشود، ماشین‌های مجازی روی آن که درحال ارائه سرویس به کاربران هستند از کار می‌افتند زیرا میزبان نتوانسته آن‌ها را مهاجرت بدهد و خودش خراب شده است. حال اگر میزبان جدید به‌جای یک میزبان دیگر قرار گیرد، ماشین مجازی به ابرناظر غیرمجاز مهاجرت پیدا می‌کند و از اطلاعات رو آن یا پردازه‌های آن می‌توان سوءاستفاده کرد. همچنین می‌توان از

[۱] Confidentiality
[۲] Availability
[۳] Denial of Service
[۴] Integrity
[۵] Consistency
[۶] Accuracy

در طول مهاجرت ماشین مجازی-زمانی که صفحات حافظه جابجا می‌شوند- مقادیر آن‌ها را تغییر داد و یا صرفا آن‌ها را خواند و تغییر اعمال نشود. این نوع حملاتی که محتوای داده را تغییر می‌دهند می‌توانند باعث تغییر در روند سرویس ارائه‌شده توسط برنامه کاربردی درون ماشین مجازی بشوند و اگر تغییر روی داده‌های آن اعمال نشود، اطلاعاتی که نباید در دسترس دیگران قرار بگیرد، قابل‌دسترس می‌شود.

۲-۴- خلاصه

در بخش اول و دوم این فصل ابتدا به تعریف و بازگو کردن پردازه به عنوان اولین واحد اجرایی پرداخته شد و ویژگی و خصوصیت آن‌ها بیان شد. سپس به فناوری مجازی‌سازی و مفهوم ماشین مجازی به عنوان دومین نوع از واحد اجرایی پرداخته شد و به صورت کلی به معرفی ابرناظرها و مجازی‌سازهای معروف در دنیا و توضیح مختصر در روند مجازی‌سازی هر یک اشاره شده. در ادامه به مبحث کانتینرها که نوع سوم واحدهای اجرایی هستند، پرداخته شده و ویژگی‌های منحصربه‌فرد هر یک از انواع واحدهای اجرایی بیان شد.

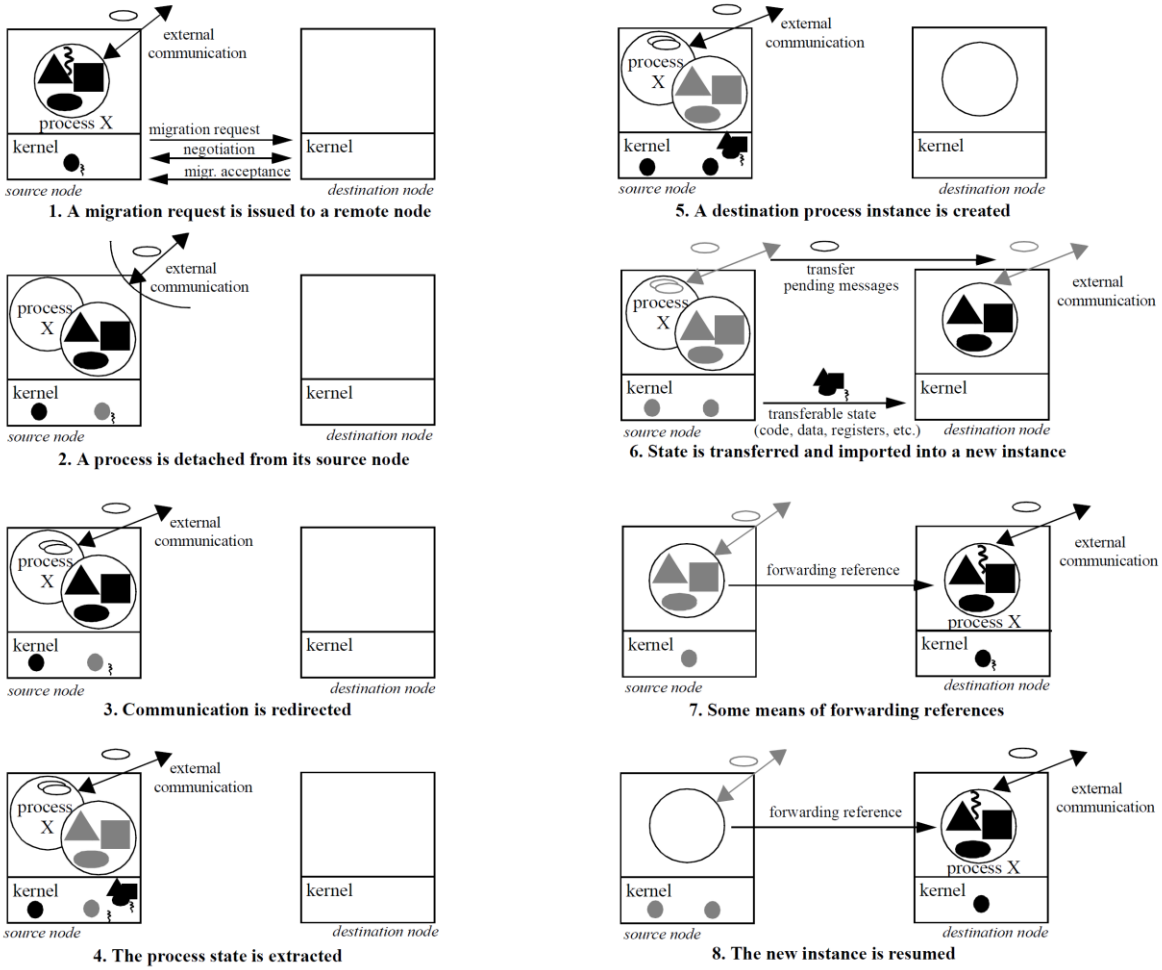
در بخش سوم این فصل به معرفی مهاجرت و بیان مفاهیم موجود در آن پرداخته شد. سپس انواع، مزایا و دلیل‌های استفاده از مهاجرت در مراکز داده بیان شد. در ادامه به معرفی کردن معیارهای کارایی هر یک پرداخته شد و در آخر، مسئله مهاجرت از دیدگاه امنیتی بررسی گردید.

فصل سوم: کارهای مرتبط

۳-۱- مقدمه

در این فصل پس از معرفی ادبیات و مفاهیم حوزه، به معرفی رویکردهای مختلف و کارهای مرتبطی که در این زمینه انجام شده است، پرداخته می‌شود. برای درک بهتر مهاجرت زنده، در ابتدا به مروری بر این روند پرداخته می‌شود.

روش‌های مختلفی برای مهاجرت زنده وجود دارد ولی در بیشتر آن‌ها قدم‌هایی مشابه انجام می‌شود که این قدم‌ها در شکل (۳-۱) قابل‌مشاهده می‌باشد.



شکل (۳-۱) نگاه کلی به قدم‌های مهاجرت زنده [17]

همان‌طور که مشاهده می‌شود، مهاجرت زنده در ۸ قدم خلاصه می‌گردد.

در قدم اول، ماشین مبدا با ماشین مقصد به مذاکره[۱] می‌پردازد و اگر نتیجه بر قبول کردن مهاجرت باشد، مهاجرت به گام بعدی می‌رود.

در قدم دوم، واحد اجرایی از روند اجرایی خود می‌ایستد و به طور موقت کانال‌های ارتباطی آن معلق[۲] می‌شود.

در قدم سوم، ارتباط پردازه با جمع شدن پیام‌های دریافتی به آن، به پردازه جدید بعد از مهاجرت تغییر مسیر[۳] می‌دهد و پردازه در طول مهاجرت نمی‌تواند پیامی را ارسال کند.

در قدم چهارم، وضیعت پردازه نظیر فضای حافظه، ثبات‌های پردازنده، وضعیت ارتباط (مثل فایل‌های باز شده و کانال‌های پیام) و فضای هسته سیستم‌عامل بیرون کشیده می‌شود[۴]. بعضی از مقادیر هسته سیستم‌عامل و کانال‌های پیام بعد از انتقال پردازه به ماشین دیگر در ماشین فعلی باقی می‌مانند.

در قدم پنجم، در سیستم‌عامل مقصد یک پردازه به وجود می‌آید تا حالت پردازه در آن انتقال یابد. پردازه جدید تا زمانی که داده‌های لازم آن به صورت کامل ارسال نشوند، شروع به کار نمی‌کند. پس از انتقال کامل وضعیت پردازه به ماشین جدید، پردازه به درجه یک پردازه عادی ارتقاء پیدا می‌کند.

در این قدم و قدم‌های بعدی از روش‌های مختلفی برای ارسال داده استفاده می‌کنند و برخی از آن‌ها به‌منظور افزایش امنیت، کانال ارتباطی را با روش‌های مختلفی نظیر IPSec رمز می‌کنند تا داده‌ها تغییر پیدا نکنند و انسجام آن‌ها حفظ شود.

در قدم ششم، وضعیت پردازه در داخل پردازه جدید کپی می‌شود، نیازی نیست که تمام وضعیت به فضای آدرس پردازه جدید وارد شود. برخی از روش‌ها وضعیت لازم آن‌ها را در زمان اجرا و برحسب نیاز ارسال می‌کنند.

در قدم هفتم، تمامی ارجاعات پردازه به منابعی مثل فایل‌ها، همچنان حفظ می‌شود. این روند برای کنترل کردن پردازه لازم است.

در قدم هشتم، زمانی که داده‌های لازم ارسال‌شده باشند، پردازه به روند اجرای خود بازمی‌گردد و مهاجرت کامل می‌شود و ممکن است داده‌های پیشین آن در ماشین مبدا حذف شود.

---

[۱] Negotiation

[۲] Suspended

[۳] Redirect

[۴] Extracted

مهاجرت زنده ماشین‌های مجازی فرآیندی است که در آن تمام سیستم‌عامل و برنامه‌های کاربردی وابسته به آن از یک ماشین فیزیکی به ماشین فیزیکی دیگری مهاجرت می‌کند. ماشین مجازی به صورت زنده و بدون اختلال در کار برنامه‌های در حال اجرا در آن، مهاجرت می‌کند. مزایای مهاجرت ماشین مجازی شامل: کم کردن مصرف انرژی ماشین فیزیکی، تعادل بار در میان ماشین‌های فیزیکی و تحمل-پذیری خدمت‌دهنده‌ها در برابر خطاهای ناگهانی، بالا بردن سطح امنیت می‌باشند.

کارهای انجام شده در مهاجرت زنده در حوزه‌های مختلفی انجام می‌شود که در ادامه به معرفی برخی از آن‌ها در حوزه‌های کارایی، مصرف انرژی، ابر، شبکه و امنیت پرداخته می‌شود.

۳-۲- حوزه کارایی

برخی از مطالعاتی که روی مهاجرت زنده انجام می‌شود، در حوزه کارایی مهاجرت صورت می‌گیرد. در این حوزه تلاش می‌شود که روند پاسخ‌دهی پردازه یا ماشین مجازی را در طول مهاجرت بهبود داده شود. تغییر روند پاسخ‌دهی برنامه کاربردی در زمان مهاجرت ماشین‌های مجازی امری غیرقابل‌اجتناب است. تاثیر مهاجرت را معمولا با ابزارهایی نظیر AB (Apache Bench Mark) و ابزارهایی که زمان پاسخ خدمت‌دهندگان وب را ارزیابی می‌کنند، اندازه‌گیری می‌شود.

مطالعه‌ای که در [23] انجام شده است روی تخمین پایداری صفحات حافظه در طول مهاجرت بوده است. محققان آن روشی ارائه دادند که می‌توان میزان پایداری صفحات حافظه ماشین مجازی را در طول مهاجرت تخمین بزنند. این تخمین به روند انتقال حافظه کمک می‌کند تا صفحاتی که احتمال تغییر در آن توسط ماشین مجازی زیاد است، در چرخه‌ی فعلی انتقال حافظه انتقال پیدا نکند. انتقال یک صفحه حافظه که قرار است در زمان انتقال آن تغییر پیدا کند، زمان کل مهاجرت را زیاد می‌کند و باعث می‌شود صفحه مورد نظر دوباره انتقال پیدا کند. ارزیابی انجام شده در این مطالعه نشان می‌دهد که برای ماشین‌های مجازی که حجم حافظ آن‌ها کمتر یا مساوی ۵۱۲ مگابایت باشد، ۱۰ تا ۱۶ درصد بهبود در انتقال حافظه حاصل می‌شود.

سان [24] در مطالعه‌ای روی مهاجرت زنده ماشین‌های مجازی بین مراکز داده داشته است. ایشان مطالعه‌ی خود را اول روی مهاجرت یک ماشین مجازی انجام داده سپس روشی بر اساس صف‌های مارکوف روی مهاجرت چندین ماشین مجازی به‌منظور بهبود بخشیدن به کارایی آن، ارائه داده است. در روش ارائه‌شده برای مهاجرت چندین ماشین مجازی، ابتدا از رویکرد پیش‌کپی برای انتقال ماشین مجازی

اول استفاده می‌کند و زمانی ماشین اول در زمان ازکارافتادگی خود قرار می‌گیرد، بقیه ماشین‌های مجازی با رویکرد پس‌کپی بر اساس موقعیت خود در صف M/M/C/C مهاجرت، شروع به مهاجرت می‌کنند. در این روش میزان انتقال حافظه نسبت به روش معمولی و مهاجرت هم‌زمان ماشین‌های مجازی، کمتر بوده و زمان ازکارافتادگی ماشین‌ها نیز کمتر می‌شود.

در پژوهش [25] میزان تاثیر مهاجرت زنده بر کارایی برنامه کاربردی در ماشین مجازی در ابرناظرهای Xen ارزیابی‌شده است. نتایجی که به دست آماده نشان‌دهنده آن است که در برخی از موارد سربار مهاجرت قابل‌تحمل است و نمی‌توان از آن گذشت، بخصوص در زمانی که در دسترس بودن و پاسخگویی توسط قراردادی که با کاربران دارند تضمین‌شده باشد. ارزیابی‌های انجام شده بر اساس بارکاریی برنامه-های است که در طبقه‌ی وب ۲ قرار می‌گیرند. در آزمایش‌های انجام شده از Olio که یک ایجادکننده بارکاری است، استفاده‌شده است و نشان داده شده است که در زمان کمی بعد از شروع شدن مهاجرت صدک‌های نودم و نود و نهم در توافق‌نامه سطح خدمات نقض می‌شود.

جین [26] در مطالعه خود روی روش پیش‌کپی در مهاجرت زنده، یک روش ارائه داده است که حافظه ماشین مجازی را در مبدا را فشرده‌سازی کرده و در ماشین مقصد آن را از حالت فشرده باز می‌کند. روش وی MECOM نامیده شده است. در روش پیش‌کپی به علت ارسال داده‌ها قبل از مهاجرت و تغییر آن‌ها توسط ماشین مجازی، میزان زیادی از باند شبکه را اشغال می‌کند و در شبکه‌هایی که عرض باند آن‌ها کم باشد سربار زیادی تولید کرده و کارایی برنامه کاربردی درون ماشین مجازی را پایین می‌آورد. در روش ارائه‌شده، سریع و پایدار بودن و تاثیر کم روی کارایی برنامه کاربردی در روند مهاجرت تضمین‌شده است. آزمایش‌های جین روی ابرناظر Xen بوده است و زمان ازکارافتادگی، زمان کل مهاجرت و میزان داده انتقال‌یافته را به ترتیب ٪۲۷,۱، ٪۳۲ و ٪۶۸,۸ بهبود بخشیده است. او با استفاده از بارهای کاری که روی نرم‌افزارهای کاربردی Apache، Tomcat، K-Compile، MUMmer و Dbench اعمال می‌شود، روند مهاجرت ماشین مجازی را ارزیابی کرده است.

در [20] تلاش شده است که تعداد چرخش را در روش پیش‌کپی مهاجرت زنده کاهش پیدا کند. این روش روی ابرناظر Xen نسخه ۳,۳,۰ پیاده‌سازی شده است. روش ارائه‌شده از بیت‌مپ برای پیدا مشخص کردن صفحات پر تغییر استفاده می‌کند و ارزیابی انجام شده نشان می‌دهد که تعداد چرخش‌ها بیشتر پنج بار نمی‌شود و مدت زمان کل مهاجرت ٪۳۲,۵ و میزان داده انتقال داده شده ٪۳۴ بهبودیافته است.

یو [27] در زمینه مهاجرت زنده کانتینرها روشی ارائه داده است که زمان ازکارافتادگی و مدت زمان

کل مهاجرت را بهبود بخشیده است. او داکر[1] را به عنوان میان‌افزار مهیاکننده کانتینرها برای آزمایش‌های خود انتخاب کرده است. داکرها فایل‌های خود را در قالب یک سری ایمیج که روی‌هم قرار می‌گیرند، ذخیره می‌کند. ایمیج‌های پایه‌ای وجود دارد که سیستم‌عامل‌های مختلف را شبیه‌سازی می‌کند و کتابخانه‌ها و برنامه‌های خاص آن را در خود جای می‌دهد. برنامه نویسان ایمیج پایه را دریافت می‌کنند و فایل‌های خود را درون یک لایه بالای آن قرار می‌دهند. سپس از لایه خود یک ایمیج می‌سازند و در مخزن‌های داکر انتقال می‌دهند.

روش یو شامل سه مرحله است. در مرحله اول ایمیج‌های کانتینر هدف را در ماشین مقصد کپی می‌کند. می‌توان در این مرحله بر اساس نیاز خود، این لایه‌ها را فشرده‌سازی کرد. این عمل زمان کل مهاجرت و سربار پردازشی را زیاد می‌کند ولی میزان داده انتقالی را کاهش می‌دهد. در مرحله دوم، رخ دادهایی که در کانتینر مبدا رخ می‌دهند، ثبت می‌شود و به کانتینر جدید ارسال می‌شود تا زمانی که میزان لاگ‌های ثبت‌شده به حد نساب خاصی برسد. در محله سوم، آخرین لاگ به کانتینر مقصد ارسال می‌شود و کانتینر در مبدا موفق می‌شود و کانتینر جدید ادامه کار خود را از سر می‌گیرد. ارزیابی انجام شده نشان داده است که روش وی در آزمون SPECweb99 توانسته مدت زمان کل مهاجرت را ٪۳۸ و مدت زمان ازکارافتادگی را ٪۴۴ بهبود بخشد.

۳-۳- حوزه مصرف انرژی

مصرف انرژی در مراکز داده از اهمیت بالا برخوردار است زیرا هر چه قدر که برق مصرف شود به همان نسبت انرژی صرف خنک کردن دستگاه‌های مصرف‌کننده آن می‌شود. معمولا در مراکز داده بزرگ ٪۵۸ هزینه‌ها صرف انرژی و خنک‌کننده‌ها می‌شود. این موضوع یکی از دلایل استفاده از مهاجرت ماشین‌های مجازی به‌خصوص مهاجرت زنده آن‌ها می‌باشد. در این حوزه تلاش می‌شود مهاجرت ماشین‌های مجازی را بر اساس میزان مصرف انرژی‌شان انجام دهند [28].

دیدگاه مصرفی انرژی محققان را به‌سوی تحقیق و فعالیت در زمینه‌های زیر سوق داده:

- معماری‌های سخت‌افزاری کارآمد ازنظر مصرف انرژی
- مجازی‌سازی منابع محاسباتی

[1] Docker

- زمان‌بندی کارها با در نظر گرفتن مصرف انرژی
- مقیاس‌پذیر کردن ولتاژ و فرکانس
- تجمیع کردن خدمت‌دهندگان[1]
- خاموش کردن گره‌های بدون استفاده

تمام متدهای فوق در برابر میزان کارایی برنامه کاربردی درون ماشین مجازی می‌باشد.

لیو [29] در مطالعه خود دو مدل برای اندازه‌گیری کارایی مهاجرت ماشین مجازی ارائه داده است. او نشان داد که پارامترهایی مثل اندازه حافظه، پهنای باند شبکه و نرخ تغییر صفحات حافظه تاثیر زیادی روی مدت زمان مهاجرت و ترافیک شبکه دارد. همچنین مدلی خطی برای ارزیابی میزان مصرف انرژی ارائه داده است و میزان دقت مدل خود را بالای ٪۹۰ ارزیابی کرده است.

در [30] آستانه مصرفی برای تجمیع خدمت‌دهنده‌ها معرفی شده است. در این روش ماشین‌های مجازی بر اساس میزان مصرف خدمت‌دهنده‌ها گروه‌بندی می‌شوند و از خدمت‌دهنده‌ای به خدمت‌دهنده دیگر مهاجرت داده می‌شود. این روش دارای دو مرحله است. در مرحله اول ماشین مجازی‌ای برای مهاجرت از یک خدمت‌دهنده با میزان مصرف منابع کم انتخاب می‌شود و در مرحله دوم اختصاص ماشین‌های مجازی را میان خدمت‌دهنده‌های فیزیکی بهینه می‌کند.

۳-۴- حوزه ابر

بخش دیگری از مطالعات انجام شده روی حوزه ابر می‌باشد. در این حوزه از دیدگاه کارایی روند مهاجرت را بین دو ابر بررسی می‌نمایند. واژه ابر به سیستمی گفته می‌شود که پنج ویژگی زیر را داشته باشد [31]:

۱) قابل‌دسترس از طریق اینترنت باشد. و کاربران بتوانند از هر جا و از طریق اینترنت به سرویس خود دسترسی پیدا کنند.
۲) کشسان[2] باشد و پاسخ نیاز کاربران برای استفاده از منابع بیشتر را زمان کوتاهی بدهد.
۳) مجموعه‌ای از منابع را در اختیار کاربران قرار دهد. منابع می‌تواند از نوع حافظه، دیسک و پردازنده و هر نوع منبعی دیگری که قابل‌استفاده کاربران است، باشد.

[1] Server Consolidation
[2] Elasticity

۴) بر اساس تقاضا باشد و کاربران بتوانند هر زمانی که نیاز به استفاده از منابع داشتند، آن‌ها را تقاضا کنند و به آن‌ها تخصیص داده شود.
۵) قابلیت اندازه‌گیری میزان استفاده‌شده از منابع را داشته باشد تا بتواند از کاربران هزینه استفاده دریافت کند.

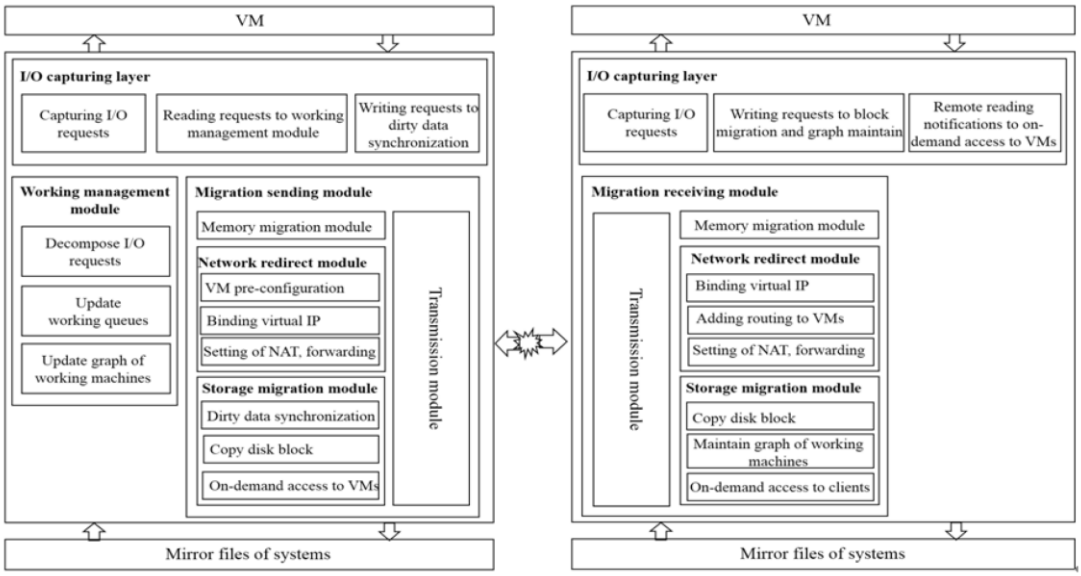
ابرها سه گونه اصلی دارند، ابر عمومی، ابر خصوصی و ابر ترکیبی. ابر عمومی ابری است که برای همه افراد قابل‌استفاده باشد و کاربران بتوانند درخواست بدهند و به آن‌ها سرویس داده شود. آمازون و آژور مثال‌هایی از ابر عمومی می‌باشند. ابر خصوصی ابری است که معمولا درون سازمان‌ها به وجود می‌آید و برای مصارف داخل سازمانی می‌باشد. ابر ترکیبی به ابری گفته می‌شود که هم برای مصارف عمومی کاربران و هم برای مصارف خصوصی درون‌سازمانی از آن استفاده می‌شود [32].

لی [32] در تحقیقات خود یک چهارچوب و میان‌افزاری را ارائه داد که با آن می‌توان ماشین‌هایی مجازی را بین ابرهای عمومی و خصوصی جابجا کرد. میان‌افزار ارائه‌شده از Qemu برای مهاجرت ماشین مجازی استفاده می‌کند. در این مطالعه میانگین زمان انتقال مهاجرت ماشین‌های مجازی در ابر خصوصی و ابر عمومی برای چهارچوب معرفی شده، اندازه‌گیری شده است و این اختلاف کمتر از یک ثانیه گزارش‌شده است. معماری استفاده‌شده در ابزار توسعه داده شده آن‌ها در شکل (۳-۲) آورده شده است و چهارچوب OEC[1] در شکل (۳-۳) قابل‌مشاهده می‌باشد.

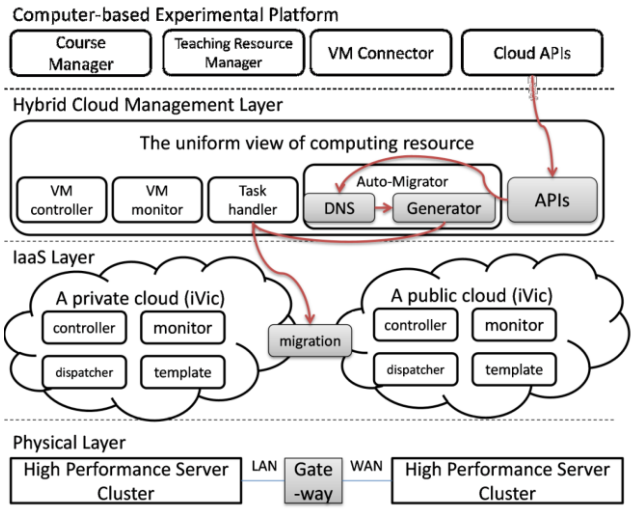
یکی دیگر مباحثی که در ابر مورد بررسی قرار می‌گیرد، بالا بردن بهره‌وری منابع و مدیریت تعادل بار در مراکز داده می‌باشد. مطالعه‌ای که در [33] انجام شده است، نشان داده است که با استفاده از روش پیش‌کپی تغییریافته می‌توان از باند شبکه به‌خوبی استفاده کرد و میزان کمتری از داده نسبت به حالت عادی انتقال داد. در روش پیش‌کپی، به صورت دوره‌ای روی صفحات حافظه چرخش صورت می‌گیرد و هر صفحه که مقدار آن تغییر کرده باشد و به ماشین مقصد ارسال‌شده باشد، دوباره ارسال می‌شود. در روش ارائه‌شده سه بیت‌مپ در نظر گرفته می‌شود که اولی نشان‌دهنده‌ی صفحات حافظه تغییریافته در چرخش قبلی می‌باشد، دومین بیت‌مپ نشان‌دهنده‌ی صفحات تغییریافته در چرخش فعلی و سومین بیت‌مپ مشخص‌کننده تعداد صفحات ارسال‌شده در چرخش قبلی می‌باشد. با داشتن این سه بیت‌مپ می‌توان تعداد صفحات انتقالی را نسبت به حالت معمولی کم کرد زیرا می‌توانیم تشخیص دهیم که کدام صفحه در چرخش بعدی احتمال تغییر دارد و نباید ارسال شود بنابراین، میزان داده ارسال‌شده کمتر می-

[1] Open Experimental Cloud

شود.



شکل (۳-۲) معماری استفاده‌شده برای مهاجرت ماشین مجازی در OEC [32]



شکل (۳-۳) چهارچوب OEC [32]

پاراون [34] روشی با استفاده از چهار بیت‌مپ را ارائه داده است که با آن می‌توان حجم داده انتقالی را کاهش داد. روش ارائه‌شده دو بخش دارد که در بخش اول تلاش می‌کند تمامی صفحات حافظه را که در چرخش قبلی تغییریافته‌اند و در این چرخه تغییر نکرده‌اند، فیلتر کند و در بخش دوم، صفحاتی که زیاد تغییر پیدا می‌کنند را در آخرین چرخش مهاجرت انتقال دهد. در بخش اول با استفاده از چهار بیت‌مپ خود که آن‌ها را to-send، to-skip، to-fix و to-send-last نام‌گذاری می‌کند، صفحات ارسالی چرخش

بعدی را مشخص می‌کند. To-send بیت‌مپی است که صفحات تغییریافته را در چرخش قبلی در خود نگه می‌دارد. To-skip بیت‌مپی است که صفحات تغییریافته را در چرخش فعلی در خود نگه می‌دارد. To-fix بیت‌مپی است که صفحات ارسالی را در آخرین چرخش مشخص می‌کند. To-send-last بیت‌مپی است که صفحات تغییریافته با نرخ بالا را مشخص می‌کند. با داشتن این مقادیر و مراجعه به جدول (۳-۱) می‌توان تشخیص داد که در چرخش فعلی کدام صفحه باید ارسال شود.

جدول (۳-۱) جدول نشان‌دهنده‌ی وضعیت صفحه برای ارسال در چرخه فعلی [34]

to-send	0	0	1	1
to-skip	0	1	0	1
ارسال شود؟	خیر	خیر	بله	خیر

در بخش بعدی، با استفاده از تاریخچه تغییر صفحات، صفحات ارسالی در آخرین چرخش را به دست می‌آورد. پاراون روش خود را در CloudSim شبیه‌سازی کرده و توانسته زمان ارسال هر ۱۰ صفحه را ۲۰ میلی‌ثانیه و زمان ازکارافتادگی را ۴۰ میلی‌ثانیه کاهش دهد.

۳-۵- حوزه امنیت

مجموعه مطالعاتی که در حوزه امنیت انجام می‌شود روی موضوعات نگه‌داشتن انسجام داده‌های ارسالی و جلوگیری از تغییرات در داده‌های حافظه، رمزنگاری کامل ارتباطی میان ماشین مبدا و مقصد، جلوگیری از حملات تحت شبکه مثل حمله مرد میانی[۱]، حملات DoS و سرریز شدن پشته[۲] است. تمامی رویکردها در این حوزه باعث افزایش زمان ازکارافتادگی ماشین مجازی و کاهش سطح سرویس آن‌ها به کاربران می‌شود ولی از حملات احتمالی به ماشین‌های در حال مهاجرت جلوگیری می‌کنند. به‌منظور استفاده از این روش‌ها می‌بایست به قراردادی که مراکز داده با مشتریان خود امضا می‌کنند توجه داشت. اگر در قراردادی میزان در دسترس بودن ماشین مجازی خیلی بالا (مثلا ۹۹,۹۹۹ درصد) در نظر گرفته شود، نمی‌توان از بیشتر روش‌هایی که باعث ازکارافتادگی بیشتر در طول مهاجرت هستند استفاده کرد.

[۱] Man in the Middle

[۲] Stack Overflow

عناصر بیشتری از دیدگاه امنیتی وجود دارد که به صورت خلاصه هر کدام با راه‌حل‌ها و چهارچوب[۱] مقابله با آن‌ها در جدول (۳-۲) آورده شده است [35].

آیش [36] با استفاده از چهارچوب CoM که خود بر اساس چهارچوب NSE-H[۲] است و از دیوار آتش و تکنیک‌ها IDS/IPS[۳] برای برقراری امنیت در شبکه استفاده می‌کند. توانسته است ماشین‌های مجازی را به صورت امن به ماشین مقصد مهاجرت دهد. همچنین در قدم بعدی از پروتکل vTPM برای تشخیص دادن انسجام ابرناظرهای مبدا و مقصد استفاده می‌کند تا بتواند ماشین مقصد را درستی‌یابی کند.

در [37] محققان کانال ارتباط داده‌ای بین ماشین‌های فیزیکی را با استفاده از پروتکل IPSec رمزنگاری کردند. استفاده از این روش زمان ازکارافتادگی را افزایش می‌دهد ولی با تغییر در MTU[۴] داده-ها شبکه توانستند زمان ازکارافتادگی را نسبت به حالت معمولی کمتر کنند و به هدف امن بودن کانال ارتباطی دست‌یابند.

در [38] محققان روشی را برای بهبود کارایی و کارآمدی[۵] مهاجرت ماشین مجازی ارائه دادند که این روش به صورت امن داده‌ها را انتقال می‌دهد. در این پژوهش از Xen به عنوان ابرناظر استفاده کردند و از روش ECC[۶] برای رمزنگاری کانال ارتباطی استفاده‌شده است. در این روش کلیدها، پیام‌ها و امضاها کوچک هستند و روند تولید کلیدها و امضاها سریع‌تر تولید می‌شوند. در آزمایش انجام شده زمان مهاجرت ۲۰۰ میلی‌ثانیه و زمان ازکارافتادگی ۱۱۹ میلی‌ثانیه کمتر از حالت معمولی بوده است.

---

[۱] Framework

[۲] Network Security Engine Hypervisor

[۳] Industrial Promotion Services / Intrusion Detection System

[۴] Maximum Tranfer Unit size

[۵] Efficiency

[۶] Elliptic Curve Cryptography

جدول (۳-۲) مقایسه عناصر مختلف امنیت [36]

عنصر امنیتی	توضیح	سناریو حمله	اقدامات لازم برای مقابله
سطح دسترسی	سامانه سطح دسترسی به منابع مختلف را اعمال کند	انتخاب سطح دسترسی نادرست می‌تواند موجب افزایش دسترسی افراد غیرمجاز به سامانه شود	جدول سطح دسترسی باید پیاده‌سازی شود تا در آن سطح دسترسی‌ها مشخص شود
احراز هویت	سامانه بتواند کاربر را شناسایی کند	قبل از شروع مهاجرت روند احراز هویت برای دو طرف ارتباط طی شود	در دیوار آتش باید اقداماتی برای اطمینان از احراز هویت پیاده‌سازی شود
عدم انکار	روش تضمین انتقال پیام از طریق امضای دیجیتال یا رمزنگاری	سامانه نتواند تمامی اقدامات کاربر را در گزارش‌گیری‌های خود ذخیره کند	تمامی اقدامات حساس و مهم باید در یک سیستم امن ذخیره شود
محرمانگی داده	جلوگیری از مطلع شدن افراد غیرمجاز از داده‌های محرمانه	پروتکل مهاجرت ماشین‌های مجازی نتواند داده‌های انتقالی را رمزنگاری کند و داده‌ها به صورت خام ارسال شوند	تمامی داده‌ها در طول مهاجرت باید با الگوریتم‌های رمزنگاری، رمز شوند
انسجام داده	داده‌ها حداقل در طول ارسالشان تغییر پیدا نکنند	حمله‌کننده می‌تواند قطعه کد مخرب را با استفاده از آسیب‌پذیری‌های موجود وارد ماشین مجازی کند	باید از نسخه‌ها و وصله‌های جدید برای نرم‌افزار مجازی‌ساز استفاده کرد و انسجام داده‌ها را با خلاصه‌گیری حفظ کرد
دسترس پذیری	داده‌ها باید در دسترس افزار مجاز باشند	حمله‌کننده می‌تواند مهاجرهای زیادی را به ماشین هدف شروع کند و سربار بیش از حد به وجود آورد	روش‌های فیلتر کردن پیام باید پیاده‌سازی شود

۳-۶- خلاصه

در این فصل از گزارش به بررسی مطالعاتی پرداخته شد که هدف آن‌ها بهبود روند مهاجرت زنده و چالش‌های روبه‌روی آن در حوزه‌هایی نظر کارایی، مصرف انرژی، ابر و امنیت هستند. بیشتر رویکردهای گفته‌شده بر اساس بهبود بخشیدن به روش پیش‌کپی بوده است و همچنین معیارهایی نظیر مدت زمان ازکارافتادگی و میزان تنزل برنامه کاربردی از اهمیت بالایی برخوردارند. حوزه کارایی به عنوان اولین حوزه پرکاربرد مورد بررسی قرار گرفت و پس‌ازآن به حوزه مصرف انرژی پرداخته شد. کارهای انجام شده در این حوزه در مراکز داده از اهمیت بالاتری برخوردار است زیرا به مصرف انرژی اهمیت بیشتری نسبت به میزان کارایی برنامه‌های کاربردی داده می‌شود. در ادامه به معرفی مطالعات روی حوزه‌های ابر که امروزه درحال گسترش است پرداخته شد و در آخر به حوزه امنیت که در مراکز داده نظامی بیشتر کاربرد دارد اشاره شد.

فصل چهارم: نتیجه‌گیری و کارهای آینده

۴-۱- نتیجه‌گیری

همان‌طور که در فصل‌های قبل بیان شد، امروزه مهاجرت واحدهای اجرایی یکی از مهم‌ترین مسئله‌های در مراکز داده و ابر محسوب می‌شود. در بین انواع واحدهای اجرایی، مهاجرت ماشین‌های مجازی نسبت به بقیه گونه‌ها دارای اهمیت بالایی است زیرا با ظهور مفهوم ابر و کاربردهای آن در دنیا و نیاز به بالا بردن بهره‌وری منابع مفهوم مهاجرت شکل‌گرفته و از آن به عنوان یک ابزار برای افزایش بهره‌وری استفاده می‌کنند. داشتن کم‌ترین زمان ازکارافتادگی و کم‌ترین زمان مهاجرت یکی از نیازهای اساسی در مهاجرت ماشین‌های مجازی است. راه‌کارهای که در این گزارش برای مهاجرت واحدهای اجرایی بررسی گردیدند، هریک با چالش‌هایی نظیر داشتن تاخیر در مهاجرت، افزایش سربار پردازشی، بالا بردن حجم داده‌های انتقالی، افزایش زمان ازکارافتادگی و تنزل برنامه کاربردی مواجه هستند. هر یک از روش‌های معرفی شده، تلاش در بهبود دادن یک یا چند معیار از معیارهای معرفی شده در زمینه مهاجرت، در حوزه‌هایی نظیر کارایی، مصرف انرژی، ابر و امنیت داشتند. در فصل ۳ این گزارش تلاش گردید تا به طور جامع، راه‌حل‌های نرم‌افزاری و سیستمی برای رفع چالش‌های مذکور معرفی گردد.

۴-۲- کارهای پیش‌رو

در سال‌های اخیر رشد چشم‌گیری در محاسبات ابری دیده شده است و در آن از فناوری مجازی‌سازی بسیار استفاده می‌شود تا بهره‌وری منابع را بالا ببرند. در این تکنولوژی و در برخی از مواقع نیاز است که ماشین‌های مجازی یا کانتینرها را از ماشینی به ماشین دیگر جهت افزایش بهره‌وری، جلوگیری از رخ دادن خطا در آن‌ها، مهاجرت دهند. امر مهاجرت توسط میان‌افزارهای زیادی روی ابرناظرهای مختلف انجام می‌شود. همچنین رویکردهای مختلفی برای مهاجرت ماشین‌های مجازی تاکنون ارائه‌شده است و هر یک از آن‌ها بسته نوع برنامه کاربردی درون ماشین مجازی متفاوت عمل می‌کنند.

در برخی از مراکز داده که به مصرف انرژی اهمیت بیشتری داده می‌شود، به الگوها و مدل‌هایی که بتوانند با آن میزان مصرف انرژی ماشین‌های مجازی را با در نظر گرفتن احتمال رخ دادن خطا در آن مشخص کنند، نیاز دارند و همراه مدل کردن رفتار مصرفی با چالش‌هایی مواجه بوده است. مدلی که رفتار ماشین مجازی را در زمان اجرای آن تحلیل کند و رویکرد مناسبی را برای مهاجرت آن در زمان‌های خاص انتخاب کند، می‌تواند در مصرف انرژی تاثیرگذار باشد.

در محاسبات ابری و محاسبات توان بالا، امر زمان‌بندی ماشین‌های مجازی و توزیع بارکاری در بین ماشین‌های فیزیکی مورد توجه قرار می‌گیرند و تلاش می‌شود تا بهره‌وری منابع سیستم افزایش یابد. یکی از روش‌هایی که در زمان اجرا برای بالا بردن بهره‌وری وجود دارد، مهاجرت زنده می‌باشد. اگر بتوان با ارائه روشی روند مهاجرت را نسبت به نوع بارکاری به‌گونه‌ای بهبود بخشید که کم‌ترین میزان سرباز پردازش و کمترین زمان مهاجرت را بین دو ماشین فیزیکی فراهم کند، تاثیر چشم‌گیری در قدرت پردازشی خواهد داشت.

یکی دیگر از معیارهایی که روی تنزل برنامه کاربردی تاثیر می‌گذارد، زمان ازکارافتادگی است. این معیار می‌تواند دارای اهمیت بیشتری نسبت به سربار شبکه و سربار پردازشی داشته باشد تا آنچه در توافق‌نامه سطح خدمات مشخص‌شده است را تضمین بدهد. به همین دلیل اگر بتوان روشی پیدا کرد که کمترین تاثیر را روی تنزل برنامه را با در نظر گرفتن معیارهای امنیت داشته باشد، کیفیت سطح خدمات ارائه‌شده را افزایش می‌دهد.

روند پاسخ‌دهی و زمان ازکارافتادگی در مهاجرت ماشین‌های مجازی از طریق شبکه‌های WAN که دارای پهنای باند کمی هستند، همواره چالشی است که روی کیفیت سرویس تاثیرگذار می‌باشد. علاوه بر زمان پاسخ، میزان داده انتقال داده شده نیز اهمیت پیدا می‌کند زیرا در شبکه‌های WAN هزینه انتقال اطلاعات نسبت به شبکه‌های درون مراکز داده بالاتر بوده است. اگر روندی پیدا شود که حجم داده‌های انتقالی را کم کند و تاثیرگذاری کمتری روی روند پاسخ‌دهی برنامه کاربردی داشته باشد، می‌تواند زمان ازکارافتادگی را کاهش دهد و هزینه‌های ارتباطی را کم کند.

با تغییر در روند رمزنگاری به‌گونه‌ای که سربار آن کاهش پیدا کند و انسجام داده حفظ شود، می‌توان اطمینان حاصل کرد که هیچ گونه تغییری در داده‌های ارسالی رخ نداده است و برنامه‌های کاربردی به درستی کار خواهند کرد. روش‌های فعلی سربار پردازشی و شبکه دارند. می‌توان این سربار را در شبکه-های داخلی و کوچک‌تر کرد و امنیت را بالا برد.

مراجع

[1] A. S. Tanenbaum, H. Bos, Modern Operating Systems, New Jersey: Pearson Education, 2015.

[2] S. Nanda and T. Chiueh, "A Survey on Virtualization Technologies," 2005.

[3] S. Campbell and M. Jeronimo, "Applied Virtualization Technology: Usage Models for IT Professionals and Software Developers," *Intel Press,* 2006.

[4] S. B. Rathod, and V. K. Reddy, "Secure Live VM Migration in Cloud Computing: A Survey," *International Journal of Computer Applications 103,* vol. 2, 2014.

[5] R. Uhlig, G. Neiger, D. Rodgers, A. L. Santoni, F. CM Martins, A. V. Anderson, S. M. Bennett, A. Kagi, F. H. Leung and L. Smith, "Intel Virtualization Technology," *Computer 38*, pp. 48-56, 2005.

[6] R. Buyya, J. Broberg and A. Goscinski, "CLOUD COMPUTING : Principles and Paradigms," 2011.

[7] M. Liaqat, S. Ninoriya, J. Shuja, R. Wasim Ahmad, and A. Gani, "Virtual Machine Migration Enabled Cloud Resource Management: A Challenging Task," 2016.

[8] M. Sudha, G. M. Harish, and J. Usha, "Performance Analysis of Linux Containers - An Alternative Approach to Virtual Machines," *International journal of Advance Research in Computer Science and Software Engineering*, vol. 4, no. 1, p. 820–824, 2014.

[9] J. E. Simons and J. Buell, "Virtualizing High Performance Computing," *ACM SIGOPS Operating Systems Review 44,* pp. 136-145, 2010.

[10] M. Rosenblum and T. Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends," *IEEE Internet Computing,* vol. 38, pp. 39-47, 2005.

[11] A. J. Younge, R. Henschel, J. T. Brown, G. V. Laszewski, J. Qiu and G. C. Fox, "Analysis of Virtualization Technologies for High Performance Computing Environments," *Cloud Computing (CLOUD), IEEE International Conference*, pp. 9-16, 2011.

[12] "Understanding Full Virtualization, Paravirtualization, and Hardware Assist," VMWare, 11 3 2008. [Online]. Available: https://www.vmware.com/techpapers/2007/understanding-full-virtualization-paravirtualizat-1008.html. [Accessed 25 9 2017].

[13] W. v. Hagen, Professional Xen Virtualization, Wrox, 2008.

[14] B. Zhang, X. Wang, R. Lai, L. Yang, Z. Wang, Y. Luo and X. Li, "Evaluating and optimizing I/O virtualization in kernel-based virtual machine (KVM)," *Network and Parallel Computing,* pp. 220-231, 2010.

[15] "Hyper-V Architecture," Microsoft, 2017. [Online]. Available: https://msdn.microsoft.com/en-us/library/cc768520(v=bts.10).aspx. [Accessed 19 6 2017].

[16] N. Kratzke, "A Lightweight Virtualization Cluster Reference Architecture Derived from Open Source Paas Platforms," *Open Journal of Mobile Computing and Cloud Computing 1*, vol. 2, pp. 17-30, 2014.

[17] D. S. Milojicic, F. Douglis, Y. Paindaveine, R. Wheeler and S. Zhou, "Process Migration," *Hp Labs, AT&T Labs-Research, EMC and University of Torento and Plateform Computing,* pp. 2-4, 1999.

[18] H. B. Arab, "Virtual Machines Live Migration," 2017.

[19] B. Jiang, J. Wu, X. Zhu and D. Hu, "Priority-Based Live Migration of Virtual Machine," *International Conference on Grid and Pervasive Computing,* pp. 376-385, 2013.

[20] F. Ma,F. Liu and Z. Liu, "Live Virtual Machine Migration Based on Improved pre-copy Approach," *In Software Engineering and Service Sciences (ICSESS), IEEE International Conference on,* pp. 230-233, 2010.

[21] D. Kapil, E. S. Pilli and R. C. Joshi, "Live Virtual Machine Migration Techniques: Survey and Research Challenges," *3rd IEEE International Advance Computing Conference,* pp. 964-967, 2013.

[22] J. Rao, D. Prasad, S. Rai, and B. Narain, "A Study of Network Attacks and Features of Secure Protocols," *Research Journal of Engineering and Technology 8*, vol. 1, pp. 4-8, 2017.

[23] K. Elghamrawy, Diana Franklin, and Frederic T. Chong, "Predicting Memory Page Stability and its Application to Memory Deduplication and Live Migration," *Performance Analysis of Systems and Software (ISPASS), IEEE International Symposium on,* pp. 125-126, 2017.

[24] G. Sun, D. Liao, V. Anand, D. Zhao, and H. Yu, "A New Technique for Efficient Live Migration of Multiple Virtual Machines," *Future Generation Computer Systems 55,* pp. 74-86, 2016.

[25] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation," *CloudCom,* pp. 254-265, 2009.

[26] H, Jin, L. Deng, S. Wua, X. Shi, H. Chen and X. Pan, "MECOM: Live Migration of Virtual Machines by Adaptively Compressing Memory Pages," *Future Generation Computer Systems,* pp. 23-35, 2014.

[27] C. Yu and F. Huan, "Live Migration of Docker Containers through Logging and Replay," *3rd International Conference on Mechatronics and Industrial Informatics,* pp. 623-626, 2015.

[28] J. Sekhar, G. Jeba and S. Durga, "A Survey on Energy Efficient Server Consolidation Through VM Live Migration," *International Journal of Advances in Engineering & Technology,* pp. 1-4, 2012.

[29] H. Liu, C. Xu, Hai Jin, J. Gong and X. Liao, "Performance and Energy Modeling for Live Migration of Virtual Machines," *HPDC,* pp. 249-264, 2011.

[30] A. Beloglazov, J. Abawajy and R. Buyya, "Energy-aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing," *Future generation computer systems 28,* pp. 755-768, 2012.

[31] "Final Version of NIST Cloud Computing Definition Published," NIST, 25 9 2011. [Online]. Available: https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published. [Accessed 12 10 2017].

[32] Y. LI, B. LI and J. NI, "OEC: An Open Experimental Cloud Based on VM

Migration," *DEStech Transactions on Social Science, Education and Human Science,* pp. 83-87, 2017.

[33] S. Ambika, R. Ezhilarasie, and A. Umamakeswari, "A Survey on Live Migration Techniques of Virtual Machines," *Indian Journal of Science and Technology 9,* 2016.

[34] P. Jain, and R. Agrawal, "An Improved Pre-copy Approach for Transferring the VM Data during the Virtual Machine Migration for the Cloud Environment," 2016.

[35] B. Lakshmipriya, R. Leena Sri and N. Balaji, "A Novel Approach for Performance and Security Enhancement during Live Migration," *ndian Journal of Science and Technology,* 2016.

[36] M. Aiash, G. Mapp and O. Gemikonakli, "Secure Live Virtual Machines Migration: Issues and Solutions," *Advanced Information Networking and Applications Workshops (WAINA), 28th International Conference, IEEE,* pp. 160-165, 2014.

[37] N. Azman Sulaiman and H. Masuda, "Evaluation of A Secure Live Migration of Virtual Machines Using IPsec Implementation," *Advanced Applied Informatics (IIAIAAI) IIAI 3rd International Conference,* pp. 687-693, 2014.

[38] R. Divyambika, and A. Umamakeswari, "Protection of Virtual Machines During Live Migration in Cloud Environment," *Indian Journal of Science and Technology 8,* pp. 333-339, 2015.

پیوست

پیوست الف: واژه‌نامه فارسی به انگلیسی

کلمه فارسی	معادل به انگلیسی
افزونه‌ای	**Extension**
انتزاع	**Abstraction**
انسجام	**Integrity**
آستانه	**Threshold**
بیرون کشیده می‌شود	**Extracted**
پایدار	**Persistance**
پس‌کپی	**Post-Copy**
پهنای باند	**Band Width**
پیش‌کپی	**Pre-Copy**
تجمیع کردن خدمت‌دهندگان	**Server Consolidation**
ترجمه دودویی	**Binary Translation**
ترکیبی	**Hybrid**
تعداد صفحات انتقال‌یافته	**Pages Transferred**
تعلقات	**Dependency**
تغییر مسیر	**Redirect**
تکراری	**Iterative**
تله	**Trap**
تنزل برنامه	**Application Degradation**
ثبات	**Register**
چارچوب	**Framework**
خدمت‌دهنده	**Server**
خطای شبکه	**Network Fault**
خطای صفحه	**Page Fault**
دسترس‌پذیری	**Availability**
راه‌انداز	**Driver**
روش	**Method**

کلمه فارسی	معادل به انگلیسی
زمان ازسرگیری	**Resume Time**
زمان آمادگی	**Preparation Time**
زمان کل مهاجرت	**Total Migration Time**
زمان‌بند	**Scheduler**
سازوکار	**Mechanism**
سامانه	**System**
سبک‌وزن	**Light Weight**
سرریز شدن پشته	**Stack Overflow**
شیار	**Tracks**
صحت	**Accuracy**
فراخوانی ابرناظر	**Hypercall**
فراخوانی سیستمی	**System Call**
قسمت‌بندی	**Partitioning**
قطاع	**Sector**
کارایی	**Performance**
کارآمدی	**Efficiency**
کانال‌های پیام	**Message Tunnels**
کانتینر	**Container**
کشسان	**Elasticity**
ماژول هسته	**Kernel Module**
متمرکزسازی	**Consolidation**
مجازی‌سازی جزئی	**Paravirtualization**
مجازی‌سازی کامل	**Full Virtualization**
مجزا	**Isolated**
محرمانگی	**Confidentiality**
محل ذخیره‌سازی	**Storage**
مدت زمان ازکارافتادگی	**Down Time**
مذاکره	**Negotiation**

کلمه فارسی	معادل به انگلیسی
مرد میانی	**Man in the Middle**
مسیریاب	**Router**
معلق شدن	**Suspended**
مقدار کل داده انتقال‌یافته	**Total Data Transmitted**
مهاجرت ایستا	**Cold Migration**
مهاجرت زنده	**Live Migration**
میان‌افزار	**Middleware**
ناظر - ابرناظر	**Hypervisor**
نقش	**Role**
هسته	**Kernel**
هماهنگی	**Consistency**
همگام	**Sync**
واحد اجرایی	**Processing Units**
وصله	**Patch**
وضعیت ارتباطی	**Communication State**
وضعیت ریشه	**Root Mode**
وضعیت غیر ریشه	**Non-root Mode**

پیوست ب: واژه‌نامه انگلیسی به فارسی

کلمه انگلیسی	معادل به فارسی
Abstraction	انتزاع
Accuracy	صحت
Application Degradation	تنزل برنامه
Availability	دسترس‌پذیری
Band Width	پهنای باند
Binary Translation	ترجمه دودویی
Cold Migration	مهاجرت ایستا
Communication State	وضعیت ارتباطی
Confidentiality	محرمانگی
Consistency	هماهنگی
Consolidation	متمرکزسازی
Container	کانتینر
Dependency	تعلقات
Down Time	مدت زمان ازکارافتادگی
Driver	راه‌انداز
Efficiency	کارآمدی
Elasticity	کشسان
Extension	افزونه‌ای
Extracted	بیرون کشیده می‌شود
Framework	چارچوب
Full Virtualization	مجازی‌سازی کامل
Hybrid	ترکیبی
Hypercall	فراخوانی ابرناظر
Hypervisor	ناظر - ابرناظر
Integrity	انسجام
Isolated	مجزا

کلمه انگلیسی	معادل به فارسی
Iterative	تکراری
Kernel	هسته
Kernel Module	ماژول هسته
Light Weight	سبک‌وزن
Live Migration	مهاجرت زنده
Man in the Middle	مرد میانی
Mechanism	سازوکار
Message Tunnels	کانال‌های پیام
Method	روش
Middleware	میان‌افزار
Negotiation	مذاکره
Network Fault	خطای شبکه
Non-root Mode	وضعیت غیر ریشه
Page Fault	خطای صفحه
Pages Transferred	تعداد صفحات انتقال‌یافته
Paravirtualization	مجازی‌سازی جزئی
Partitioning	قسمت‌بندی
Patch	وصله
Performance	کارایی
Persistance	پایدار
Post-Copy	پس‌کپی
Pre-Copy	پیش‌کپی
Preparation Time	زمان آمادگی
Processing Units	واحد اجرایی
Redirect	تغییر مسیر
Register	ثبات
Resume Time	زمان ازسرگیری
Role	نقش

کلمه انگلیسی	معادل به فارسی
Root Mode	وضعیت ریشه
Router	مسیریاب
Scheduler	زمان‌بند
Sector	قطاع
Server	خدمت‌دهنده
Server Consolidation	تجمیع کردن خدمت‌دهندگان
Stack Overflow	سرریز شدن پشته
Storage	محل ذخیره‌سازی
Suspended	معلق شدن
Sync	همگام
System	سامانه
System Call	فراخوانی سیستمی
Threshold	آستانه
Total Data Transmitted	مقدار کل داده انتقال‌یافته
Total Migration Time	زمان کل مهاجرت
Tracks	شیار
Trap	تله

Abstract

With the advent of virtualization technology and containers, processes are no longer the only execution units in the operating systems. Using virtualization technology, virtual machines (VMs) can be introduced to the operating system as a single or multiple processes and acquire resources. With the rise of the cloud computing and high performance computing, response time and efficient usage of resources have become very important. These fields use virtualization, and to increase the utilization of physical resources like processors, memory or I/O, they rely heavily on live migrations. Live migration is a procedure in which a virtual machine is moved from one hypervisor to another without pausing it. This procedure always degrades the response time and the efficiency of the applications running in the VM, and suffers from computational and network overheads. There have been many studies on VM live migrations seeking to reduce response time and increase efficiency and were usually able to improve one or more metrics related to the procedure. This report tries to review the problems and solutions with their benefits and disadvantage for the live migration mechanisms.

Keywords: Execution Units, Virtualization, Virtual Machine, Live Migration.

Iran University of Science and Technology
School of Computer Engineering

A Study of The Challenges and Solutions to Live Migration of Execution Units

A Master of Science Degree Seminar Report
In Computer Engineering, Software Discipline
(Computer Systems Concentration)

By:
Hooman Behnejad Fard

Supervisor:
Dr. Mohsen Sharifi

October - 2017