مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (41) العدد (2) 2019
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (41) No. (2) 2019

تحسين الزمن الكلي للهجرة الحية في مراكز البيانات الافتراضية

الدكتور أحمد صقر أحمد*

حيدر خليل**

(تاريخ الإيداع 11 / 2 / 2019. قُبِل للنشر في 23/ 4 / 2019)

□ ملخّص □

تعتبر الهجرة الحيَة (live migration) من أهم السمات التي تقدمها البيئات الإفتراضية ، وتعرَف على أنها نقل الآلة الإفتراضية من مخدم فيزيائي الى مخدم اخر دون انقطاع لخدمة التطبيقات التي تعمل عليها.تعتبر خوارزمية النسخ المسبق (pre–copy) من أهم الخوارزميات التي تنفذ الهجرة الحية ،حيث يعتمد مبدأ عمل هذه الخوارزمية على النقل المتكرر لصفحات الذاكرة المتغيرة (dirty page) أثناء تطبيق الهجرة الحية بين المصدر و الهدف، بحيث يتوقف التكرار عند عتبة معينة . إن التغير الكبير في صفحات الذاكرة سوف يزيد من عدد الصفحات المرسلة عبر الشبكة مما يزيد من حجم الذاكرة الكلي المنقول، وبالتالي سيزداد معدل استهلاك الشبكة و الزمن الكلي للهجرة و زمن التوقف ، وهي المعايير الأساسية لتقييم الأداء أثناء تنفيذ الهجرة الحية.

لذلك كان الغرض من هذا البحث تحسين الزمن الكلي للهجرة و تحسين انتاجية الشبكة (throughput) من خلال دراسة تأثير تغير نمط البيئة الإفتراضية على الهجرة الحية انطلاقا من أن مدير الالة الإفتراضية (hypervisor)هو المسؤول الرئيسي عن عملية تنفيذ الهجرة الحية و إدارة و تخصيص الموارد و طرق الاتصال بين الالة الإفتراضية و الموارد المادية للعتاد الصلب ، حيث قدمنا نموذج عمل يكون فيه ال hypervisor من نمط البيئة الإفتراضية الجزئي مع خوارزمية النسخ المسبق لتحسين الزمن الكلي للهجرة الحية في مراكز البيانات الإفتراضية ، ومقارنة النتائج مع النموذج الأساسي المتبع ، وهو تنفيذ خوارزمية النسخ المسبق مع نمط البيئة الإفتراضية الكامل .

قمنا بتطبيق النموذجين باستخدام نظام التشغيل Centos 7 و hypervisor من نوع XEN و لتحليل النتائج استخدمنا محلل الأداء NMON .أظهرت النتائج أن النموذج المقترح قد حسن من الزمن الكلي للهجرة الحية كما حسن من معدل استهلاك الشبكة ،ولكن لاحظنا زيادة في معدل استهلاك المعالج.

الكلمات المفتاحية : الهجرة الحية ،الآلة الافتراضية، مدير الآلة الإفتراضية .

* أستاذ – قسم النظم والشبكات الحاسوبية – كلية الهندسة المعلوماتية – جامعة تشرين – اللاذقية – سورية.

** طالب دراسات عليا(دكتوراه)–قسم النظم والشبكات الحاسوبية–كلية الهندسة المعلوماتية–جامعة تشرين– اللاذقية – سورية.

e-mail: drahmad1961@gmail.com ، haiderkh1987@gmail.com

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (41) العدد (2) 2019
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (41) No. (2) 2019

Improve the total migration time of live migration in virtualization data centers

Dr. Ahmad Saker Ahmad*
Haider Khalil**

(Received 11 / 2 / 2019. Accepted 23 / 4 / 2019)

□ ABSTRACT □

Live migration is one of the most important features offered by virtual environments. It is defined as the transfer of the virtual machine from one physical server to another server without interrupting the service of the applications on which it operates. The pre-copy algorithm is one of the most important algorithms that implement migration Where the principle of this algorithm depends on the repeated transfer of the dirty page during the live migration between the source and the target, so that the frequency stops at a certain threshold. The large change in the memory pages will increase the number of pages sent over the network, increasing the total amount of transmitted memory, thus increasing the network consumption rate, total migration time
and downtime, which are the basic criteria for evaluating performance during the implementation of live migration .
Therefore, the purpose of this research was to improve the total migration time and average network consumption by examining the effect of changing the pattern of the virtualization on live migration, since the virtual hypervisor is responsible for the implementation of live migration, resource management, Between the virtual machine and the physical resources of the hardware. presented a model in which the hypervisor of the paravirtualization environment with the pre-copying algorithm to improve the total time of live migration in the virtual data centers and compare the results with the
basic model followed, the implementation of the pre-copy algorithm with the fullvirtualization environment pattern.
We applied the two models using the Centos operating system and the XEN hypervisor. To analyze the results we used the NMON performance analyzer. The results showed that the proposed model improved the total migration time and improved network consumption, but we noticed an increase in CPU consumption.

Keywords: live migration, virtual machine ,hypervisor .

***Professor, Department of System and Computing Network Engineering , Faculty of Informatics, Engineering, Tishreen University, Lattakia, Syria.**
****Postgraduate Student, Department of System and Computing Network Engineering , Faculty of Informatics, Engineering, Tishreen University, Lattakia, Syria.**
e-mail:. drahmad1961@gmail.com ، haiderkh1987@gmail.com

مقدمة

تتطور الحوسبة السحابية بشكل سريع وذلك لما تقدمه من مرونة في الحصول على الخدمات المختلفة و حرية التوسع و توفير الطاقة و المال ، الأمر الذي جعلها هدفا للبحث و السعي الدائم من قبل الباحثين لتطوير مكوناتها بهدف الحصول على أفضل أداء ممكن. تعد البيئات الإفتراضية (virtualization) البنية الأساسية و المكون الأهم من مكونات الحوسبة السحابية و ذلك لما تقدمه من الاستخدام الأمثل للموارد من خلال تحميل أنظمة تشغيل متعددة على جهاز مادي واحد مع إمكانية العزل الكامل بين هذه الأنظمة لتعمل بشكل مستقل. تقوم الحوسبة السحابية بتوزيع مهام و خدمات الحوسبة على مجموعة كبيرة من الآلات الإفتراضية (virtual machine). عندما يزداد الحمل على أحد المخدمات الفيزيائية نتيجة تزايد الطلب على الخدمات المنصبة على آلته الإفتراضية يجب نقل عدد معين من آلاته الإفتراضية إلى مخدم فيزيائي آخر دون انقطاع الخدمة التي تقدمها التطبيقات على هذه الآلات. إن عملية نقل الآلة الإفتراضية VM من مخدم الى اخر دون انقطاع خدمات التطبيقات التي تعمل عليها تسمى بالهجرة الحية للآلة الإفتراضية . في الماضي كانت عملية نقل الآلة الإفتراضية بين مخدمين تتطلب إيقاف الآلة الإفتراضية على المخدم المصدر ومن ثم حجز الموارد من ذاكرة و وحدة معالجة على المخدم الهدف و بعد ذلك يتم النقل.

إن ترحيل الآلة الإفتراضية من مخدم إلى أخر يتضمن نقل صفحات الذاكرة و حالة المعالج و مجاري الدخل و الخرج و طلبات المستخدم من المخدم المصدر الى المخدم الهدف[1] .تعتبر خوارزمية النسخ المسبق من أول و أهم الخوارزميات المستخدمة لتنفيذ الهجرة الحية، تعتمد هذه الخوارزمية في مبدأ عملها على تكرار نقل صفحات الذاكرة المتغيرة (dirty page) بين المصدر و الهدف بحيث يتوقف التكرار عند عتبة معينة ،[2] بالتالي إن التغير الكبير في صفحات الذاكرة سوف يزيد من عدد الصفحات المرسلة عبر الشبكة مما يزيد من حجم الذاكرة الكلي و بالتالي يزداد الزمن الكلي للهجرة و حجم البيانات الإضافية المرسلة عبر الشبكة مما يزيد من معدل استهلاكها.

أهمية البحث و أهدافه :

تعتبر الهجرة الحية من أهم الميزات التي تقدمها البيئات الإفتراضية في مراكز البيانات الإفتراضية ، وذلك لما تقدمه من ميزات لمدراء هذه المراكز من مرونة في عمليات الصيانة و موازنة الحمل على المخدمات و الحفاظ على البيانات من الكوارث عن طريق ترحيل الآلة الإفتراضية الى اماكن أمنة ، لذلك كان الهدف من هذا البحث تحسين الزمن الكلي للهجرة الحية و تقليل الحمل على الشبكة أثناء تطبيق الهجرة الحية من خلال تقديم نموذج عمل يعتمد على نمط البيئة الإفتراضية الجزئي مع خوارزمية النسخ المسبق لتنفيذ الهجرة الحية .

طرائق البحث و مواده :

قدمنا نموذج عمل يكون فيه ال hypervisor من نمط البيئة الإفتراضية الجزئي مع خوارزمية النسخ المسبق لتحسين زمن الهجرة الحية و زمن انقطاع الخدمة في مراكز البيانات الإفتراضية وسمينا هذا النموذج Paravirtual_pre_copy، ومقارنة النتائج مع النموذج الأساسي المتبع وهو تنفيذ خوارزمية النسخ المسبق مع نمط البيئة الإفتراضية الكامل و سمينا هذا النموذج Fullvirtual_pre_copy . الشكل(1) يوضح البيئة العملية للتجربة ، حيث استخدمنا في تجربتنا ثلاث أجهزة مادية كمخدمات فيزيائية كل جهاز مزود بمعالج core i3 و ذواكر 4 غيغابايت . تم تنصيب نسخة centos 7 على كلا المخدمين و تنصيب البيئة الإفتراضية Xen لتعمل كـ

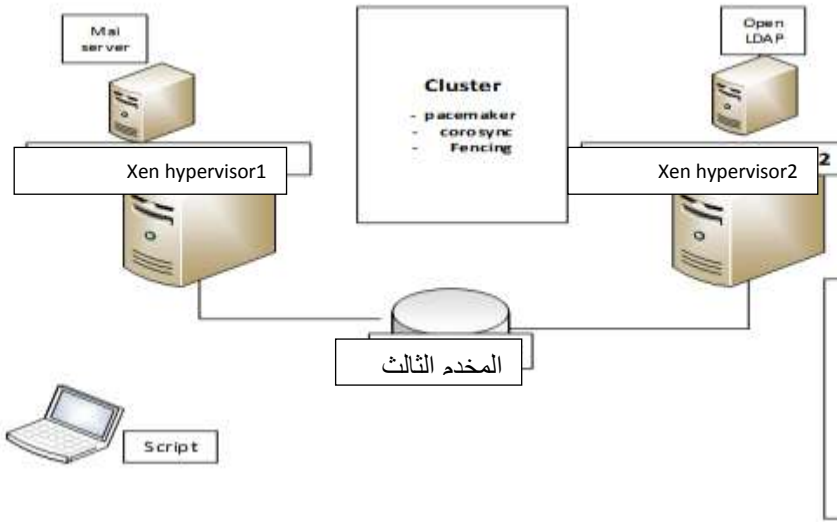
hypervisor من النمطين الكلي و الجزئي على كلا المخدمين و استخدام المخدم الثالث كجهاز تخزين مشترك للقرص الإفتراضي الخاص بالآلات الإفتراضية ، كما وضعنا كلا المخدمين في عنقود. قمنا بتجهيز آلة افتراضية بحجم ذاكرة 2 غيغابايت على أحد المخدمين تقدم خدمة البريد الإلكتروني و طبقنا الهجرة الحية على هذه الآلة و قمنا بترحيلها إلى المخدم الآخر . طبقنا حمل على الآلة الإفتراضية أثناء ترحيلها من خلال تغيير عدد الزبائن طالبي خدمة البريد الإلكتروني وذلك بهدف إجراء تغيير على صفحات الذاكرة أثناء الهجرة الحية حيث أن تغيير صفحات الذاكرة أثناء نقل الآلة الإفتراضية يعتبر من أهم العوامل في زيادة الزمن الكلي للهجرة ، كما قمنا بتغيير حجم الملف المرسل وذلك بهدف إشغال الشبكة أثناء تطبيق الهجرة الحية. قمنا بتشغيل محلل الأداء nmon على المخدم المصدر للحصول على النتائج و تحليليها .

- تم كتابة script يقوم بإرسال رسائل الكترونية وفق الكود التالي :

```
For I in  to n do
For j in  to n do
echo . | mail -r test$i.user@Xenlab.local -a file -s "This is test" test$j.user@Xenlab.loc
```

حيث أن :n عدد المستخدمين و file الملف متغير الحجم ، و هي قيم يتم تغييرها للحصول على تجارب متعددة .

فمثلا من أجل n=8 و حجم ملف 6 ميغابايت سيكون عدد الرسائل الإلكترونية على الشبكة 64 (8 *8) رسالة بحسب حلقة for المتداخلة و حجم البيانات المنقولة على الشبكة هو (64*6) 384 ميغابايت .



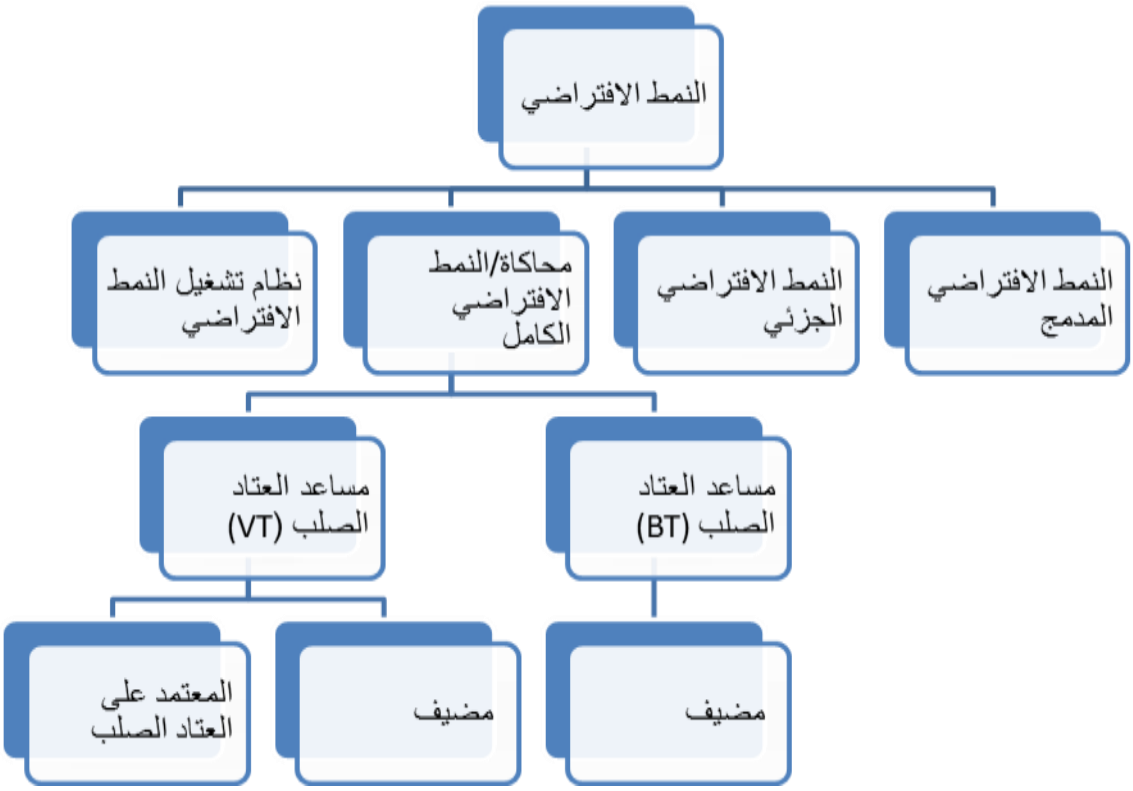
الشكل (1) بيئة عمل التجربة

4- البيئة الإفتراضية (Virtualization) :

هي تقنية تهدف بشكل أساسي إلى تقسيم موارد الحاسب المادية من معالج و ذاكرة، بحيث يمكن تشغيل أكثر من نظام تشغيل على العتاد المادي نفسه، حيث يعمل كل نظام تشغيل على آلة افتراضية VM لها ذاكرة افتراضية و وحدة معالجة افتراضية حسب حاجة نظام تشغيل الضيف الموجود عليها. تتعامل التطبيقات الموجودة على أنظمة تشغيل الضيف مع الموارد المادية الحقيقية من خلال مدير الآلة الإفتراضية (Hypervisor) وهو برمجية مثبتة على الجهاز المضيف للآلة الإفتراضية ، و المسؤول عن تشكيل طبقة مجردة افتراضية بين التطبيقات و الموارد المادية .

يقوم ال hypervisor بتخصيص الموارد المادية افتراضيا من وحدة المعالجة المركزية والذاكرة والشبكة و أجهزة التخزين لكل نظام تشغيل افتراضي أو لكل تطبيق يعمل على نظام تشغيل افتراضي ، و يعالج اتصالات النظام الافتراضي الضيف مع العتاد المادي الفيزيائي.هناك أنواع مختلفة للمحاكاة الافتراضية تعتمد على مستوى العزل بين الآلات الإفتراضية . الشكل (2) يبين الأنواع المختلفة للبيئات الإفتراضية.

سيتم التركيز في هذا البحث على النمط الإفتراضي الكامل (Full Virtualization) و النمط الافتراضي الجزئي (Paravirtualization)، و دراسة تأثير النمط الافتراضي الجزئي مع خوارزمية النسخ المسبق على الهجرة الحية للآلة الإفتراضية.

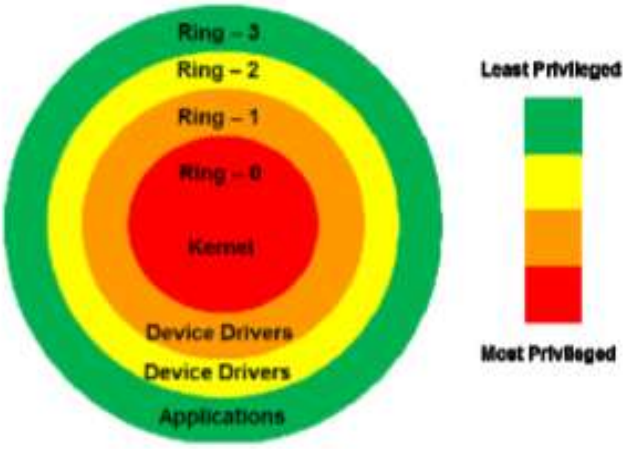


الشكل(2) أنواع البيئات الإفتراضية

5- البيئة الإفتراضية و حلقات الحماية (Virtualization and Protection Rings) :

تعتبر بنية حلقات الحماية آلية رسمية لفصل نظام التشغيل الموثوق عن برامج المستخدم غير الموثوق بها ،حيث تتيح هذه الحلقات مستويات مختلفة من العزل و الامتياز في الوصول الى الموارد المادية لنظام الحاسب [3] . يتم ترتيب الحلقات بطريقة هرمية من الأكثر امتيازا، أي الوصول الأكثر موثوقية وغير المقيد إلى الموارد إلى الأقل امتياز أي الأقل موثوقية ومقيدة الوصول إلى الموارد. الحلقة (Ring- 0) هي الحلقة الأكثر امتيازًا التي تتفاعل مباشرة مع موارد الأجهزة المادية [3-4] ،لا تستطيع الحلقات الأقل امتيازا الوصول إلى الحلقات الداخلية بدون تعليمات محددة مسبقًا . [5]توفر بنى المعالج الأكثر شيوعاً أربع حلقات حماية ويتم عادة استخدام الحلقتين Ring-0 و Ring-3 فقط. يسمى نمط العمل على مستوى الحلقة Ring - 0 بنمط النواة (kernel mode) و يُعرف أيضًا باسم وضع المشرف أو مساحة النظام ،و نمط العمل على مستوى الحلقة Ring-3 بنمط المستخدم user mode .. يتم تنفيذ التعليمات البرمجية في وضع النواة دون أي تقييد في الوصول إلى الموارد المادية الأساسية من معالج و ذاكرة ، يمكن في هذا الوضع تنفيذ أي تعليمة من تعليمات وحدة المعالجة المركزية، ويمكن الرجوع الى أي عنوان في الذاكرة . يتم تنفيذ التعليمات البرمجية في وضع النواة مع عدم قدرة الوصول المباشر إلى الموارد المادية الأساسية أو الرجوع إلى أي عنوان ذاكرة،وبالتالي يجب تفويض التعليمات البرمجية التي تعمل في هذا الوضع إلى واجهات برمجة تطبيقات النظام (API)للوصول إلى العتاد المادي من معالج أو الذاكرة[3-6] .

في البيئة الافتراضية يعمل مدير الآلة الإفتراضية Hypervisor في وضع ال kernel mode، لأن من أهم مسؤوليات ال hypervisor تعيين موارد الأجهزة وتخصيص عناوين الذاكرة إلى الآلات الإفتراضية VMs التي سيتم استضافتها عليه، كما يتحكم نوع البيئة الإفتراضية بنمط الحماية الذي تعمل عليه نواة نظام التشغيل الضيف الموجود على الآلة الإفتراضية. يوضح الشكل (3) امتيازات الوصول الى الموارد الحاسوبية في كل حلقة حماية .



الشكل(3)امتيازات الوصول الى الموارد الحاسوبية في كل حلقة حماية

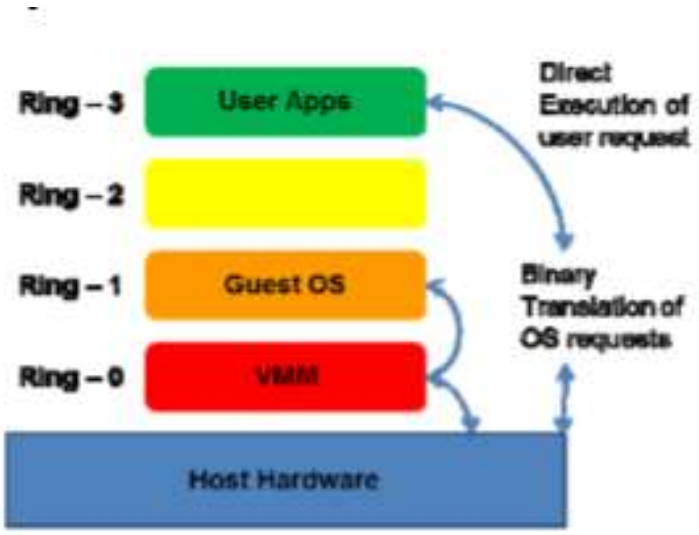
6- النمط الافتراضي الكامل (Full virtualization) :

المحاكاة الافتراضية الكاملة هي محاكاة افتراضية لا يدرك فيها نظام التشغيل الضيف أنه في بيئة عمل افتراضية، وبالتالي يتم محاكاة العتاد المادي افتراضياً عن طريق نظام التشغيل المضيف، بحيث يمكن للضيف أن يصدر الأوامر و التعليمات إلى ما يعتقد أنه عتاد مادي حقيقي **[7]** . يوفر هذا النمط أمان و عزل كامل لأنظمة التشغيل التي تعمل على الآلة الإفتراضية حيث يمكن تشغيل أكثر من نظام تشغيل على نفس المضيف و في نفس الوقت دون أن يحدث تداخل في الموارد المادية .

هناك نوعان من النمط الافتراضي الكامل:

- **النمط الافتراضي الكامل المعتمد على العتاد البرمجي** (Software assisted fullvirtualization) :

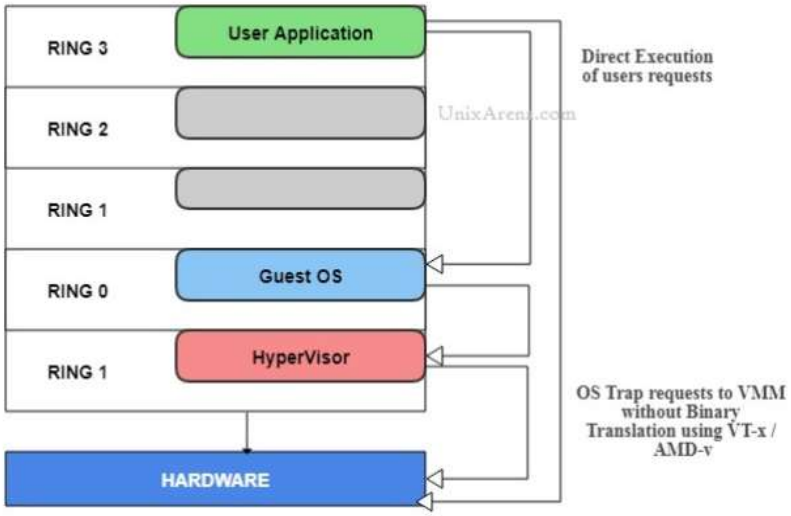
يعتمد بشكل كامل على الترجمة الثنائية (Binary translation) ،حيث يتم محاكاة جميع موارد العتاد الصلب افتراضيا عن طريق مجموعة من التعليمات البرمجية ، ويتم تنفيذ أوامر التطبيقات المنصبة على الآلة الإفتراضية عن طريق الترجمة الثنائية لهذه الأوامر. الشكل (4) يوضح النمط الإفتراضي الكامل المعتمد على العتاد البرمجي و حلقة الحماية التي يعمل عليها نظام التشغيل الضيف في هذا النمط.



الشكل(4) النمط الإفتراضي الكامل المعتمد على العتاد المادي

- **النمط الافتراضي الكامل المعتمد على العتاد الصلب** (Hardware assisted full virtualization) :

في هذا النمط تحتوي بنية المعالج الدقيق على تعليمات خاصة للمساعدة على محاكاة العتاد المادي افتراضياً [4].قد تسمح هذه التعليمات بإعداد سياق ظاهري بحيث يمكن للضيف تنفيذ التعليمات المميزة (privileged instructions) مباشرة على المعالج دون التأثير على المضيف . يتم في هذا النمط توسعة عمل المعالجات من خلال إضافة عتاد مادي لدعم المحاكاة الإفتراضية دون الحاجة لتقنية الترجمة الثنائية، تعتبر معالجات Intel (VT) و (AMD-V) من أولى المعالجات التي تدعم ماديا المحاكاة الإفتراضية. يعتبر هذا النمط مع خوارزمية النسخ المسبق النموذج الاساسي لتطبيق مفهوم الهجرة الحية في مراكز البيانات وفي جميع الدراسات و الأبحاث. الشكل (5) يوضح النمط الافتراضي الكامل المعتمد على العتاد الصلب و حلقة الحماية التي يعمل عليها نظام تشغيل الضيف.

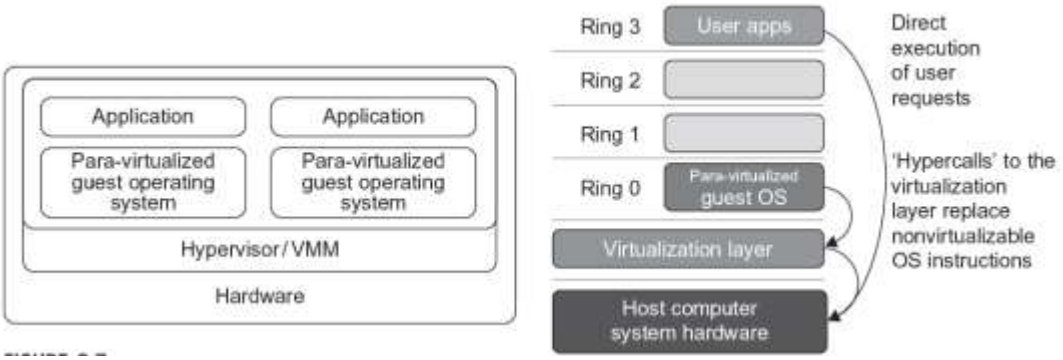


الشكل (5) النمط الافتراضي الكامل المعتمد على العتاد الصلب

7- النمط الافتراضي الجزئي (Paravirtualization) :

في هذا النمط يقوم مدير الآلة الإفتراضية (hypervisor) بإجراء تعديل على نواة نظام تشغيل الضيف لتشغيله في البيئة الافتراضية بحيث يكون هناك تعاون بين الآلة الإفتراضية و ال hypervisor لتحقيق أفضل أداء ممكن [3] . بدل من عمل طبقة افتراضية بالكامل تقوم hypervisors التي تعتمد هذا النوع بتزويد النظام الضيف بما يسمى النوافذ المبرمجة(Application Programming Interface) ومختصرها API [7] . تسمح هذه النوافذ للنظام الضيف من استعمال العتاد الحقيقي (Physical Hardware)من خلال التخاطب مع ال hypervior عن طريق استدعاءات تسمى بال hypercalls .أكثر ما يميز هذه التقنية هي أن الآلات الإفتراضية تكون مدركة لحقيقة أنهم يعملون في بيئة افتراضية [8-9] . يمتاز الـ Para Virtualization بالمرونة في إضافة العتاد و حذفه عند الحاجة دون الحاجة الى عمل إعادة تشغيل للنظام الضيف، فمثلاً تستطيع إضافة مساحات أخرى من الـ RAM للنظام الضيف عند حاجته لذلك دون أن تقوم بإيقاف عمل النظام وإعادة تشغيله مرة أخرى .الشكل(6) يوضح الهيكل العام للنمط الإفتراضي الجزئي و مستوى الحماية الذي يعمل عليه ال hypervisor .

في هذا البحث قمنا بتعديل ال hypervisor نوع Xen ليعمل في النمط الافتراضي الجزئي و طبقنا خوارزمية النسخ المسبق (pre-copy) للهجرة الحية لترحيل الالة الإفتراضية من مخدم الى آخر.



الشكل(6) النمط الافتراضي الجزئي

8-البنية المعمارية لل Xen hypervisor :

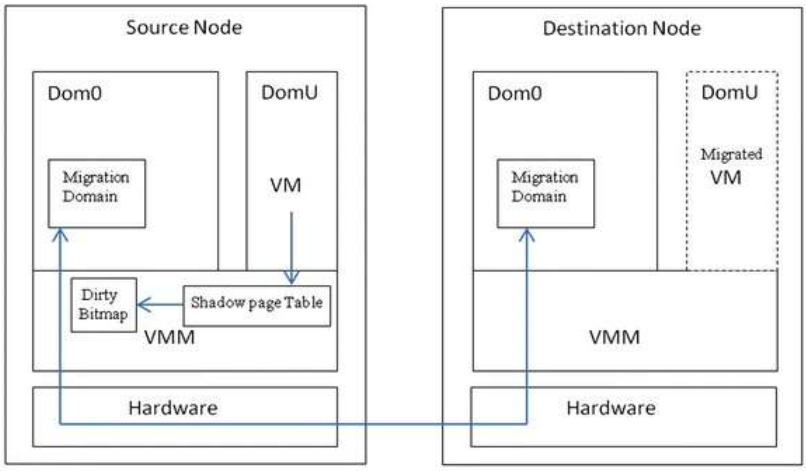
يعمل Xen hypervisor كطبقة مجردة أساسية بين الآلات الإفتراضية والعتاد الصلب، و يتحكم في أداء الآلات الافتراضية ، كما يتحكم بعمليات الجدولة الخاصة بوحدة المعالجة المركزية بالإضافة الى تقسيم الذاكرة بين الآلات الافتراضية المختلفة والتي تعمل على نفس العتاد الصلب . تم اعتماد Xen بشكل موسع كمدير للآلات الإفتراضية مفتوح المصدر ، حيث أنه يعمل في النمط الافتراضي الكامل ، و يمكن تعديله ليعمل كنمط افتراضي جزئي ، كما أنه يدعم الهجرة الحية للآلة الإفتراضية . تتكون البنية الأساسية لل Xen عند تشغيله كنمط افتراضي جزئي من مجالين : المجال (0) و المجال (U) [13] . المجال (0) هو آلة افتراضية فريدة و يعتبر مستخدم خاص ومميز يمتلك حق الوصول إلى موارد الإدخال / الإخراج الفعلية والتفاعل مع الآلات الافتراضية الأخرى [12] .

تدعى جميع الآلات الإفتراضية الاخرى التي تعمل على ال Xen hypervisor بالمجال (U) . كما يتطلب تعديل Xen ليعمل كنمط افتراضي جزئي أن يتم تشغيل المجال (0) أولاً ومن ثم المجال (U) ، بالإضافة إلى تنصيب كل من XENd و Xm و Libxenctrl .

إن Xend هو تطبيق python [12] ، ويعتبر مديراً لبيئة Xen ، و يتعامل مع الطلبات القادمة من Xen عبر المجال (0) . Xm هي أداة سطر الأوامر التي تأخذ المدخلات وتسلمها إلى Xen عبر XML RPC (Remote Procedure call) . Libxenctrl هي مكتبة C تدعم Xend للتخاطب مع Xen عبر المجال (0).

يعتبر المجال U مستخدماً محظوراً لا يتمتع بإمكانية الوصول المباشر للعتاد الصلب الفعلي وتتم ادارته عبر المجال 0 (0) الشكل(7) يبين بنية Xen مع خوارزمية النسخ المسبق، حيث أنه لتنفيذ خوارزمية النسخ المسبق pre-copy يستخدم Xen بنى معطيات خاصة لنقل صفحات ذاكرة الآلة الافتراضية بشكل فعال أثناء الهجرة الحية للآلة الافتراضية . يستخدم Xen جدول صفحات ظل (shadow page table) لتسجيل صفحات الذاكرة المتغيرة على الألة الافتراضية أثناء الهجرة ، كما يستخدم خريطة ثنائية أخرى تدعى (Dirty log bitmap) لتسجيل الصفحات المتغيرة أو (dirty pages) .

إن كلٍ من جدول صفحات الظل (shadow page table) و (dirty log bitmap) يستخدمان لإدارة عمليات نقل صفحات الذاكرة للآلة الافتراضية في وقت الهجرة ، ومن أجل كل تكرار يتم فحص الخريطة الثنائية لتحديد موضع الصفحات المتغيرة ليتم تهجيرها.



الشكل (7) بنية ال Xen مع خوارزمية النسخ المسبق pre-copy

9- الهجرة الحية (live migration) :

هي ترحيل الآلة الإفتراضية من مخدم فيزيائي إلى مخدم فيزيائي آخر دون إيقافها عن العمل مع ضمان استمرارية الخدمة التي تقدمها بزمن توقف صغير جدا يقترب من الصفر، وذلك بهدف موازنة الحمل بين المخدمات الفيزيائية أو توفير الطاقة أو من أجل إجراء عمليات صيانة للمخدم الفيزيائي المصدر[10]. إن ترحيل الآلة الإفتراضية يتضمن نقل كل من صفحات الذاكرة و حالة المعالج و عمليات الدخل و الخرج و الطلبات من المخدم المصدر إلى الهدف .

يعتبر مدير الآلة الإفتراضية (hypervisor) المسؤول الرئيسي عن عملية الهجرة الحية لذلك يجب أن يحقق الشروط التالية أثناء تطبيق الهجرة الحية[11]:

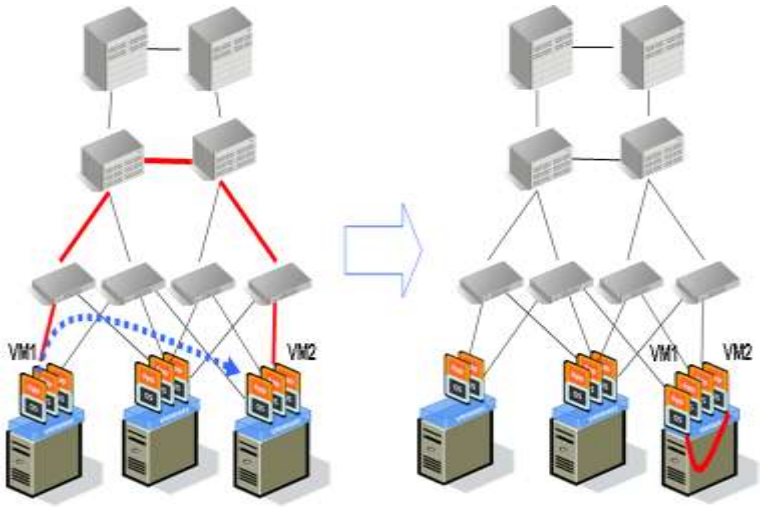
1- **استمرارية الخدمة** : إن تطبيق الهجرة الحية يجب ألا يسبب تدهور في أداء التطبيقات التي تعمل على الآلة الإفتراضية .

2- **الإستهلاك الأمثل للموارد** : يجب ألا تستهلك الموارد بشكل كبير أثناء تطبيق عملية الترحيل.

3- **التنبؤ** : يجب أن يكون من الممكن التنبؤ بزمن الهجرة الكلي و زمن التوقف و الموارد التي سيتم استهلاكها على المخدم الهدف من ذاكرة و وحدة معالجة و عرض الحزمة على الشبكة.

4- **الشفافية** : يجب أن تكون عملية الترحيل شفافة لكل من التطبيقات التي تعمل على الآلة الإفتراضية و مستخدمي هذه التطبيقات. الشكل (8) يوضح نقل الآلة الإفتراضية vm1 من مخدم الى آخر.

تعتبر خوارزمية النسخ المسبق (pre-copy) أول و أهم الخوارزميات لتنفيذ الهجرة الحية و في ما يلي سيتم شرح آلية عمل هذه الخوارزمية .



الشكل(8)الهجرة الحية للآلة الافتراضية

10- خوارزمية النسخ المسبقpre-copy :

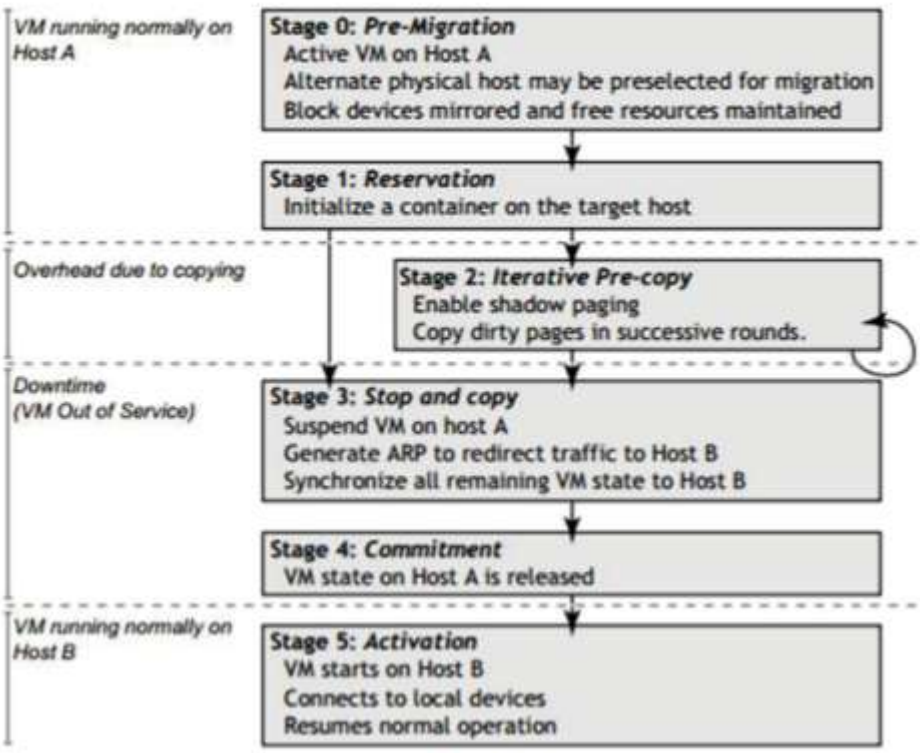
تقوم هذه الخوارزمية بتنفيذ الهجرة الحيّة و فق المراحل التالية [12] :

1- **مرحلة التحضير** : يتم حجز الموارد اللازمة لعمل الآلة الإفتراضية على المضيف الهدف.

2- **مرحلة النسخ المتكرر** :يقوم hypervisor بنسخ جميع صفحات الذاكرة من المصدر إلى الوجهة بينما لا تزال الآلة الإفتراضية تعمل على المصدر، إذا تغيرت صفحات الذاكرة (dirty pages) أثناء هذه العملية سيتم إعادة إرسالها و تكرار العملية من 2 الى n-1 مرة حيث شرط التوقف عن الإرسال و الانتقال الى الطور التالي هو عدد تكرارات 29 أو حجم الصفحات المتغيرة في الإرسال السابق 256 kb و هي قيم افتراضية للخوارزمية.

3- **مرحلة التوقف و النسخ:** يتم إيقاف الآلة الإفتراضية VM على المصدر و يتم نقل جميع الصفحات المتبقية و سجلات المعالج الى الهدف و من ثم استئناف عمل ال vm على الهدف.

الشكل (9) يوضح مراحل تنفيذ خوارزمية النسخ المسبق pre-copy .



الشكل (9) مراحل تنفيذ خوارزمية النسخ المسبق Pre-copy

يتم قياس أداء خوارزميات الهجرة الحية من خلال أربعة معايير [12]:

1- **عدد الصفحات المنقولة:** هو إجمالي عدد الصفحات المنقولة أثناء الترحيل. للحصول على أفضل أداء يجب أن تكون هذه القيمة اقل ما يمكن، كما يجب أن تكون مساوية للعدد الإجمالي لصفحات الآلة الإفتراضية التي يتم ترحيلها، ولكن في خوارزمية النسخ المسبق pre-copy هو دائما أكثر بسبب النقل المتكرر لصفحات الذاكرة المتغيرة خلال أدوار متعددة. يتم تعريف إجمالي الصفحات المنقولة Vmig على أنها العدد الإجمالي للصفحات في جميع التكرارات n و يعطى بالعلاقة التالية :

$$Vmig = \sum_{i=}^{n}, Vi \qquad (1)$$

حيث Vi هو عدد الصفحات المنقولة في التكرار الواحد و n هو العدد الإجمالي للتكرار.

2- **الزمن الكلي للترحيل:** هو الوقت المستغرق في نقل الآلة الإفتراضية بالكامل من المصدر إلى الهدف. ويجب أن يكون هذا الزمن أقل ما يمكن ويعطى بالعلاقة التالية:

$$Tmig = \sum_{i=}^{n}, Ti \qquad (2)$$

حيث ال Ti هو الزمن المستغرق لإنجاز التكرار i .

3- **زمن التوقف** : إنه الوقت المستغرق في عملية الترحيل لإيقاف الآلة الإفتراضية عند المصدر واستئنافها في العمل على المضيف الهدف. يؤثر هذا الزمن بشكل مباشر على توفر الخدمة ، حيث تعتمد قيم هذا الزمن على الصفحات المتبقية في التكرار الأخير.

4- يتم قياس زمن التوقف باعتباره الوقت الذي يستغرقه تكرار عملية الترحيل الأخيرة أو يمكن حسابه من خلال حساب زمن انقطاع الخدمة .

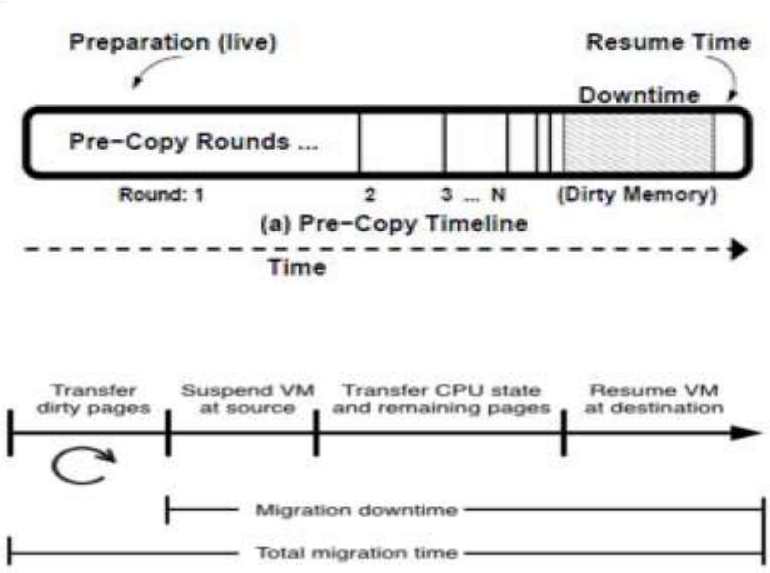
5- **overhead:** هي البيانات الإضافية التي يتم نقلها أثناء الترحيل والتي يتم تعريفها على أنها حجم صفحات الذاكرة المرسلة خلال التكرارات على حجم الصفحات الحقيقي للآلة الافتراضية و يعطى بالعلاقة :

$$Rd=\frac{Vmig}{Vmem} \quad (3)$$

حيث أن Vmig هو الحجم الكلي لصفحات الذاكرة التي تم نقلها خلال الترحيل و Vmem هو حجم الذاكرة الفعلي للآلة الإفتراضية.

يجب أن يكون ال overhead أقل ما يمكن للحصول على الأداء الأفضل .

إن الاستهلاك الزائد لوحدة المعالجة المركزية ، والتغير السريع في صفحات الذاكرة أثناء الترحيل ،والاستهلاك الكبير لعرض النطاق الترددي للشبكة ، وإجمالي عدد مرات التكرار هي بارامترات تؤثر على أداء الهجرة الحية للآلة الإفتراضية ويمكن دراستها بهدف تحسين أداء الترحيل. الشكل (10) يوضح الخط الزمني للهجرة الحية و الزمن الكلي وزمن التوقف أثناء الترحيل.



الشكل (10) الخط الزمني للهجرة الحية و الزمن الكلي وزمن التوقف أثناء الترحيل

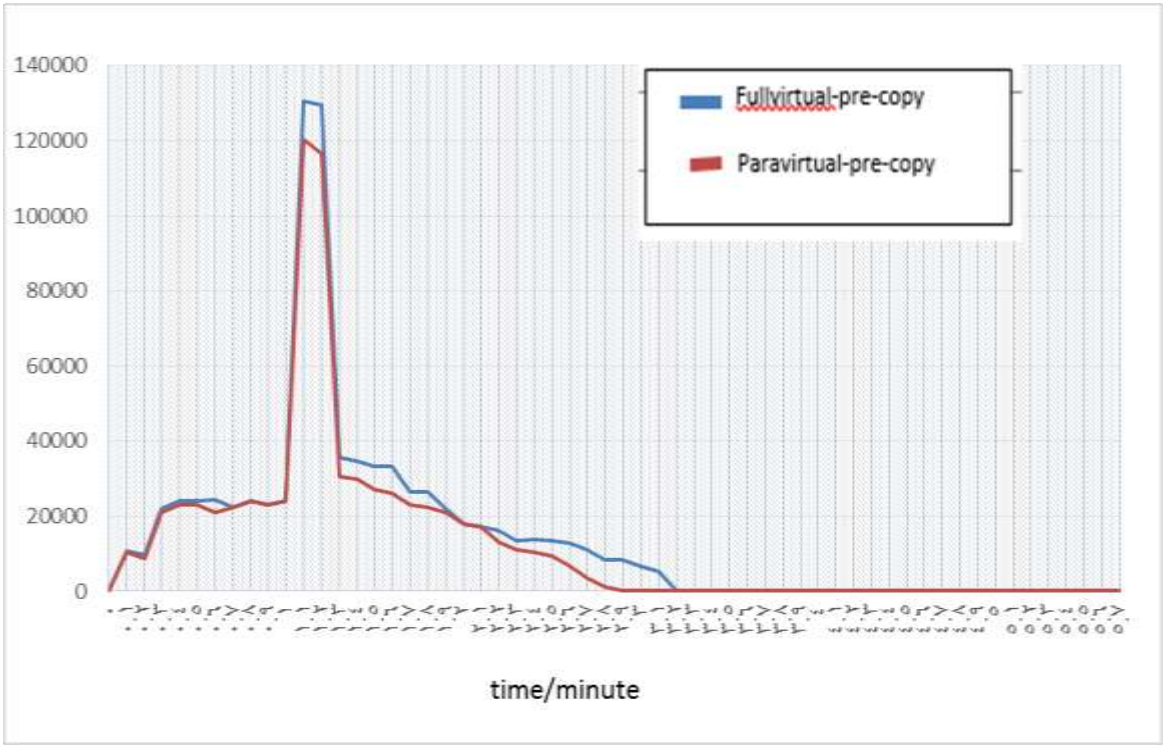
النتائج و المناقشة :

- **السيناريو الأول** من أجل n=8 و حجم الملف المرسل 6 ميغا :

عندما تأخذ n القيمة 8 سيكون عدد الرسائل الإلكترونية 64 رسالة و كل رسالة تنقل حجم ملف 6 ميغا و بالتالي حجم البيانات المنقولة عبر الشبكة 64*6 =384 ميغابايت.

الشكل (11) يبين حجم البيانات المنقولة عبر الشبكة أثناء ترحيل الآلة الإفتراضية التي تقدم خدمة البريد الإلكتروني من hyper visor 1 إلى hypervisor 2 وذلك عند 64 طلب (8*8) و حجم ملف 6 ميغا ،يمثل المحور الأفقي الزمن حيث يعرض حجم البيانات المنقولة كل 6 ثانية بعد أخذ المتوسط الحسابي للفترات السابقة و المحور العمودي حجم البيانات المنقولة (صفحات الذاكرة ،طلبات الزبائن) خلال الترحيل. يمثل المنحني ذو اللون الأحمر نموذجنا

المقترح Paravirtual_pre_copy و المنحني ذو اللون الأزرق يمثل نموذج العمل fullvirtual_precopy . قمنا بتنفيذ أمر الهجرة في الدقيقة 1 و تم حساب الزمن الكلي للهجرة من لحظة تنفيذ الأمر و حتى تشغيل البيئة الإفتراضية على المخدم الهدف ،و يمكن حسابها من المخطط من لحظة التنفيذ و حتى توقف نقل البيانات إلى الشبكة عند المخدم المصدر ،حيث أظهرت النتائج أن الزمن الكلي للهجرة في نموذجنا المقترح 2.1 دقيقة في حال كان 2.25 دقيقة في نموذج fullvirtual_pre-copy .نلاحظ أن معدل استهلاك الشبكة قد زاد بشكل كبير أثناء تطبيق الهجرة الحية و يظهر هذا الاستهلاك في ذروته بين الفترتين الزمنيتين 0.9 و 1.4 و ذلك لان خوارزمية النسخ المسبق تقوم في الطور الاول بنقل كامل الذاكرة للآلة الإفتراضية و من ثم تقوم بإرسال صفحات الذاكرة المتغيرة . يظهر المخطط أن النموذج المقترح قد خفض من حجم صفحات الذاكرة المنقولة خلال عملية الترحيل و بالتالي قلل حجم البيانات الإضافية المنقولة أثناء عملية الترحيل.



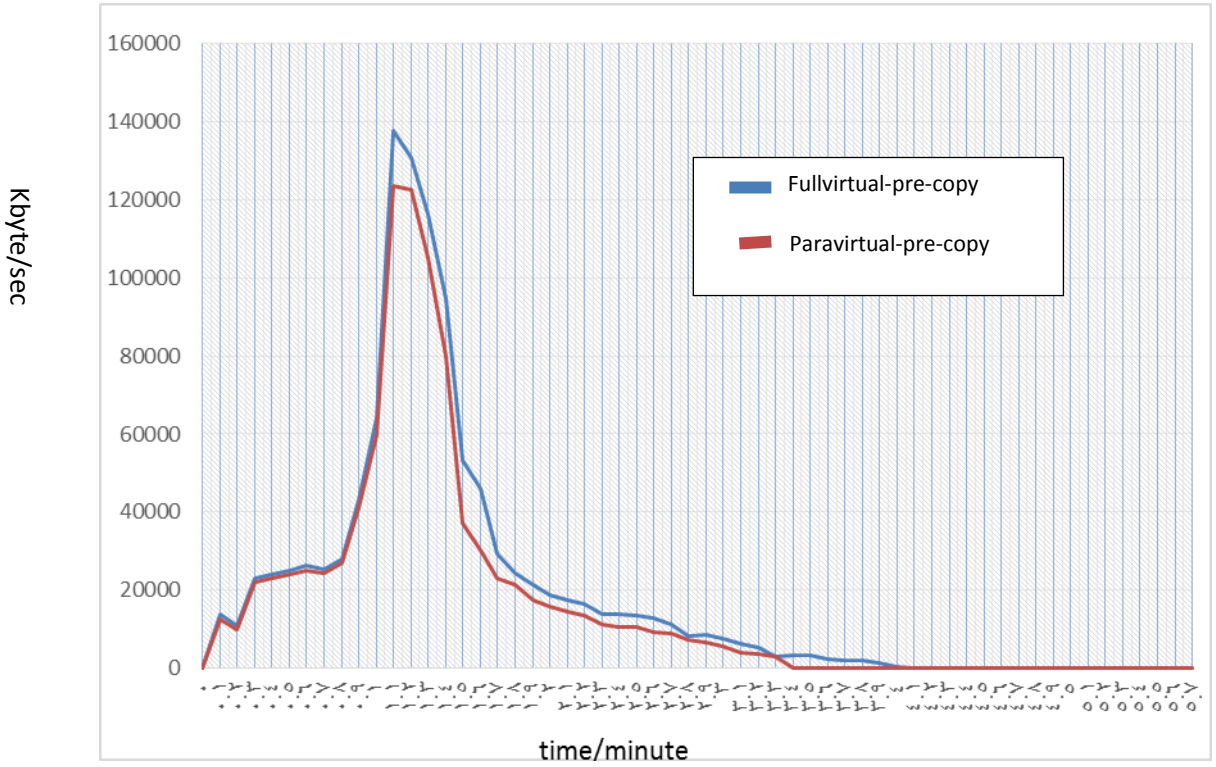
الشكل (11) حجم البيانات المنقولة عبر الشبكة أثناء ترحيل الآلة الإفتراضية مع 64 طلب و حجم ملف 6 ميغابايت

- **السيناريو الثاني** n=12 و حم الملف 6 ميغا :

عندما تأخذ n القيمة 12 سيكون عدد الرسائل الإلكترونية 144 رسالة و كل رسالة تنقل حجم ملف 6 ميغا و بالتالي حجم البيانات المنقولة عبر الشبكة 144*6 =864 ميغابايت. إن زيادة n بمقدار صغير سيزيد حجم البيانات المنقولة عبر الشبكة بشكل كبير كما هو مبين ضمن حلقتي for .

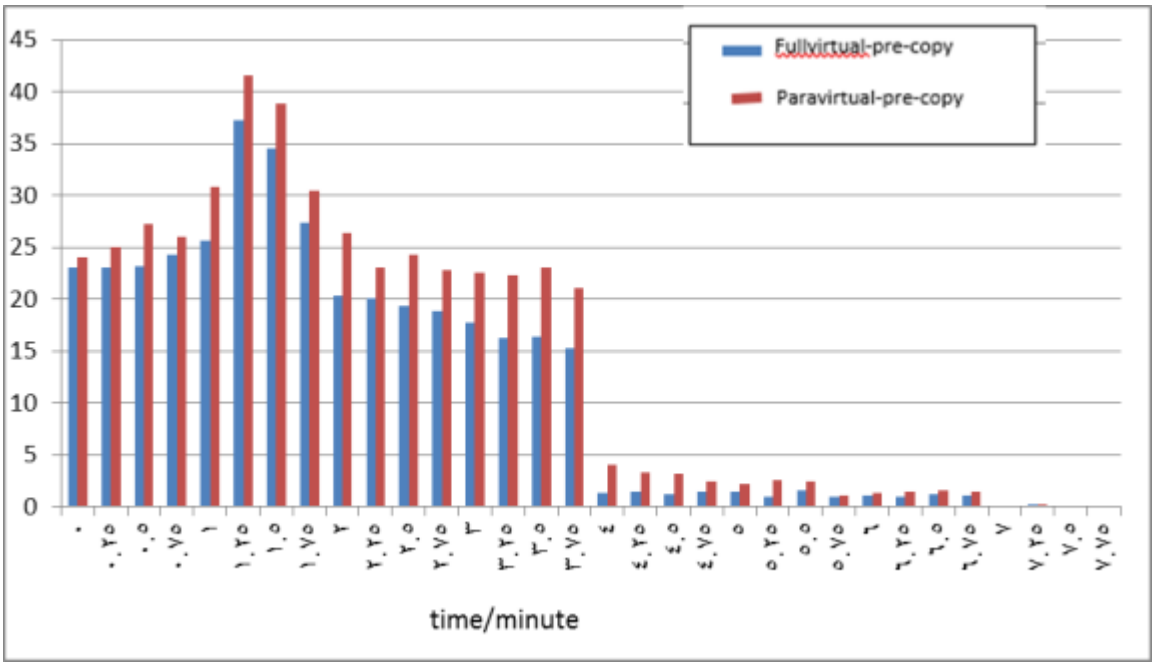
الشكل (12) يبين حجم البيانات المنقولة عبر الشبكة أثناء ترحيل الآلة الإفتراضية التي تقدم خدمة البريد الإلكتروني من المخدم A إلى المخدم B وذلك عند 144 طلب (12*12) و حجم ملف 6 ميغا .قمنا بتنفيذ أمر الترحيل في الدقيقة 1.

حيث أظهرت النتائج أن الزمن الكلي للهجرة في النموذج المقترح 2.4 دقيقة في حال كان 2.95 دقيقة في نموذج fullvirtual_pre-copy .نلاحظ أن معدل استهلاك الشبكة قد زاد بشكل كبير أثناء تطبيق الهجرة الحية و يظهر هذا الإستهلاك في ذروته بين الفترتين الزمنيتين 1 و 1.6 و ذلك لان خوارزمية النسخ المسبق تقوم في الطور الاول بنقل كامل الذاكرة للآلة الإفتراضية و من ثم تقوم بإرسال التغيرات في صفحات الذاكرة ،يظهر المخطط أن نموذجنا المقترح قد قلل من حجم صفحات الذاكرة المنقولة خلال عملية الترحيل و بالتالي قلل حجم البيانات الإضافية المنقولة أثناء عملية الترحيل.



الشكل (12)حجم البيانات المنقولة عبر الشبكة أثناء ترحيل الآلة الإفتراضية مع 144 طلب و حجم ملف 6 ميغابايت

الشكل (13) يبين نسبة استهلاك المعالج على المخدم المصدر أثناء تطبيق الهجرة الحية وذلك عند 144 طلب وحجم ملف 6 ميغا .نلاحظ من المخطط أن نموذجنا المقترح Paravirtual-pre-copy يستهلك المعالج بشكل أكبر من النموذج الأساسي fullvirtual-pre-copy و تظهر الزيادة بشكل واضح بين الدقيقة 1 و الدقيقة 3.75 و هي الفترة الزمنية التي تم فيها ترحيل الآلة الإفتراضية، ولكن هذه الزيادة مقبولة و لا تتخطى مستوى اتفاقيات جودة الخدمة



الشكل (13) استهلاك المعالج نسبة استهلاك المعالج أثناء تطبيق الهجرة الحية وذلك عند 144 طلب و حجم ملف 6 ميغابايت

الاستنتاجات والتوصيات :

إن استخدام النمط الإفتراضي الجزئي مع خوارزمية النسخ المسبق قد حسن من زمن الهجرة الحية للآلة الإفتراضية مقارنة مع استخدام النمط الإفتراضي الكلي مع خوارزمية النسخ المسبق و ذلك لأنه في النمط الإفتراضي الجزئي يعمل نظام التشغيل الضيف و مدير الالة الإفتراضية Hypervisor معًا بكفاءة أعلى ، دون الحاجة إلى فرض أعباء إضافية ناتجة عن محاكاة موارد النظام، كما قلل من الحجم الكلي لصفحات الذاكرة المنقولة عبر الشبكة و بالتالي حسن من معدل استهلاك الشبكة.

من خلال التجارب لاحظنا أن استخدام النط الإفتراضي الجزئي قد زاد من استهلاك المعالج أثناء تطبيق الهجرة الحية لكن هذه الزيادة تبقى ضمن حدود اتفاقيات مستوى الخدمة ISA .

المراجع

(1) PANKAJDEEP KAUR AND ANITA RANI CSE DEPARTMENT GNDU, RC, JALANDHAR, *Virtual Machine Migration in Cloud Computing* , International Journal of Grid Distribution Computing. Vol. 8, No.5, 2015, pp.337-342.

(2)ERIK GUSTAFSSON,*Optimizing Total Migration Time in Virtual Machine Live Migration*,Examensrbete 30 hp Mars 2016.

(3) J. HOOPES, *Virtualization for Security*, Syngress - Elsevier Inc., 2009 .

(4) D. BARRETT, AND G. KIPPER, *Virtualization and Forensics:A Digital Forensic Investigator's Guide to Virtual Environments*, Syngress - Elsevier Inc., 2010.

(5) C. GEBHARDT, C. DALLON, AND A. TOMLINSON, *Seperating Hypervisor Trusted Computing Base Supported by Hardware, Proc*. 5th ACM Workshop on Scalable Trusted Computing, STC'10, Octboer 4, 2010, Chicage, Illinois, USA, pp. 79-84.

(6) FATMA BAZARGAN,CHAN YEOB YEUN, MOHAMED JAMAL ZEMERLY, Electrical and Computer Engineering Department, *State-of-the-Art of Virtualization, its Security Threats and Deployment Models*, International Journal for Information Security Research (IJISR), Volume 2, Issues 3/4, September/December 2012.
(7) D. JANNARM AND T. KAEWKIRIYA, *Framework of Dynamic Resource Allocation System for Virtual Machine in Virtualization System*, International Journal of Computer Theory and Engineering, Vol. 8, No. 4, August 2016.
(8)] MONCHAI BUNYAKITANON,MENGYUAN PENG, *Performance Measurement of Live Migration Algorithms*, Master Thesis Electrical Engineering September 2014.
(9) SARASWATHI AT A, KALAASHRI.Y.RA B, DR.S.PADMAVATHI C1, *Dynamic Resource Allocation Scheme in Cloud Computing*, Procedia Computer Science 47 (2015) 30 – 36.
(10) Umesh Bellur and Prof. Purushottam Kulkarni, Department of Computer Science and Engineering Indian Institute of Technology Bombay, Performance Modeling of Virtual Machine Live Migration, March 13, 2015.
(11) D. JANNARM AND T. KAEWKIRIYA, *Framework of Dynamic Resource Allocation System for Virtual Machine in Virtualization System*, International Journal of Computer Theory and Engineering, Vol. 8, No. 4, August 2017.
(12) SANGEETA SHARMA AND MEENU CHAWLA, *A three phase optimization method for precopy based VM live migration*, Springerplus. 2016; 5(1): 1022.
(13) Dr.KASSEM KABALAN ,HAIDER KHALIL,*Performance evaluation of virtual machine manager by using DRBD as a shared storage of virtual disk,*ISSN:2079-3081,vol.39,NO.3,2017.