تعديل بروتوكول الاتصال XENVMC لتحسين كفاءة الهجرة الحية للآلات الافتراضية المتصلة والمشتركة في الاستضافة

طالب الدكتوراه: حيدر خليل

كلية الهندسة المعلوماتية – جامعة تشرين

اشراف الأستاذ الدكتور: أحمد صقر أحمد

الملخص

تعتبر الهجرة الحية للآلات الافتراضية من أهم الميزات التي تقدمها البيئات الافتراضية، و تعرّف على أنها ترحيل الآلة الافتراضية من مضيف مادي إلى مضيف مادي آخر مع الحفاظ على استمرارية خدمة التطبيقات التي تعمل على هذه الآلة.

تعد خوارزمية النسخ المسبق PRE-COPY من أهم خوارزميات الهجرة الحية، يعتمد مبدأ عمل هذه الخوارزمية على النقل المتكرر لصفحات الذاكرة المتغيرة (dirty page) بين المصدر و الهدف، بحيث يتوقف التكرار عند عتبة معينة. تعاني هذه الخوارزمية من مشكلة إعادة الإرسال (page re-send problem) ، وهي نقل صفحات الذاكرة نفسها مراراً في كل دور ، مما يزيد من كمية البيانات المرسلة عبر الشبكة ، و بالتالي يزداد الزمن الكلي للهجرة الحية و يصبح مجموع صفحات الذاكرة المنقولة أكبر بكثير من الحجم الفعلي للذاكرة .

قمنا في هذا البحث ببناء نموذج لتحسين الزمن الكلي للهجرة الحية في حال كانت الآلات الافتراضية مشتركة في الاستضافة و متصلة فيما بينها.

قمنا بإجراء تعديل على بروتوكول الاتصال XENVMC، و هو بروتوكول يعتمد على مفهوم الذاكرة المشتركة لمعالجة الاتصال و تسريع نقل البيانات بين الآلات الافتراضية المشتركة في الاستضافة .

إن تعديل البروتوكول XENVMC قد حسن الزمن الكلي للهجرة الحية وقلل من عدد الصفحات المنقولة عبر الشبكة، وذلك لسببين أساسيين الأول تقليل عدد الصفحات المتغيرة في الذاكرة الناتج عن الاتصال بين الآلات الافتراضية، و السبب الثاني يعود إلى تسريع الاتصال و تبادل البيانات بين الآلات الافتراضية المشتركة في الاستضافة.

الكلمات المفتاحية: الهجرة الحية، البيئة الافتراضية، الآلات الافتراضية المتصلة، الذاكرة المشتركة.

Modify the XENVMC communication protocol to improve the efficiency of live migration for co-resident virtual machines

Abstract

One of the most important features offered by virtualization is the live migration of virtual machines, and it is defined as the migration of a virtual machine from one physical host to another physical host while maintaining the continuity of service for applications running on this machine.

The PRE-COPY algorithm is one of the most important live migration algorithms. The working principle of this algorithm is based on the frequent transfer of changed memory pages (dirty page) between the source and the target, so that the repetition stops at a certain threshold. This algorithm suffers from a page re-send problem, which is moving the same memory pages repeatedly in each Itern, which increases the amount of data sent over the network, and thus the total live migration time increases and the total of memory pages transferred becomes much larger than The actual size of the memory. In this paper, we build a model to improve the total time of live migration in the event that co-resident Inter-VM communication.

We have made a modification to the XENVMC communication protocol, which is a protocol based on the concept of shared memory to handle communication and speed up data transfer between co-resident virtual machines. The modification of the XENVMC protocol has improved the total live migration time and reduced the number of pages transmitted over the network, for two main reasons: the first is to reduce the number of changed pages in memory caused by communication between virtual machines, and the second reason is due to the acceleration of communication and data exchange between that co-resident Inter-VM communication.

Key words: live migration, virtualization, co-resident VM, shared memory.

1- المقدمة

تتطور الحوسبة السحابية بشكل سريع نظراً لما تقدمه من ميزات من مرونة و قابلية التوسع و توفير الطاقة و المال، الأمر الذي جعلها هدفا للبحث والسعي الدائم من قبل الباحثين لتطوير مكوناتها بهدف الحصول على أفضل أداء ممكن.

تعتبر تقنية البيئة الافتراضية (VIRTUALIZATION) من أهم مكونات الحوسبة السحابية ، نظرا لما تقدمه من ميزات من خلال تشغيل أنظمة متعددة على نفس الجهاز الفيزيائي مع توفير العزل الكامل بين هذه الأنظمة، ليعمل كل نظام بشكل مستقل على آلة افتراضية منفصلة (VM) .

تعتبر الهجرة الحية (live migration) من أهم الميزات التي تقدمها البيئات الافتراضية، و تعرف على أنها نقل الآلة الافتراضية VM من جهاز فيزيائي إلى آخر دون انقطاع خدمة التطبيقات التي تعمل عليها بهدف توفير الطاقة و المال و توزيع الحمولة و إجراء عمليات الصيانة.

إن مدير الآلة الافتراضية (Virtual Machine Manager (VMM)) هو المسؤول الرئيسي عن تشغيل أكثر من آلة افتراضية على جهاز فيزيائي واحد، كما أنه مسؤول عن عملية الهجرة الحية للآلة الافتراضية و عن آلية الاتصال بين الآلات الافتراضية . يعتمد مبدأ عمل خوارزميات الهجرة الحية على النقل المتكرر لصفحات الذاكرة المتغيرة (dirty page) 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 الحية ، و زمن التوقف كما سيزداد زمن الاستجابة للتطبيقات التي تعمل على الآلة الافتراضية التي يتم ترحيلها.

تعاني خوارزمية pre-copy من مشكلة إعادة إرسال الصفحات نفسها مرارا في كل دور ، مما يزيد من كمية البيانات المرسلة عبر الشبكة، و يصبح مجموع صفحات الذاكرة المنقولة أكبر بكثير من الحجم الفعلي للذاكرة مما يزيد الحمل بشكل كبير على الشبكة.

إن تطبيق الهجرة الحية على آلة افتراضية VM1 موجودة على مضيف HOST1 لها اتصال و تناقل بيانات مع آلة افتراضية VM2 على نفس المضيف، أو على مضيف آخر سوف يؤدي إلى أعباء إضافية و تغيرات كبيرة في الذاكرة أثناء عملية الترحيل ناتج عن الطلبات القادمة من الزبون، و عن البيانات المتبادلة بين الآلات الافتراضية المتصلة مع بعضها البعض . لذلك تم التوجه في بحثنا هذا لتقديم نموذج يستند على منهج الذاكرة المشتركة بين الآلات الافتراضية المتصلة مع بعضها البعض لتحسين كفاءة الهجرة الحية من حيث الزمن و الإنتاجية و تقليل الحمل على الشبكة ، من خلال تجاوز اتصال TCP/IP التقليدي و الاستدعاءات و التبديل غير الضروري بين الآلات الافتراضية المشتركة في الاستضافة .

2- أهمية البحث و أهدافه:

اعتمدت معظم الدراسات السابقة على دراسة الهجرة الحية و تحسين خوارزمياتها من جانب واحد فقط و هو تطبيق أمر الترحيل على آلة افتراضية ليس لديها اتصال أو تناقل بيانات مع أي آلة أخرى سواء كانت موجودة على نفس المضيف المادي أو على مضيف مادي آخر.

لذلك كان الهدف من هذا البحث تحسين كفاءة الهجرة الحية من حيث الزمن و الإنتاجية و الاستهلاك الأمثل لموارد الشبكة، من خلال تطبيق نموذج يستند على منهج الذاكرة المشتركة لمعالجة الاتصال بين الآلات الافتراضية المتصلة و المشتركة في الاستضافة .

3- طرائق البحث و مواده:

قمنا بتعديل البروتوكول XENVMC لكي يدعم الهجرة الحية للآلات الافتراضية ، بحيث يتم الاستفادة من آلية عمل البروتوكول و خوارزمية الـ Pre-cop للحصول على النموذج المقترح (Advance-XENVMC) .

الشكل (1) يوضح البيئة العملية للتجربة ، حيث استخدمنا في تجربتنا ثلاث أجهزة حاسب ، كل جهاز مزود بمعالج core i3 و ذواكر 4. GB تم تنصيب نسخة centos 7 على كلا الجهازين و تنصيب البيئة الافتراضية XEN لتعمل كـ hypervisor من النمط الافتراضي الجزئي على كلا الجهازين، و استخدمنا الجهاز الثالث ليعمل كجهاز تخزين

مشترك للقرص الافتراضي الخاص بالآلات الافتراضية ، كما وضعنا كلا المخدمين في عنقود. قمنا بتجهيز آلة افتراضية VM1 بحجم ذاكرة Mb 1على أحد المخدمين تقدم خدمة البريد الإلكتروني، و كتبنا برنامج بلغة الـ shell على آلة افتراضية أخرى VM2 تقوم بإرسال و استقبال رسائل البريد الإلكتروني ، بحيث يتم تبادل بيانات بين VM1 و VM2 أثناء ترحيل الآلة الافتراضية VM1.

قمنا بتطبيق أحمال مختلفة أثناء ترحيل الآلة الافتراضية VM1 من خلال تغيير عدد الزبائن و حجم الملف المرسل، وذلك بهدف إجراء تغيير على صفحات الذاكرة ، و إشغال الشبكة أثناء تنفيذ أمر الهجرة الحية ، حيث أن تغيير صفحات الذاكرة أثناء نقل الآلة الافتراضية يعتبر من أهم العوامل في زيادة الزمن الكلي للهجرة الحية . إن تغيير عدد الطلبات و حجم الملف المتبادل يتم من خلال البرنامج التالي ، و الذي اطلقنا عليه اسم :Mclient

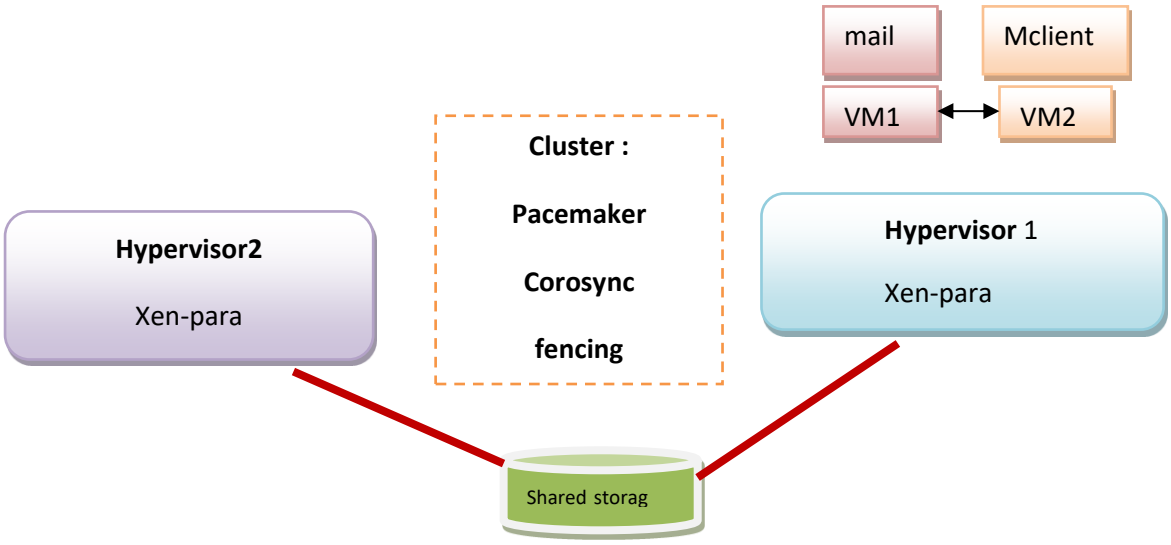
```
For I in 1  to n do
For j in 1  to n do

echo . | mail –r test$i.user@XENlab.local –a file –s "This is test"

test$j.user@XENlab.local
```

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

فمثلا من أجل n=20 سيكون عدد الرسائل الإلكترونية على الشبكة 400 رسالة بحسب حلقة for المتداخلة و كل رسالة تحتوي على ملف حجمه 3 MB.

قمنا بإجراء سيناريوهات مختلفة، و حساب الزمن الكلي للهجرة الحية في كل سيناريو ، و مقارنة النتائج أثناء ترحيل الآلة الافتراضية في حال تطبيق النموذج المقترح، و في حال تطبيق خوارزمية ال pre–copy بدون النموذج.

قمنا بتشغيل محلل الأداء NMON للحصول على النتائج و تحليليها.



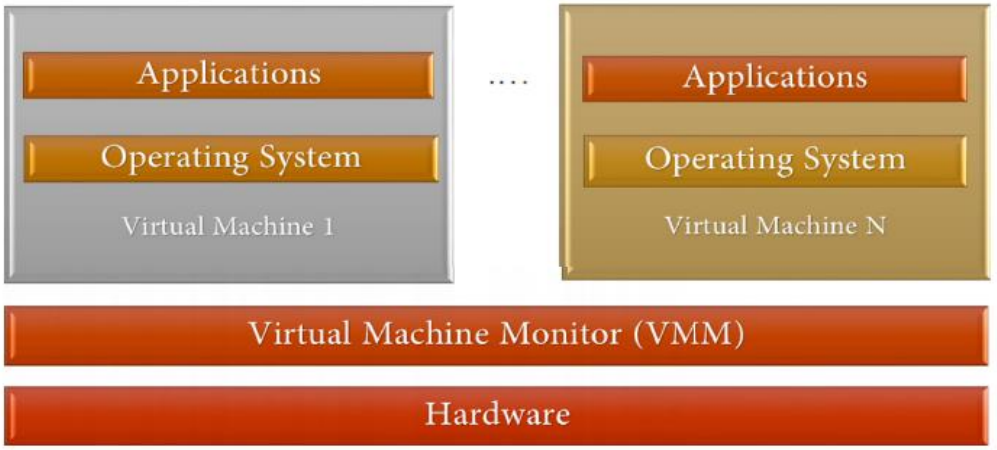
الشكل (1) بيئة عمل التجربة

4- البيئة الافتراضية Virtualization [1] :

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 ، مما يقدم استهلاكاً امثل للموارد الفيزيائية وخصوصا بعد التطور الكبير في القدرة الحسابية و العتاد المادي للأجهزة الحاسوبية ، و الذي لا يتم غالبا استخدامه و استثماره بشكل كامل من قبل أنظمة التشغيل و التطبيقات المختلفة التي تعمل على هذا العتاد. تتعامل التطبيقات الموجودة على أنظمة تشغيل الآلات الافتراضية مع الموارد المادية الحقيقية من خلال مدير الآلة الافتراضية (Hypervisor) و يسمى VMM وهو برمجية مثبتة على الجهاز المضيف للآلة الافتراضية ، و المسؤول عن تشكيل طبقة مجردة افتراضية بين التطبيقات و الموارد المادية .

يقوم مدير الآلة الافتراضية بتخصيص الموارد المادية افتراضياً من وحدة المعالجة المركزية والذاكرة و بطاقة الشبكة و أجهزة التخزين لكل نظام تشغيل يعمل على آلة افتراضية.

تختلف بنية البيئات الافتراضية بحسب مستوى العزل بين الآلات الافتراضية و بناءاً على موقع مدير الآلة الافتراضية بالنسبة للعتاد الفيزيائي . الشكل (2) يبين موضع الآلات الافتراضية على البيئة الافتراضية.



الشكل (2) البيئة لافتراضية

5- الهجرة الحية للآلات الافتراضية Live migration :

تعتبر الهجرة الحية للآلات الافتراضية من أهم الميزات التي تقدماها البيئة الافتراضية ، و تعرف على أنها ترحيل الآلة الافتراضية من مخدم فيزيائي إلى مخدم فيزيائي آخر، 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[2] .

إن ترحيل الآلة الافتراضية يتضمن نقل كل من صفحات الذاكرة و حالة المعالج و عمليات الدخل و الخرج من المخدم المصدر إلى الهدف. يعتبر مدير الآلة الافتراضية

(hypervisor) المسؤول الرئيسي عن عملية الهجرة الحية ، لذلك يجب أن يحقق الشروط التالية أثناء تطبيق الهجرة الحية [3] :

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

2- **الاستهلاك 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 الافتراضية و مستخدمي هذه التطبيقات.

تعتبر خوارزمية النسخ المسبق (pre-copy) أول و أهم الخوارزميات التي نفذت الهجرة الحية للآلة الافتراضية [4] .

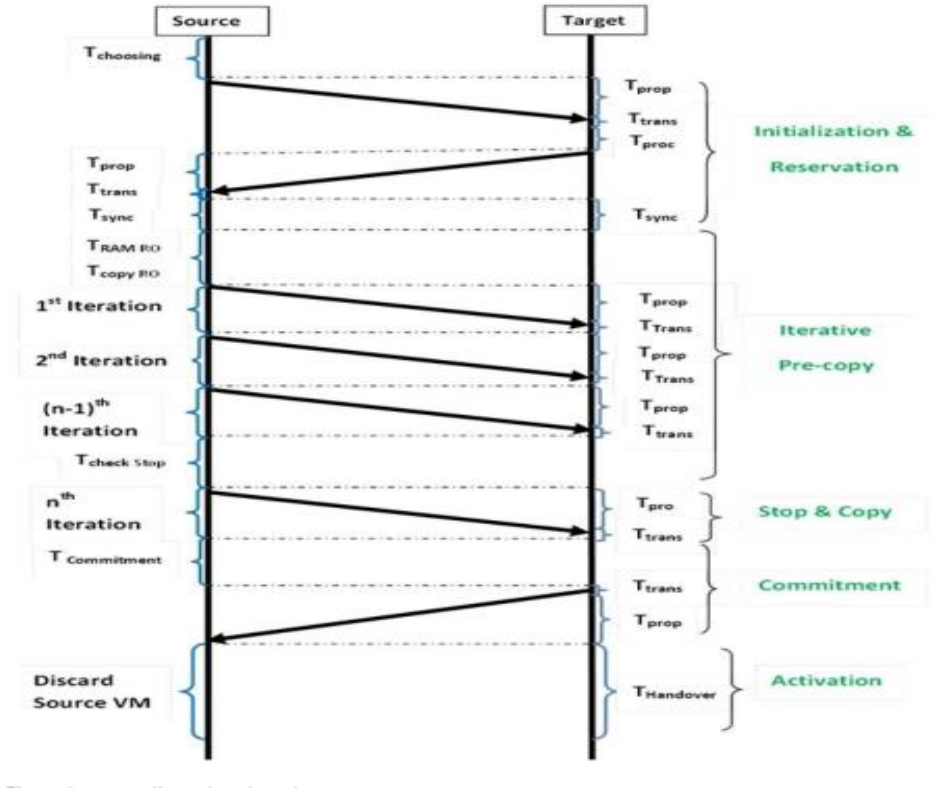
6- خوارزمية النسخ المسق (pre-copy) [5] :

يتم تنفيذ هذه الخوارزمية وفق عدة مراحل كما يبين الشكل(3) وهي :

a- **مرحلة التحضير:** يتم حجز الموارد اللازمة لعمل الآلة الافتراضية بعد تحديد المضيف الهدف.

b- **مرحلة النسخ المتكرر:** يقوم مدير الآلة الافتراضية بنسخ جميع صفحات الذاكرة من المصدر إلى المخدم الهدف ، بينما لا تزال الآلة الافتراضية 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 إلى الطور التالي هو عدد تكرارات 29 أو حجم الصفحات المتغيرة في الإرسال السابق 256 KB و هي قيم افتراضية للخوارزمية .

c- **مرحلة التوقف و النسخ:** يتم إيقاف الآلة الافتراضية VM على المصدر، و يتم نقل جميع الصفحات المتبقية و سجلات المعالج إلى الهدف، و من ثم استئناف عمل الـ VM على الهدف.



الشكل (3) مراحل تنفيذ خوارزمية النسخ المسبق Pre-copy

يتم قياس أداء خوارزميات الهجرة الحية من خلال أربعة معايير [6]:

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 الافتراضية 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 :

$$Vmig=\sum_{i=0}^{n} Vi \qquad (1)$$

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

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

$$Tmig=\sum_{i=0}^{n} Ti \qquad (2)$$

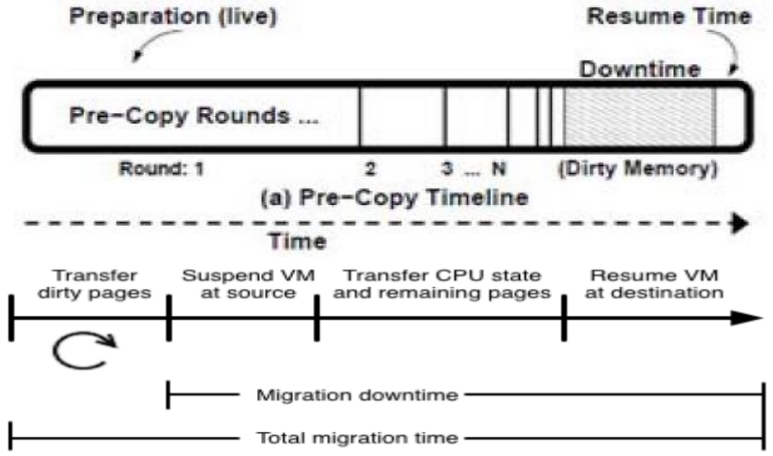
حيث أن Ti هو الزمن المستغرق لإنجاز التكرار i .

3- **زمن التوقف** : إنه الوقت المستغرق في عملية الترحيل لإيقاف الآلة الافتراضية عند المصدر واستئنافها العمل على المضيف الهدف. يؤثر هذا الزمن بشكل مباشر على توفر الخدمة ، حيث تعتمد قيم هذا الزمن على الصفحات المتبقية في التكرار الأخير. يتم قياس زمن التوقف باعتباره الوقت الذي يستغرقه تكرار عملية الترحيل الأخيرة ، أو يمكن حسابه من خلال حساب زمن انقطاع الخدمة . الشكل (4) يبين الخط الزمني للهجرة الحية.

4- **الحمل الإضافي** : 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:

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

حيث أن Vmig هو الحجم الكلي لصفحات الذاكرة التي تم نقلها خلال الترحيل و Vmem هو حجم الذاكرة الفعلي للآلة الافتراضية. يجب أن يكون ال overhead أقل ما يمكن للحصول على الأداء الأفضل .



الشكل (4) الخط الزمني للهجرة الحية و الزمن الكلي وزمن التوقف أثناء الترحيل

تعاني خوارزمية النسخ المسبق من المشاكل التالية [7]:

- مشكلة معدل النقل (transfer rate problem) :

إن الصفحات المتغيرة (dirtied page) تزداد بمعدل أسرع من معدل نقلها عبر الشبكة ، مما يستهلك الشبكة بشكل كبير، الأمر الذي يؤثر على استمرارية الخدمة و يزيد من زمن استجابة التطبيقات التي تعمل على الآلة الافتراضية ، و قد يتسبب في قطع اتصال الزبون .

- مشكلة إعادة إرسال الصفحات (page re-send problem) :

إن إرسال الصفحات المتغيرة في الادوار من 2 الى n-1 قد يؤدي الى نقل الصفحات نفسها مراراً ، مما يزيد من كمية البيانات المرسلة عبر الشبكة ، و بالتالي يزداد الزمن الكلي للهجرة للحية ، و يصبح مجموع صفحات الذاكرة المنقولة أكبر بكثير من الحجم الفعلي للذاكرة ، مما يزيد الحمل بشكل كبير على الشبكة .

إن تنفيذ أمر الهجرة الحية على الآلات الافتراضية المتصلة و المشتركة في الاستضافة أثناء تناقل البيانات فيما بينها ، سيزيد من المشكلتين السابقتين بشكل كبير.

7- البنية المعمارية للبيئة الافتراضية XEN :

يعمل XEN كطبقة مجردة أساسية بين الآلات الافتراضية والعتاد الصلب، و يتحكم في أداء الآلات الافتراضية كما يتحكم بعمليات الجدولة الخاصة بوحدة المعالجة المركزية ، بالإضافة إلى تقسيم الذاكرة بين الآلات الافتراضية المختلفة والتي تعمل على نفس العتاد الصلب. تم اعتماد XEN بشكل موسع كمدير للآلات الافتراضية مفتوح المصد، حيث أنه يعمل في النمط الافتراضي الكامل ، و يمكن تعديله ليعمل كنمط افتراضي جزئي ، كما أنه يدعم الهجرة الحية للآلة الافتراضية .

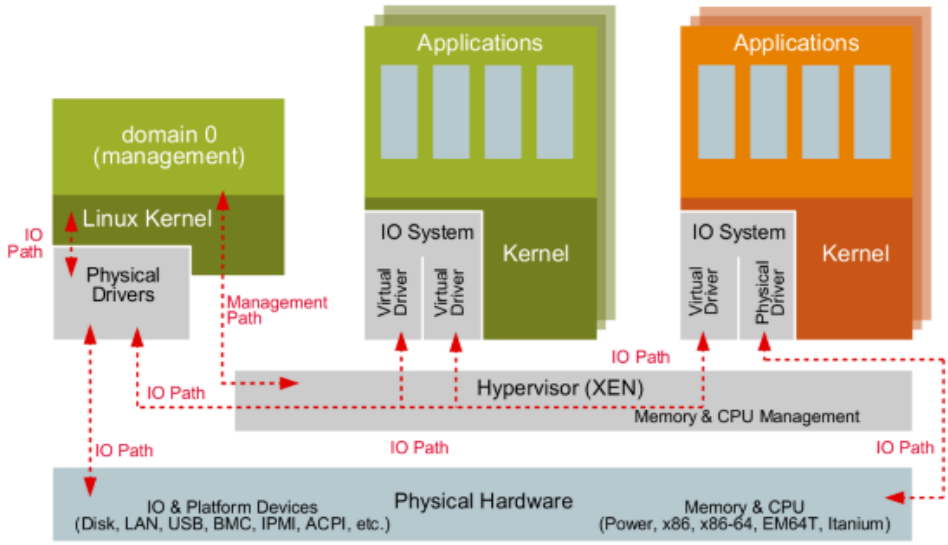
تتكون بنية XEN عندما يعمل في النمط الافتراضي الجزئي من مجالين : المجال (0) و المجال (U) . يعتبر المجال (0) مستخدم خاص ومميز يمتلك حق الوصول إلى موارد الإدخال / الإخراج الفعلية والتفاعل مع الآلات الافتراضية الأخرى [8] . تدعى جميع الآلات الافتراضية الأخرى التي تعمل على XEN بالمجال (U) .

إن تعديل XEN ليعمل كنمط افتراضي جزئي يتطلب تشغيل المجال (0) أولاً ومن ثم المجال (U) ، بالإضافة إلى تنصيب كل من الحزم XENd و Xm و LibXENctrl .

يعتبر المجال (U) مستخدماً محظوراً لا يتمتع بإمكانية الوصول المباشر للعتاد المادي الفعلي، و تتم ادارته عبر المجال (0) .

الشكل (5) يبين بنية XEN مع خوارزمية النسخ المسبق ، حيث أنه لتنفيذ خوارزمية النسخ المسبق pre-copy يستخدم XEN بنى معطيات خاصة لنقل صفحات الذاكرة للآلة الافتراضية . يستخدم XEN جدول صفحات ظل (shadow page table) لتسجيل صفحات الذاكرة المتغيرة على الآلة الافتراضية أثناء الهجرة ، كما يستخدم خريطة ثنائية أخرى تدعى (Dirty log bitmap) لتسجيل الصفحات المتغيرة أو (dirty pages) . إن كلٍ من جدول صفحات الظل و الخريطة الثنائية يستخدمان لإدارة

عمليات نقل صفحات الذاكرة للآلة الافتراضية أثناء تنفيذ أمر الهجرة، ومن أجل كل تكرار يتم فحص الخريطة الثنائية لتحديد موضع الصفحات المتغيرة ليتم ترحيلها .



الشكل (5) بنية XEN مع خوارزمية النسخ المسبق pre-copy

8- البنية الشبكية للبيئة الافتراضية XEN [9]:

يقوم XEN بتشكيل واجهة شبكية افتراضية لكل آلة افتراضية تعمل على DOMU، بحيث يتم الوصول إلى العتاد الشبكي الفيزيائي الحقيقي عن طريق المجال DOM0، أو ما يسمى المجال ذو السماحيات الأعلى و المعزول(IDD) .

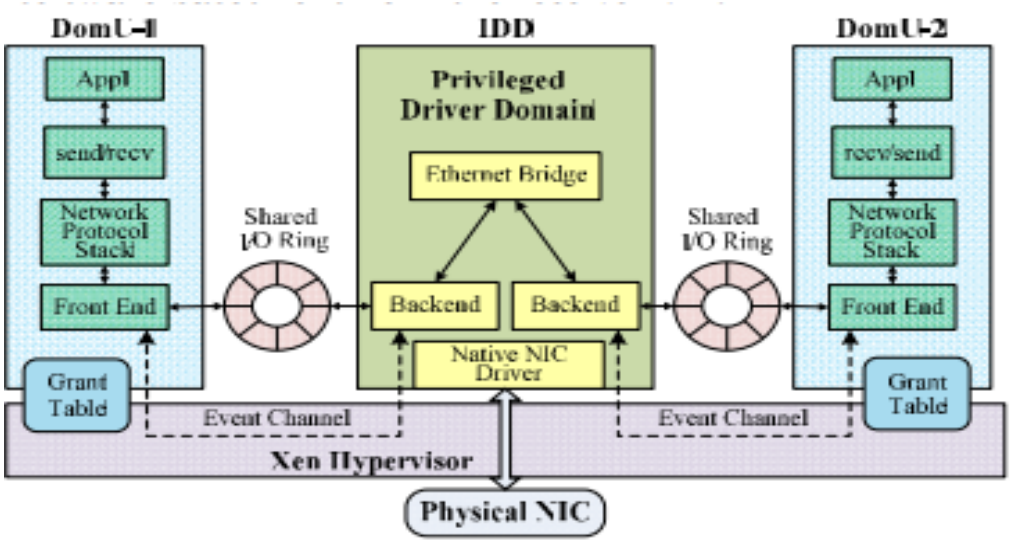
يحوي IDD برنامج تشغيل شبكي خلفي (BackEnd) لكل آلة افتراضية بحيث تتصل هذه الآلة عن طريق واجهتها الشبكية الامامية (FrontEnd) مع BackEnd المخصص لها ضمن IDD ، و ذلك لتستطيع الوصول إلى العتاد الشبكي الفيزيائي و إرسال و استقبال البيانات عبره. يوضح الشكل (6) بنية الشبكة و مجاري الدخل و الخرج في XEN .

يوفر XEN المخازن المؤقتة المشتركة بين الواجهة الأمامية والخلفية لتبادل عمليات الإدخال و الإخراج بين الآلات الافتراضية المتصلة (shared I/O Ring buffers) .

تتم إدارة عمليات مشاركة صفحات الذاكرة للحزم التي يتم تبادلها من خلال جدول المنح (granted table) و قناة الأحداث .إن نقل البيانات بين المرسل و المستقبل وفق هذا النموذج سيؤدي الى تغيرات كبيرة في صفحات الذاكرة الرئيسية لدى كل من الآلة الافتراضية المرسلة و المستقبل من اجل كل عملية تبادل بيانات ، و الذي بدوره سيزيد من الأعباء أثناء تنفيذ أمر الهجرة الحية على أحد الآلتين الافتراضيتين.

على سبيل المثال إذا أرسلت VM1 حزمة بيانات إلى VM2، فإن برنامج تشغيل الشبكة الأمامية (Frontend) في VM1 يقوم أولاً بتشكيل مراجع جدول المنح (GR) ، والتي تشير إلى صفحة الذاكرة التي تحتوي على بيانات الحزمة في مخزن الذاكرة المشترك المؤقت ، ثم يقوم بإعلام برنامج تشغيل الشبكة الخلفية (Backend) في IDD من خلال قناة الحدث لجلب مرجع جدول المنح GR. بعد الحصول على GR ، يستخدم VMM واجهة (GNTTABOP_map) لتعيين صفحة الذاكرة المشار إليها بواسطة GR على ذاكرته الخاصة، و وضع البيانات في بنية sk_buf ، بحيث يمكن معالجة هذه البيانات بواسطة مكدس الشبكة .

ويقوم بعد اكتشاف أن وجهة هذه الحزمة هي VM2 وهي مقيم مشارك، يحصل برنامج التشغيل الخلفي في VMMعلى GR من ذاكرة التخزين المشترك لـ VM2، يقوم بنسخ sk_buf في الذاكرة المشار إليها بواسطة GR الخاص بـ VM2 باستخدام واجهة GNTTABOP_copy ، وأخيراً يقوم بتنبيه الآلة الافتراضية VM2 من خلال قناة الحدث بوجود حزمة بيانات يجب استلامها [10].



الشكل (6) بنية شبكة XEN

باختصار إن نقل حزم البيانات بين المرسل و المستقبل سيمر عبر مسار طويل مارا بـ VMM من جهة المرسل، ومن ثم مكدس TCP/IP، ثم بـ VMM من جهة المستقبل ، على الرغم من أن المرسل و المستقبل متواجدان على نفس البيئة الفيزيائية .

يتم عنونة جميع الحزم المنقولة من قبل ذاكرة المرسل و المستقبل مما يسبب تغيرات كبيرة في الذاكرة، و الذي يسبب أعباء إضافية أثناء تطبيق الهجرة الحية، حيث أن الذاكرة المستخدمة لنقل و إرسال البيانات هي ذاكرة افتراضية يشكلها مدير الآلة الافتراضية، لذلك كان لا بد من إيجاد برتوكول أو نموذج اتصال جديد يتجاوز اتصال TCP/IP التقليدي، و يحقق الاتصال بين الآلات الافتراضية الموجودة على نفس المضيف (co–resident VM) دون تدخل مدير الآلة الافتراضية VMM عبر جسر الاتصال IDD. وكان من أهم هذه النماذج النموذج المعتمد على الذاكرة المشتركة لتحسين الاتصال بين الآلات الافتراضية الموجودة على نفس المضيف، وهو ما سنتكلم عنه في الفقرات التالية.

9- بروتوكول الاتصال XENVMC:

بروتوكول اتصال شبكي يعالج الاتصال بين الآلات الافتراضية ، يتميز بالشفافية و القدرة على إدراك مكان تواجد الآلات الافتراضية المتصلة مع بعضها ، هل هي على نفس المضيف (Hypervisor) أو على مضيفين مختلفين.

يقوم البروتوكول XENVMC بتسريع الاتصال بين الآلات الافتراضية من خلال تخطي مسار النقل التقليدي بين Linux DomuS، و استخدام منهج الذاكرة المشتركة لتأسيس الاتصال بين الآلات الافتراضية المتصلة و الموجودة على نفس المضيف المادي، حيث تم تصميم XENVMC وفق بحيث يحقق المبادئ التالية [11]:

- **الأداء العالي High Performance :**

يعيد XENVMC توجيه طلبات الشبكة على مستوى مدير الآلة الافتراضية إلى قنوات الذاكرة المشتركة ، مما يقود إلى مسارات اتصال أقصر وتبديلات أقل بين VMs و VMM ، ويظهر التقييم أن XENVMC يحسن إنتاجية الشبكة حتى حوالي 8 مرات مقارنةً بوضع netfront–netback الخاص بالبنية التحتية لشبكة XEN ، وذلك عند الاتصال بين الآلات الافتراضية الموجودة على نفس المضيف.

- **شفافية متعددة المستويات :**

يمكن استخدام XENVMC دون إجراء أي تعديل على XEN أو نواة LINUX ، أو حتى على التطبيقات التي تعمل على الآلة الافتراضية.

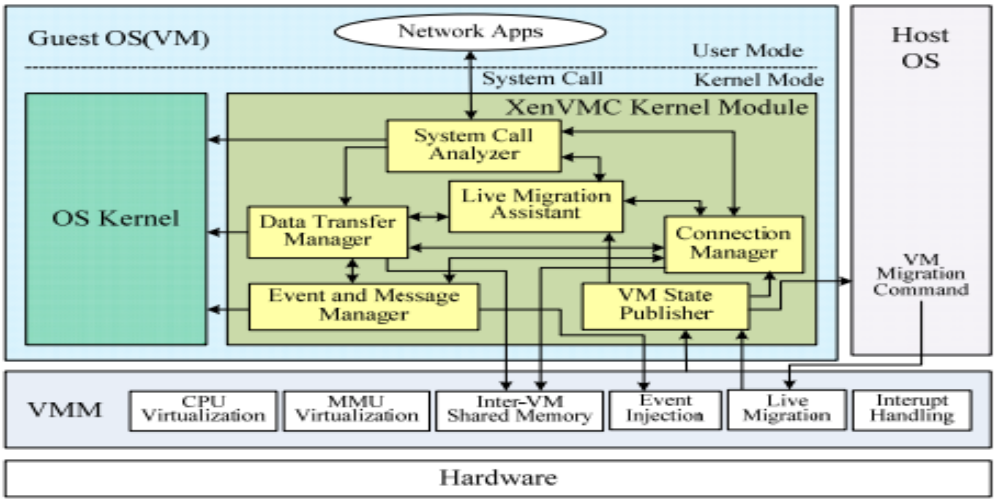
9-1 وحدة نواة XENVMC [12]:

تحتوي وحدة نواة XENVMC على ست وحدات فرعية كما هو موضح في الشكل (7) و هي :

- **مدير الاتصال (Connection Manager):** مسؤول عن إنشاء أو إلغاء الاتصالات القائمة على الذاكرة المشتركة بين نظيرين من الآلات الافتراضي، و

يقوم بترميز كل اتصال من خلال الزوج (ip/domID,port) ليتم الوصول إليه لاحقا.

- **مدير النقل (Data Transfer Manager)**: مسؤول عن إرسال البيانات واستلامها ، و يدعم كلا من وضع الحظر و فك الحظر عبر المخزن المؤقت FIFO.
- **مدير الأحداث والرسائل (Event And Message Manager)**: يتعامل مع إشعارات التنبيه المتعلقة بنقل البيانات بين المرسل والمستقبل، بحيث يتم تنفيذ آلية الإشعارات و الرسائل بين الأجهزة الافتراضية عبر حدود الأجهزة المادية .
- **محلل استدعاءات النظام (System Call Analyzer)**: اعتراض استدعاءات النظام بشكل شفاف. يعترض استدعاءات النظام ذات الصلة ويحللها. إذا تم تحديد اتصال بين آلات افتراضية مقيمة مشتركة co-resident VMs، فإنه يتجاوز مسارات TCP/IP التقليدية.
- **ناشر حالة الآلة الافتراضية (Vm State Publisher)**: هو المسؤول عن الإعلان عن تعديل عضوية الإقامة المشتركة للآلات الافتراضية الموجودة على نفس المضيف .
- **مساعد الهجرة الحية (Live Migration Assistant)**: يدعم التبديل الشفاف بين اتصال الوضع المحلي والبعيد للآلات الافتراضية .



الشكل (7) وحدة نواة XENVMC

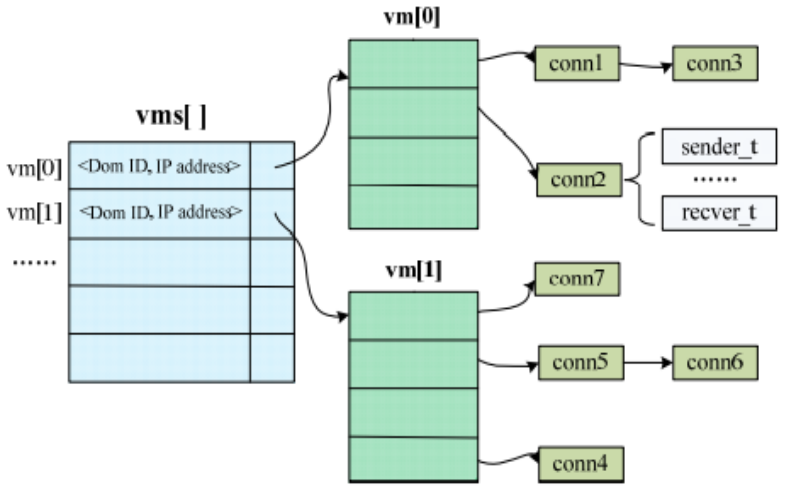
9-2 آلية عمل البروتوكول XENVMC :

1- تحديث معلومات الإقامة المشتركة للآلات الافتراضية المتصلة:

يستخدم البروتوكول XENVMC المصفوفة []VMs للتحديث و الكشف التلقائي عن الآلات الافتراضية المشتركة في الاستضافة، بحيث تحتفظ كل آلة افتراضية بنسختها الخاصة من المصفوفة []VMs. يبين الشكل (8) إن كل عنصر في المصفوفة []VMs يمثل آلة افتراضية تعتبر مقيم مشترك على نفس المضيف يتم تحديده بشكل فريد من خلال الزوج (ip,domID) ، و يشير إلى جدول (Hash table) يحوي جميع اتصالات الآلات الافتراضية المشتركة في الاستضافة ، و التي لديها اتصال مع الآلة الافتراضية الحالية .

يتم تمثيل الاتصال في نظام XENVMC بواسطة السجل(conn_t) و الذي يتكون من مجموعة من الحقول و هي المنفذ المحلي (lport) و المنفذ البعيد (rport)، السجلين (sender_t) و (recever_t). يحتوي السجل sender_t مؤشر يشير إلى الآلة الافتراضية المرسلة، و قائمة بالعمليات التي

تنتظر كتابة البيانات ، كما يحتوي مؤشر يشير إلى الذاكرة الظاهرية المشتركة المخصصة لنقل البيانات . تشبه بنية السجل (recever_t) بنية السجل السابق (sender_t). يتم ضبط جميع هذه العمليات من خلال البنية (Atomic_t) للسماح بالوصول المتوازي لعدد من المستخدمين [13] .



الشكل (8) مصفوفة الاتصال في البروتوكول XENVMC

يتم تحديث المصفوفة []VMs مباشرة عند إنشاء آلة افتراضية جديدة على نفس المضيف، أو عند ترحيل آلة افتراضية ، إلى المضيف الحالي، أو من المضيف الحالي إلى مضيف آخر، و يتم تبادل هذه التحديثات بين جميع الآلات الافتراضية الموجودة على نفس المضيف .

2- اعداد و إلغاء الاتصال المعتمد على منهج الذاكرة المشتركة [14] :

عندما تكتشف إحدى الآلات الافتراضية الخاصة بالضيف أول حركة مرور للشبكة إلى آلة افتراضية مشتركة معها في الاستضافة، يتم تهيئة و إعداد اتصال الآلة الافتراضية المرسلة وفق الخطوات التالية:

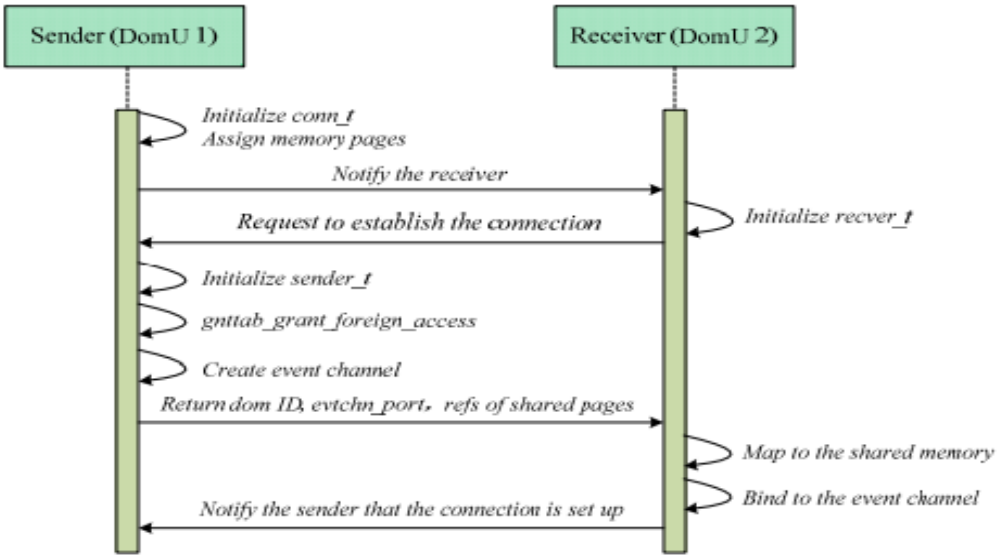
1- يتم تهيئة بنية الاتصال conn_t، و تعيين صفحات الذاكرة كمخزن مؤقت للبيانات و كتابة البيانات فيها.

2- يقوم المستلم بتهيئة نفسه عن طريق تهيئة البنية recver_t و يطلب من المرسل إنشاء الاتصال، مع إرفاق domID .

3- يقوم المرسل بتهيئة sender_t بطريقة مماثلة، ثم يتم تنشيط التابع gnttab_grant_foreign_acc الذي يوفره جدول منح XEN لمشاركة صفحات الذاكرة المخصصة مع المستقبل. والمرسل يستدعي HYPERVISOR_event_channel_op لإنشاء قناة الحدث.

4- يتم ارسال كل من معرف domID الخاص بالمرسل، و evtchn_port، و مؤشر إلى صفحات الذاكرة المشتركة من خلال مدير الرسائل و التنبيهات.

5- بعد تلقي هذه المعلومات من المستقبل، يرتبط المستقبل بـ evtchn_port، و هو منفذ الحدث الخاص بالمرسل، يقوم بتعيين صفحات الذاكرة المشتركة على مساحة العنوان الخاصة به، ويقرأ البيانات من الذاكرة المشتركة، ثم تنبيه المرسل بانتهاء إجراء إعداد الاتصال. الشكل (9) يوضح خطوات تهيئة الاتصال بين المرسل و المستقبل.



الشكل (9) تأسيس الاتصال في البروتوكول XENVMC

عندما يتم تنفيذ أمر الهجرة الحية على إحدى الآلات الافتراضية الضيف أو إيقاف تشغيلها أو إلغاء تثبيت نسخة نواة XENVMC، يتم تحرير الاتصال وفق الخطوات التالية :

1- يتم تحرير و هدم الذاكرة المشتركة الخاصة بجميع اتصالات الضيف، ويقوم ناشر الأحداث بتحديث معلومات الإقامة المشتركة عن طريق تحديث المصفوفة []VMs.

2- إيقاف إرسال البيانات وإخطار المستلم لبدء عملية الإلغاء.

3- إيقاف جميع اجرائيات المستقبل عن استقبال البيانات من خلال قفل البنية recerver_t بعد ذلك، يتم إلغاء تعيين صفحات الذاكرة المشتركة وإغلاق قناة الحدث، ثم يتم تحرير البنية recver_t.

4- يتم إغلاق قناة الحدث من قبل المرسل، حيث يتم تحرير صفحات الذاكرة المشتركة، و تحرير البنية Struct sender_t بعد هدم الاتصال.

5- يتم تحرير البنية conn_t، و تحديث []VMs على جميع الآلات الافتراضية المشتركة بالاستضافة.

3- خوارزمية إرسال و استقبال البيانات [15]:

يتم تنظيم إرسال و استقبال البيانات بين الآلات الافتراضية المشتركة بالاستضافة وفق قائمة انتظار دائرية FIFO، تبنى هذه القائمة وفق جدول المنح الخاص بـ XEN. تم تطوير قنوات اتصال الذاكرة المشتركة بحيث يستطيع XENVMC استقبال البيانات بشكل متزامن مع النوى المتعددة [15] .

إن إرسال و استقبال البيانات وفق خوارزمية XENVMC يتم وفق المراحل التالية:

- **إرسال بيانات :**

1- يتم التأكد من أن المرسل و المستقبل على نفس المضيف.

2- يتم البحث في المصفوفة []VMs في حال عدم وجود قناة اتصال محلية للمرسل، يقوم مدير نقل البيانات باستدعاء مدير الاتصال لتهيئة بنية conn_t، و إدراجها في المصفوفة []VMs.

3- إذا كانت المساحة غير المستخدمة في المخزن المؤقت FIFO كافية لإرسال البيانات، يقوم المرسل ينسخ البيانات إلى المخزن المؤقت FIFO، و يبلغ المستقبل عبر قناة الحدث بأن البيانات جاهزة ليتم استقبالها، و إلا يدخل المرسل في مرحلة انتظار وفق نمط القفل I/O mode، و ينتظر حتى يقوم المستقبل بنسخ البيانات من المخزن المؤقت و يقوم بتنبيه المرسل بذلك.

4- يتم تحديث Struct sender_t لتمييز المساحة المتاحة في المخزن المؤقت FIFO.

5- إذا تم تنفيذ أمر الهجرة الحية على الآلة الافتراضية المرسلة أو إيقافها وفق أي مرحلة من المراحل السابقة، سيتم مباشرة إلغاء الاتصال و حذف الآلة الافتراضية من الاستضافة، و تحديث المصفوفة []VMs و يتم إرسال البيانات و فق النمط TCP/IP.

- **استقبال البيانات :**

1- يتم البحث في المصفوفة []VMs، في حال عدم وجود قناة اتصال محلية، يتم تهيئة conn_t و إدراجها في المصفوفة []VMs.

2- يحدد المستقبل ما إذا كانت هناك بيانات كافية جاهزة في المخزن المؤقت (FIFO) للقراءة . إذا كانت الإجابة بنعم ، فإنه ينسخ البيانات من المخزن المؤقت و يقوم بإعلام المرسل باستلام البيانات، وإلا في حال دخول المرسل في نمط قفل I/O، ينتظر المستقبل ملف المرسل لنسخ البيانات إلى المخزن المؤقت قبل القراءة، و عندما يقوم المرسل بتنبيه المستقبل بانتهاء عملية الإرسال ، يقوم المستقبل بنسخ البيانات من المخزن المؤقت.

3- يتم تحديث بنية recver_t لتسجيل البيانات المستلمة.

4- إذا تم تنفيذ أمر الهجرة الحية على الآلة الافتراضية المستقبلة أو إيقافها وفق أي مرحلة من المراحل السابقة ، سيتم مباشرة إلغاء الاتصال و حذف الآلة الافتراضية من الاستضافة، و تحديث المصفوفة []VMs و يتم استقبال البيانات و فق النمط TCP/IP.

10- تعديل خوارزمية إرسال و استقبال البيانات في بروتوكول الاتصال XENVMC:

قمنا باستخدام منهج الذاكرة المشتركة لنقل و تبادل البيانات بين الآلات الافتراضية المتصلة و المشتركة بالاستضافة، و ذلك بهدف تحسين كفاءة الهجرة الحية، من خلال حصر التغير الكبير في الذاكرة الناتج عن عمليات تبادل البيانات بين الآلات الافتراضية أثناء عملية الترحيل، حيث أن جميع هذه التغيرات تكون على ذاكرة الاتصال المشتركة، و هي ذاكرة لا يتم إرسالها إلى المضيف الهدف أثناء تنفيذ الهجرة الحية، مما يقلل بشكل كبير إرسال الصفحات المتغيرة في كل دور من أدوار خوارزمية Pre-copy. قمنا ببناء النموذج المقترح و الذي أطلقنا عليه Advance-XENVMC من خلال تعديل خوارزمية الإرسال و الاستقبال في بروتوكول الاتصال XENVMC، حيث تم تعديل الخوارزمية وفق التالي:

- **لإرسال البيانات**

يتم إرسال البيانات وفق ثلاثة أطوار :

- **الطور الأول:** إذا كان المرسل و المستقبل على نفس المضيف، يتم الانتقال إلى المرحلة الثانية، و إلا يتم إرسال و البيانات عبر النموذج TCP/IP.
- **الطور الثاني:** قمنا بكتابة التابع **keep-con-with- migration** و الذي يقوم بالمهام التالية:

1- **في أي مرحلة من مراحل الطور الثاني، إذا تم تلقي استدعاء callback من مدير الآلات الافتراضية لترحيل الآلة الافتراضية المرسلة، يتم تنفيذ أمر الترحيل دون قطع الاتصال و يستمر تنفيذ مراحل الطور الثاني.**

2- يتم البحث في المصفوفة []VMs في حال عدم وجود قناة اتصال محلية للمرسل، يقوم مدير نقل البيانات باستدعاء مدير الاتصال لتهيئة بنية conn_t، و إدراجها في المصفوفة []VMs .

3- إذا كانت المساحة غير المستخدمة في المخزن المؤقت FIFO كافية لإرسال البيانات، يقوم المرسل ينسخ البيانات إلى المخزن المؤقت FIFO، و يبلغ المستقبل عبر قناة الحدث بجاهزية البيانات للاستقبال، و إلا يدخل المرسل في مرحلة انتظار وفق نمط القفل I/O mode، و ينتظر حتى يقوم المستقبل بنسخ البيانات من المخزن المؤقت و يقوم بتنبيه المرسل بذلك.

4- يتم تحديث Struct sender_t لتمييز المساحة المتاحة في المخزن المؤقت FIFO.

5- **إذا تم تلقي تنبيه بانتهاء عملية الترحيل في أي خطوة من خطوات الطور الثاني، يتم استدعاء التابع finish_mig، و الانتقال الى الطور الثالث.**

- **الطور الثالث: قمنا بتنفيذ مراحل هذا الطور من خلال كتابة التابع Finish_mig، و الذي يقوم بالاختبار التالي:**
  - إذا انتهت عملية **إرسال** البيانات، يتم حذف اتصالات و بيانات الآلة الافتراضية من جميع مصفوفات []VMS، وإلغاء عضويتها المشتركة على نفس المضيف .
  - إذا لم تنتهي عملية **إرسال** البيانات ، يتم إتمام نقل البيانات من خلال نموذج الاتصال TCP/IP التقليدي، و يتم حذف اتصالات و بيانات الآلة الافتراضية من جميع مصفوفات []VMS ، وإلغاء عضويتها المشتركة على نفس المضيف .

- **ثانيا: لاستقبال البيانات**

يتم استلام البيانات أيضاً وفق ثلاثة أطوار:

الطور الأول: إذا كان المرسل و المستقبل على نفس المضيف، يتم الانتقال إلى المرحلة الثانية، و إلا يتم استقبال البيانات عبر النموذج TCP/IP.

- **الطور الثاني:** يتم استدعاء التابع **keep–con–with –migration**، و الذي يقوم بالمهام التالية :

1- في أي مرحلة من مراحل الطور الثاني إذا تم تلقي استدعاء callback من مدير الآلة الافتراضية لترحيل الآلة الافتراضية المستقبل، يتم تنفيذ أمر الترحيل دون قطع الاتصال و يستمر تنفيذ مراحل الطور الثاني.

2- يتم البحث في المصفوفة []VMs، في حال عدم وجود قناة اتصال محلية ، يتم تهيئة conn_t وإدراجها في المصفوفة []VMs.

3- يحدد المستقبل في ما إذا كانت هناك بيانات كافية جاهزة في المخزن المؤقت (FIFO) للقراءة. إذا كانت الإجابة نعم، فإنه ينسخ البيانات من المخزن المؤقت، و يقوم بإعلام المرسل باستلام البيانات، و إلا في حال دخول المرسل في نمط قفل I/O، ينتظر المستقبل ملف المرسل لنسخ البيانات إلى المخزن المؤقت قبل القراءة، و عندما يقوم المرسل بتنبيه المستقبل بانتهاء عملية الإرسال، يقوم المستقبل بنسخ البيانات من المخزن المؤقت.

4- يتم تحديث بنية recver_t لتسجيل البيانات المستلمة.

5- إذا تم تلقي تنبيه بانتهاء عملية الترحيل في أي خطوة من خطوات الطور الثاني، يتم استدعاء التابع finish_mig، و الانتقال الى الطور الثالث .

- **الطور الثالث : يقوم التابع Finish_mig بتنفيذ الاختبار التالي :**
  - إذا انتهت عملية **استقبال** البيانات، يتم حذف اتصالات و بيانات الآلة الافتراضية من جميع مصفوفات []VMS، وإلغاء عضويتها المشتركة على نفس المضيف.
  - إذا لم تنتهي عملية **استقبال** البيانات، يتم إتمام نقل البيانات من خلال نموذج الاتصال TCP/IP التقليدي، و يتم حذف اتصالات و بيانات الآلة الافتراضية من جميع مصفوفات []VMS، وإلغاء عضويتها المشتركة على نفس المضيف.

11- النتائج و المناقشة :

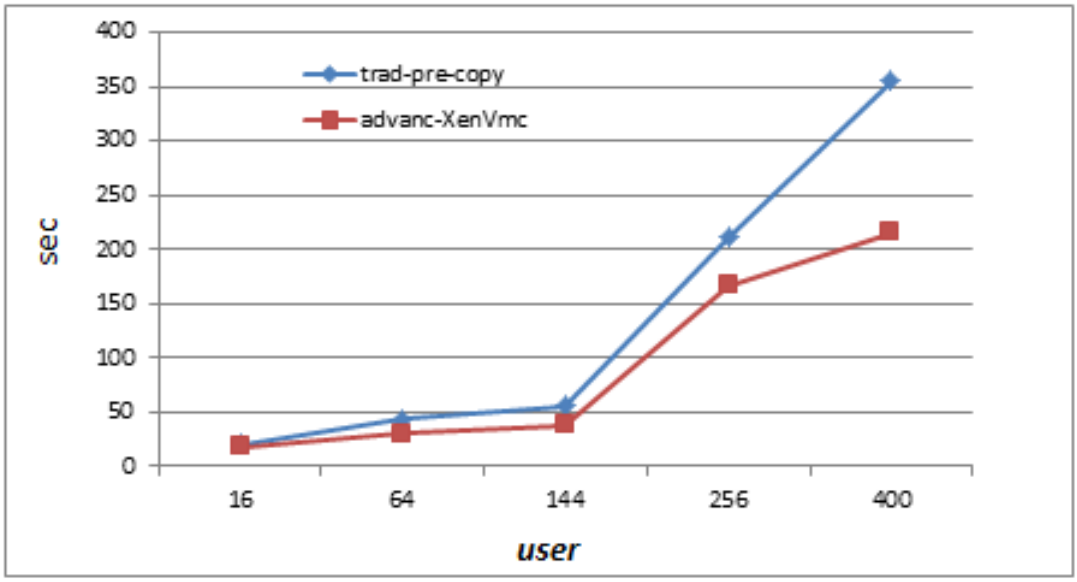
قمنا بإجراء سيناريوهات مختلفة بحيث يتم زيادة الحمل من خلال زيادة عد الطلبات و زيادة حجم الملف المرسل، وتم حساب الزمن الكلي للهجرة الحية في كل سيناريو في حال تم تطبيق نموذجنا المقترح و مقارنة النتائج في حال تطبيق خوارزمية pre-copy فقط لتنفيذ أمر الهجرة الحية.

1- الاختبار الأول قمنا بتغيير عدد الطلبات من خلال تغيير قيمة n في البرنامج Mclient و تثبيت حجم الملف MB 1 وفق الجدول (1):

الجدول (1) عدد الزبائن و حجم الملف المرسل

n	4	8	12	16	20
user	16	64	144	256	400
Filesize/mb	1	1	1	1	1

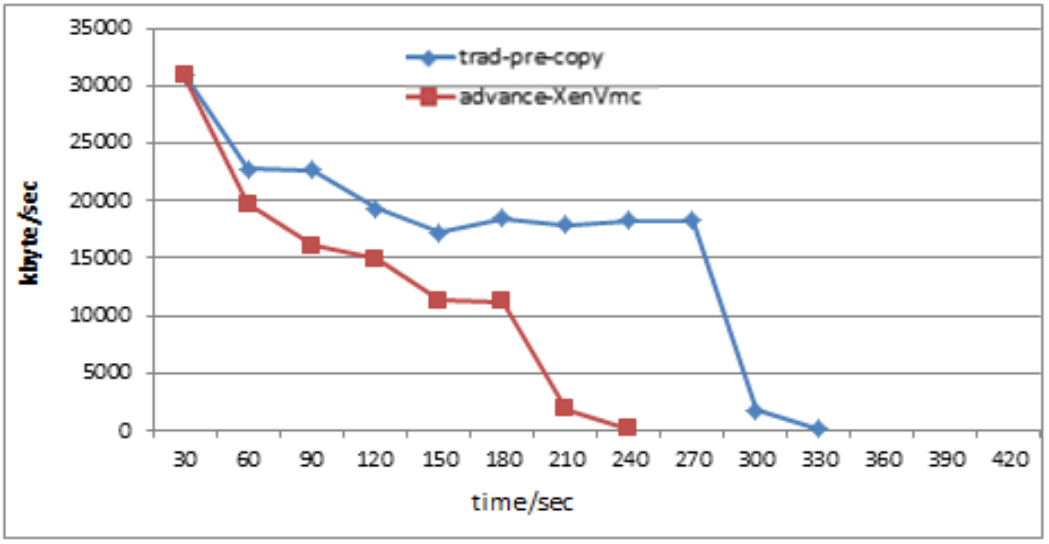
- الشكل (10) يبين الزمن الكلي للهجرة الحية، حيث يمثل المحور الأفقي عدد الـ User و المحور العمودي الزمن بالثانية . تم حساب الزمن الكلي للهجرة الحية أثناء تطبيق الهجرة الحية على الآلة الافتراضية VM1 من لحظة تنفيذ الأمر و حتى تشغيل البيئة الافتراضية على المخدم الهدف ، حيث تم الحصول على هذه القيم من خلال الأداة Xenmotion.
- تظهر النتائج بأن نموذجنا المقترح قد حسن الزمن الكلي للهجرة الحية بشكل ملحوظ ، و يظهر ذلك واضحا كلما ازداد عدد الطلبات حيث قد تصل نسبة التحسين إلى 50 بالمئة، يعود السبب في ذلك إلى كون النموذج المقترح يقلل من التغيرات الإضافية في الذاكرة نتيجة الاتصال بين الآلة الافتراضية vm1 و الآلة VM2 أثناء تنفيذ أمر الهجرة الحية، حيث تنحصر هذه التغيرات على الذاكرة المشتركة الموجودة على نفس المضيف، و لا يتم تكرار نقلها إلى الهدف أثناء ترحيل الآلة الافتراضية كما يحدث في حال استخدام خوارزمية pre-copy فقط.



الشكل (10) الزمن الكلي للهجرة الحية عند إرسال ملف بحجم 1 MB

- بالنسبة لحجم البيانات المنقولة عبر الشبكة إن الشكل (11) يبين أن استخدام النموذج المقترح قد قلل صفحات الذاكرة المنقولة عبر الشبكة أثناء ترحيل الآلة الافتراضية، و الذي أدى إلى تقليل overhead بشكل واضح. يمثل المحور الأفقي زمن تنفيذ التجربة، بحيث يقوم محلل الأداء NMON بأخذ المتوسط الحسابي للبيانات المنقولة عبر الشبكة كل 30 ثانية، و المحور العمودي يمثل حجم البيانات المنقولة خلال عملية ترحيل الآلة الافتراضية.

نلاحظ من الشكل (11) أن حجم نقل البيانات في البداية كان في ذروته في كلا المخططين، و يعود السبب في ذلك أن خوارزمية pre–copy تقوم في الدور الأول في كلا النموذجين بإرسال كامل الذاكرة إلى مكان الترحيل . كما نلاحظ انتهاء عملية الترحيل و توقف نقل البيانات في النموذج المقترح عند الزمن 245 ثانية في حال استمر نقل و تبادل البيانات في النموذج الأساسي إلى الفترة الزمنية 350 ثانية.



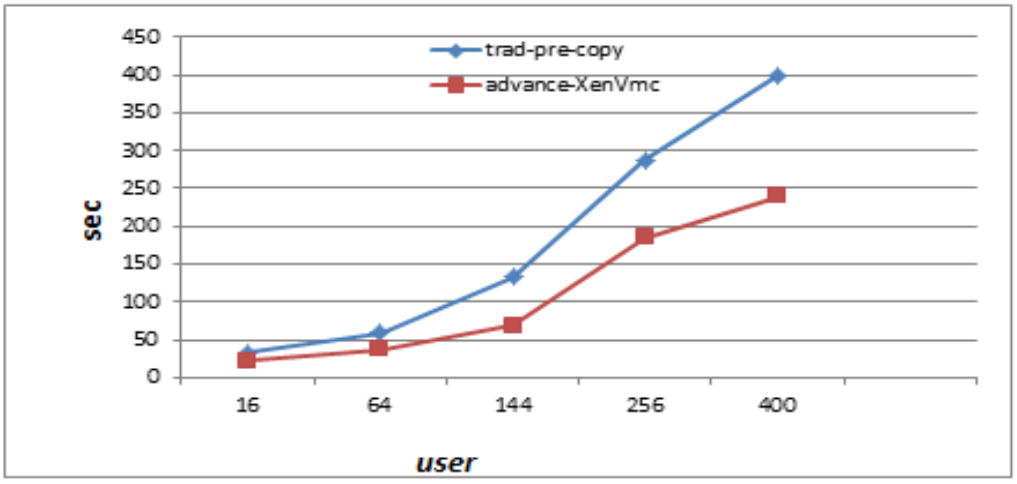
الشكل (11) حجم صفحات الذاكرة المنقولة عبر الشبكة أثناء تنفيذ الهجرة الحية

1- الاختبار الثاني: قمنا بتغيير عدد الطلبات و تثبيت حجم الملف 3 MB كما يبين الجدول (2):

2- الجدول (2) عدد الزبائن و حجم الملف المرسل

20	16	12	8	4	**n**
400	256	144	64	16	**user**
3	3	3	3	3	**Filesize/mb**

الشكل (12) يبين أن زيادة حجم الملف قد زاد الحمل على الشبكة و زاد من الزمن الكلي للهجرة الحية في كلا النموذجين. تظهر النتائج أن نموذجنا المقترح قد حسن الزمن الكلي للهجرة الحية بشكل ملحوظ و ويظهر ذلك واضحا كلما ازداد عدد الطلبات.



الشكل (12) الزمن الكلي للهجرة الحية أثناء إرسال ملف بحجم 3 MB

الخلاصة :

قدمنا في بحثنا هذا نموذج يعتمد على منهج الذاكرة المشتركة لتحسين و تسريع الاتصال بين الآلات الافتراضية المتصلة و المشتركة في الاستضافة، مما يؤثر إيجاباً بتحسين الزمن الكلي للهجرة الحية أثناء ترحيل الآلات الافتراضية المتصلة و المشتركة في الاستضافة و ذلك لسببين:

- **أولاً:** سرعة تبادل البيانات بين المرسل و المستقبل سيؤدي إلى سرعة في تنفيذ التطبيقات التي تعمل على الآلة الافتراضية التي يتم ترحيلها.
- **ثانياً:** تقليل التغيرات الكبيرة في الذاكرة و الناتجة عن الاتصال بين الآلة الافتراضية التي يتم ترحيلها و الآلات الأخرى الموجودة على نفس المضيف مما يقلل بشكل كبير إرسال صفحات الذاكرة في كل دور من أدوار خوارزمية النسخ المسبق.
- تظهر النتائج أن النموذج المقترح (Advance–XENVMC) قد حسن الزمن الكلي للهجرة الحية بشكل ملحوظ و ويظهر ذلك واضحا كلما ازداد عدد الطلبات ، حيث تصل نسبة التحسين إلى 50 بالمئة في حال ازدياد الحمل الناتج عن الاتصال بشكل كبير.
- حسّن النموذج المقترح من استهلاك الشبكة أثناء تطبيق الهجرة الحية.

من المتوقع أن نحصل على نتائج أفضل في حال تم دمج النموذج المقترح مع خوارزميات الهجرة الحية التي تقوم بإرسال الصفحات التي تتغير بشكل قليل في الدور الحالي و الصفحات المتغيرة بشكل كبير يتم تأجيل إرسالها إلى الدور الأخير، و هذا ما سنقوم بتنفيذه مستقبلاً .

المراجع :References

[1] VIRENDRA TIWARI, DR.AKHILESH A.WAOO, BALENDRA GARG, ASSISTANT PROFESSOR, ASSOCIATE PROFESSOR ,ASSISTANT PROFESSOR,2020- **Study On Virtualization Technology And Its Importance In Cloud Computing Environment**. International Journal of Creative Research Thoughts ; Volume 8, ISSN: 2320-2882 .

[2] PRATEEK JAIN, 2021- **Optimized Pre-Copy Live Virtual Machine Migration for Memory-Intensive Workloads.** National College of Ireland.

[3] JANNARM AND T. KAEWKIRIYA, 2016- **Framework of Dynamic Resource Allocation System for Virtual Machine in Virtualization System**. International Journal of Computer Theory and Engineering, Vol. 8, No. 4.

[4] EZHILARASIE,RAJAPACKIYAM,ARVIND VENKATESA SUBRAMANIAN, UMAMAKESWARI ARUMUGAM, 2020- **Live Migration of Virtual Machines using Mirroring Technique**. Journal of Computer Science; DOI: https://doi.org/10.3844/jcssp.2020.543.550,Volume 16, 543-550.

[5] MOHAMED ESAM ELSAID, HAZEM M. ABBAS ,CHRISTOPH MEINEL, 2021- **Virtual machines Pre-copy live migration cost modeling and prediction: a survey.** Distributed and Parallel Databases; https://doi.org/10.1007/s10619-021-07387-2.

[6] SANGEETA SHARMA, AND MEENU CHAWLA, 2016- **A three phase optimization method for precopy based VM live migration**. Sharma and Chawla SpringerPlus 5:1022 DOI 10.1186/s40064-016-2642-2.

[7] AHMAD SAKER AHMAD ,HAIDER KHALIL,2018- **Improve the total migration time of live migration in virtualization data centers**. Tishreen University Journal, k,ISSN:2079-3081,vol.39.

[8] G. SUNITHA REKHA, 2018- **A Study On Virtualization And Virtual Machines**. International Journal of Engineering Science Invention (IJESI), Volume 7 Issue 5 Ver. III .

[9] Dr.KASSEM KABALAN ,HAIDER KHALIL,2017- **Performance evaluation of virtual machine manager by using DRBD as a shared storage of virtual disk**. ISSN: 2079-3081,vol.39.

[10] YI REN, QI ZHANG,JIANBO GUAN, JINZHU KONG, HUADONG DAI, and LISONG SHAO, 2016 - **Shared-Memory Optimizations for Inter Virtual Machine Communication**. ACM Computing Surveys,Volume 48, No: 49.

[11] QI ZHANG, LING LIU, YI REN, KISUNG LEE, YUZHE TANG, XU ZHAO, YANG ZHOU, 2013- **Residency Aware Inter-VM Communication in Virtualized Cloud: Performance Measurement and Analysis.** IEE ,DOI: 10.1109/CLOUD.2013.116.

[12] YI REN,LIU RENSHI,YOU ZIQI(ZIVE*), 2016*- **XENVMC - A Residency Aware Transparent Inter-VM Network Communication Accelerator.** XENVMC Group of NUDT.

[13] YI REN, LING LIU, XIAOJIAN LIU, JINZHU KONG, HUADONG DAI, QINGBO WU1, YUA, 2013- **A Fast and Transparent Communication Protocol for Co-Resident Virtual Machines.** IEE, ISBN:978-1-4673-2740-4.

[14**]** YI REN, RENSHI LIU, QI ZHANG, JIANBO GUAN, ZIQI YOU, YUSONG TAN, QINGBO WU, 2020- **An Efficient and Transparent Approach for Adaptive Intra-and Inter-Node Virtual Machine Communication in Virtualized Clouds**. IEEE, DOI 10.1109/ICPADS47876.

[15] YOU ZI-QI, REN YI, LIU REN-SHI, GUAN JIAN-BO AND LIU LI-PENG, 2018- **Optimization of Co-resident Inter-VM Communication Accelerator XENVMC Based on Multi-core.** Computer Science, Vol. 45 , Issue (3): 102 107.doi: 10.11896/j.issn.1002-137X.2018.03.017.