تخفيف الضغط على الشبكة الحاسوبية باستخدام مبدأ الكاش

د. رزق غانم[1]

الملخص

أدّى تنوع بنى الشبكات، وتضخم أحجامها، وازدياد عدد مستثمريها وحجم البيانات المنقولة عبرها، فضلاً عن زيادة شعبية شبكة الويب، إلى زيادة ملحوظة في كمية حركة المرور على شبكة الانترنت الذي أدى بدوره إلى خلق مشكلات عدّة منها الاختناقات، والبطء في نقل البيانات، وزيادة تحميل المخدمات، ممّا يؤدي إلى تأخير وصول المستخدمين إلى بياناتهم عبر الشبكة إما بسبب تلقي المخدمات طلبات أكثر بكثير مما تستطيع تحمله، أو بسبب ازدحام المسار الشبكي بين المستخدم والمخدم الذي يطلبه نتيجة لحركة المرور المتزايدة على بعض الروابط التأسيسية أو على كلّها الذي بدوره يؤدي إلى تدني مستوى الشبكة كلّها. وهنا برزت الحاجة إلى استخدام مفهوم تخبئة محتوى الويب، حيث تخزّن مكونات الويب بالقرب من المستخدم، ومن ثمّ تزداد كفاءة تصفح الويب من خلال تسريع وصول المستخدمين إلى الويب، وكذلك تخفيف الضغط عن المخدمات. وغيرها العديد من الميزات التي تحققها هذه التقنية .

هدف هذا البحث إلى دراسة طرائق تخبئة صفحات الويب، ومدى تأثيرها في تحسين سرعة وصول المستخدمين الى بياناتهم عبر الشبكة ومن ثمّ تحسين أداء الشبكة كلّها، وتم اختبار النتائج على شبكة تحتوي مخدم وكيل المتوافر.

كلمات مفتاحية: الشبكات، المخدمات، المسار الشبكي، تخبئة محتوى الويب، ذاكرة التخبئة كاش، بروتوكول ICP

[1] أستاذ مساعد- قسم الحواسيب والاتمتة- كلية الهمك.

Decrease of server loads on the computer Network using the principle of cache

Dr. Rzek ghanem[(1)]

Abstract

The diversity of networks infrastructures, swell its size, increasing number of its investors and increasing the size of data transferred by it, in addition to the increasing popularity of the Web has led to noticeable increasing in the amount of traffic on the internet, which in turn led to creation of many problems such as bottlenecks, slow data transfer and servers overload leading to Delay user access to their data on the network , either because of the servers receiving requests more than it can handed it, or because of congestions network path between the user and the server which in turn leads to lower network performance as a whole.

Here there was a need to use the concept of Web Caching to save a copy of web content close to end user and thus improve the efficiency of browsing the Web as well as decrease of server loads and many other features provided by this technique.

This research aims to study ways to cache web pages and what is its impact in improving users' access to their data over the network and thus improve the network performance as a whole.

We focus on this research on proxy cache method to cache web content where the application of this research has been on two infrastructures and the results was taken from proxy servers available on the network.

Keywords: Networks, servers, network path, caching web content, cache, ICP

(1) Assistant Professor –Department of Computer& Automation Engineering- Electrical faculty Damascus university.

1-المقدمة

مع ازدياد تطور النظم المعلوماتية، وتشعبها، وتضخم الشبكات الواصلة بينها، واتساع انتشار شبكة الانترنت، وتنّوع تطبيقاتها، وازدياد الضغط عليها، أدى ذلك إلى جعلها المسبب الرئيسي لتدني مستوى أداء الشبكة الحاسوبية، وظهر مفهوم الكاش في البداية في المعالجات اذ استخدم مصطلح ذاكرة الكاش للدلالة على الذاكرة التي توضع بالقرب من وحدة المعالجة المركزية CPU من أجل تسريع النفاذ، وبشكل مشابه فانّ ذاكرة الكاش الخاص بالقرص الصلب أو ما يسمى (مخبأ القرص)disk cache هي ذاكرة تخزن صفحات القرص الصلب التي يتم الوصول إليها بشكل متكرر بهدف تسريع النفاذ أيضاً.

2- أهمية البحث وأهميته

يهدف هذا البحث إلى دراسة طرائق تخبئة صفحات الويب، ومدى تأثيرها في تحسين سرعة وصول المستخدمين الى بياناتهم عبر الشبكة الحاسوبية ،ومن ثمّ تحسين أداء الشبكة الحاسوبية كلّها، وتم التركيز في هذا البحث على طريقة مخبأ الوكيل. وتظهر أهمية البحث من خلال تحسين أداء الشبكة الحاسوبية وتقليل التأخيرات الزمنية للوصول الى الشبكة الحاسوبية والتحكم في عرض المجال المتوافر في الشبكة الحاسوبية، كما يخفف الحمل عن مخدمات الويب.

3-آلية عمل تخبئة محتوى الويب:

يوضّح الشكل (1) آلية عمل تخبئة محتوى الويب عند طلب موقع معين مثل www.yahoo.com، اذ يجري في البداية تفحص شرط هل كان هذا المحتوى قابلاً للتخبئة؟ فإذا لم يكن كذلك تجري العودة إلى المخدم الأساسي(أي مخدم موقع yahoo)، ومن ثم تعاد الصفحة الرئيسة لموقع yahoo للمستخدم النهائي.

أمّا إذا كان المحتوى قابلاً للتخبئة، فيجري الرجوع إلى ذاكرة التخبئة كاش، والبحث عن إمكانية وجوده ضمنها وهنا لدينا احتمالان: إمّا أن المحتوى موجود في الكاش، ومن ثمّ يجلب من مخدم الكاش، ومن ثم تعاد الصفحة الرئيسة الى الموقع المطلوب إلى المستخدم النهائي، أو ألاّ يكون المحتوى موجوداً في الكاش، ومن ثمّ يجلب المحتوى من المخدم الأساسي، ومن ثمّ توضع نسخة عنه في مخدم الكاش، وتعاد الصفحة الرئيسة الى المستخدم النهائي، في هذه الحالة إذا طلب المستخدم هذه الصفحة مرة أخرى يمكنه الوصول إليها بكل سهولة وسرعة عن طريق الكاش.

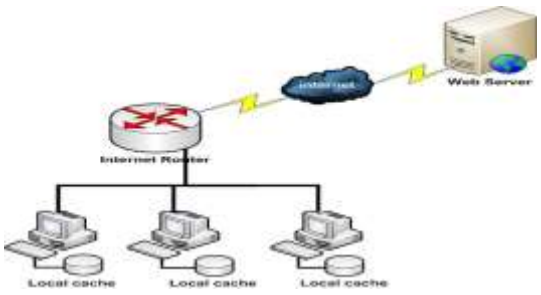
الشكل (1) آلية عمل تخبئة محتوى الويب[7]

4-طرائق تخبئة محتوى الويب:

توجد طرائق عدّة لتخبئة كائنات الويب وهي:

4-1- مخبأ متصفح الويب (مخبأ المستخدم) browser cache

يقوم المستعرض بتخزين مؤقت لكائنات الويب على جهاز المستخدم (أي على القرص الصلب الخاص بحاسوب المستخدم، كما هو مبيّن في الشكل (2)،

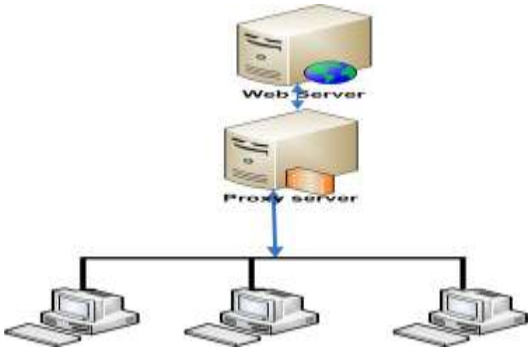


الشكل (2) عملية تخبئة محتوى الويب عن طريق استخدام مخبأ المستخدم

وعند طلب المستخدم لأي رابط، يجري في البداية البحث عن الكائنات الموجودة في الكاش الخاصة به قبل طلبها من موقع الويب، وهذا بدوره يؤدي إلى تسريع تصفح الويب. وتعدّ هذه الطريقة فعّالة من ناحية أن المستخدمين عادة ما يقومون بالدخول إلى نفس المواقع تقريباً معظم الوقت ومن ثمّ يجري الرجوع إلى الكاش غالباً بدلاً من الرجوع الى المخدم الأساسي. فعلى سبيل المثال يستخدم معظم المستخدمين الموقع google.com فإذا ما خزّن الشعار، وقوائم التصفح الخاصة بالموقع في مخبأ المتصفح الخاص بالمستخدم، عندها يستطيع المتصفح جلبهم من الكاش فوراً دون الحاجة إلى طلب هذه المكونات من المخدم الأساسي، الذي يعدّ أسرع كثيراً من جلبها من الموقع المحدد.

4-2- مخبأ الوكيل proxy cache:

يُستخدم مخدم الوكيل proxy server بشكل عام للسماح للمستخدمين بالوصول إلى الانترنت عبر الجدار الناري. ولأسباب أمنية تقوم الشركات بتشغيل نوع خاص من مخدمات HTTP تسمى Proxy على جهاز الجدار الناري الخاص بهم. اذ يقوم المخدم الوكيل بمعالجة الطلبات ضمن الجدار الناري عن طريق إعادة توجيهها إلى المخدمات البعيدة، ومن ثم يعيد إرسال الإجابة إلى العملاء، ونظراً الى أنّ طلبات المستخدمين جميعها تمر عبر المخدم الوكيل، وكذلك نظراً الى أنّ كائنات الويب جميعها أيضاً تمر عن طريق الوكيل، فإذا قام مستخدم معين بطلب مستندات محددة من الويب عن طريق المخدم الوكيل، فلم لا يتم تخزين نسخة من هذه المستندات ليتم استخدامها لاحقاً عند طلبها مرة أخرى من قبل مستخدم آخر. وبهذه الطريقة تقلّل التأخيرات الزمنية اللازمة للوصول إلى المستندات فضلاً عن توفير في عرض المجال المتوافر في الشبكة.



الشكل (3) عملية تخبئة محتوى الويب عن طريق استخدام مخدم وكيل)

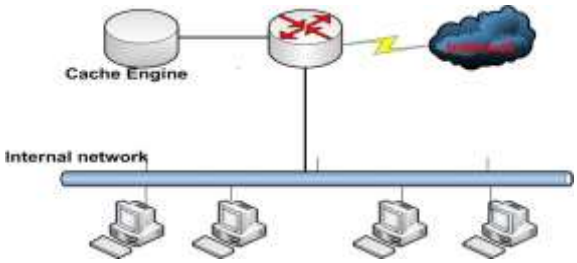
4-3- ذاكرة مخبأ الوكيل العكسية: Reverse Proxy caching

يتم في هذا النوع المكونات بالقرب من المخدم تخزّن توي عليها بدلاً من تخزينها بالقرب من العميل بهدف تخفيف الحمل على المخدمات، وتعدّ هذه الطريقة فعّالة من أجل المخدمات التي تتوقع عدداً كبيراً من الطلبات، وتريد تأكيد على جودة الخدمة لكل عميل، ويبيّن الشكل (4) طريقة مخبأ الوكيل العكسية.

الشكل (4) المخطط الصندوقي لذاكرة مخبأ الوكيل العكسية

4-4- ذاكرة مخبأ العميل الشفافة: Transparent proxy Caching

يتطلب استخدام ذاكرة مخبأ العميل العادية إعداد المستعرض بشكل مناسب حتى تجري عملية التخبئة بشكل صحيح، بينما في هذا النوع من التخبئة يتم اعتراض طلبات مستعرض الويب دون أن يكون للمستعرض دور في هذه العملية. عادةً توضع ذاكرة المخبأ الشفافة عند البوابة الافتراضية حتى تمر طلبات الويب كلّها تلقائياً عبر الوكيل (الملقم)، كما هو مبين بالشكل (5).



الشكل (5) ذاكرة مخبأ العميل الشفافة

يطبّق هذا النوع من التخبئة ضمن مستويين: إمّا مستوى المبدل switch level أو على مستوى الموجه Router level، اذ يتم في مستوى الموجه استخدام التوجيه المعتمد على سياسة معينة policy-based routing لتوجيه الطلبات إلى أماكن التخزين المناسبة، بما يُمكّن من ربط الطلبات القادمة من عملاء معينين بذاكرة تخزين محددة .

أمّا في مستوى المبدل فيعمل المبدل كموازن للحمل، وتعدّ هذه المنهجية فعّالة جداً لأنّها تؤدي إلى تقليل الحمل المترتب عادة على التوجيه المعتمد على السياسة فضلاً عن أنّها(أي الموجهات) تضيف تكاليف إضافية الى التصميم كون الموجهات تعدّ أغلى ثمناً من المبدلات.

5- بروتوكولات تخبئة محتوى الويب:

يستخدم الويب كاش بروتوكولات تسمى بروتوكولات تخبئة محتوى الانترنت Internet Cache protocols لتبادل المعلومات عن كائنات الويب التي تخبّأ عن طريق هذه البروتوكولات اذ تستخدم ذاكرة التخبئة هذه المعلومات لاتخاذ قرار عن المكان الذي يجري منه استرجاع كائن الويب، فمن الممكن أن يكون الحصول على الكائن من ذاكرة التخزين القريبة أفضل من الحصول عليها من موقع الانترنت، وعليه تستخدم بروتوكولاتu]~m من أجل تنظيم عملية التخزين في ذاكرة الكاش مثل ICP(Internet Cache Protocol) وHTCP (Hyper Text Caching Protocol)...الخ.

5-1- بروتوكول ICP (Internet Cache Protocol):[3,4]

يُستخدم بروتوكول ICP من أجل تنسيق مخابئ الانترنت، والغرض الأساسي منه هو إيجاد المكان الأكثر ملاءمة لاسترجاع الكائن المطلوب في حالة استخدام عدة ذواكر كاش في الموقع نفسه بهدف استخدام الكاش بشكل أكثر فعالية وتقليل عدد الطلبات القادمة إلى المخدم الأصلي.

5-1-1- كيفية تطبيق بروتوكول ICP:[3,4]

تحدثنا عن كيفية تطبيق هذا البروتوكول في رزم مخدم الويب الخاص بـSquid اذ تنفّذ الخطوات الآتية:

تستقبل ذاكرة التخبئة المحلية طلب HTTP من عميل الكاش ثم ترسل ذاكرة التخبئة المحلية طلبات ICP، ومن ثم تستقبل ذواكر التخبئة المقابلة الطلبات من ذاكرة التخبئة

المحلية وترسل ردود ICP وأخيراً تستقبل ذاكرة التخبئة المحلية، الردود، وتقرر إلى أين سيوجّه الطلب.

5-2- بروتوكول HTCP:[1,2]

يستخدم بروتوكول "HTCP Hyper Text Caching Protocol" في استكشاف ذواكر التخبئة الخاصة بـHTTP، والبيانات المخبئة cached data، وفي إدارة مجموعة من ذواكر تخبئة HTTP، وفي مراقبة نشاط الكاش.

يسمح هذا البروتوكول باستخدام ترويسات كاملة للطلبات والردود في إدارة الكاش، وكذلك بتوسعة مجال إدارة الكاش ليتضمن مراقبة إضافة ذواكر التخزين البعيدة أو حذفها. وطلب إجراء حذف عاجل، أو إرسال تلميح عن كائنات الويب، مثل مواقع الطرف الثالث third party للكائنات المخبئة، أو عدم توفر بعض كائنات الويب.

5-3- بروتوكول نقل النصوص التشعبية HTTP:

يوفر هذا البروتوكول آليتين لمساعدة الكاش في تحديد زمن الصلاحية للكائن. الأولى حقل الصلاحية Expires في الترويسة، اذ يمكن للمخدم توفير وقت وتاريخ معين يمكن عدّه الكائن بعده مهملا وبحاجة للتحديث من المخدم الأساسي. أمّا الآلية الثانية فهو حقل "آخر زمن تعديل" (last modified time)، وهو الزمن الذي تم فيه إمّا إنشاء الكائن أو تعديله، فإذا لم يخزّن الكائن مدة طويلة فمن المنطقي عندها عدّ أنّ الكائن لن يتغير في المدة القريبة القادمة.

5-3-1- طرائق التخبئة في البروتوكول HTTP (HTTP caching methods):

كما ذكرنا فانّه عند القيام بعملية تخبئة لكائن محدد (صورة معينة أو شعار مثلاً) تحفظ نسخة من هذا الكائن في ذاكرة التخبئة ليتم استرجاعه لاحقاً عند طلبه مرة أخرى، وهنا تواجهنا مشكلة كبيرة، وهي في حال غيّرت هذه الصورة أو الشعار، واسترجعت النسخة القديمة من ذاكرة التخبئة لتظهر الصفحة المنتهية الصلاحية بدلاً من الصفحة الحديثة، لذلك كان لابدّ من البحث عن طرائق لحل هذه المشكلة نذكر منها الطرائق الآتية:

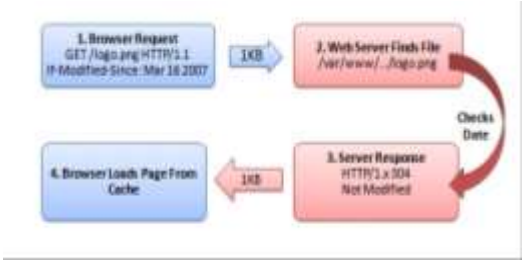
1- الطريقة الأولى: آخر تعديل Last modified:

يقوم المخدم بإخبار المستعرض بإصدار الملف الذي أرسل (وليكن شعار مثلاً باسم logo.png) كما يأتي:

Last-modified: Fri, 16 Mar 2007 04:00:25 GMT

File Contents (could be an image, HTML, CSS, JavaScript...)

وعليه يعلم المستعرض بأنّ تاريخ إنشاء الكائن logo.png هو 16 آذار 2007، وعندما يحتاج المستعرض هذا الشعار يجري فحصاً خاصاً مع المخدم الأصلي قبل استرجاع الكائن من الكاش والشكل (6) طريقة استرجاع كائن من الكاش بعد فحص تاريخ التعديل..



الشكل (6) طريقة استرجاع كائن من الكاش بعد فحص تاريخ التعديل [5]

2- الطريقة الثانية: المعرّف ETag

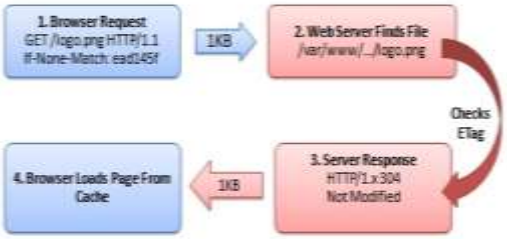
إن طريقة مقارنة تاريخ التعديل جيدة ،ولكنها قد تقود إلى بعض المشكلات في بعض الحالات، فمثلاً من الممكن أن تكون ساعة المخدم الأصلية خطأ ومن ثم تعدّل، كذلك من الممكن أن يحدث التوقيت الصيفي أو

الشتوي باكراً، ولا يتم تحديث المخدم... عندها تصبح النسخة المخبأة في ذاكرة التخبئة غير دقيقة.

ETag هو عبارة عن معرف فريد يعطى لكل ملف يمثل بصمة لكل ملف، وعند تغيير أي جزء من الملف تتغير هذه البصمة كذلك. لذا فبدلاً من إرسال زمن التعديل يقوم المخدم بإرسال البصمة ETag

ETag: ead145f
File Contents (could be an image, HTML, CSS, JavaScript...)

قد تكون ETag أي سلسلة تعرف الملف بشكل فريد. وفي مثالنا عندما يريد المستعرض logo.png يحدث السيناريو الآتي:



الشكل (7) طريقة استرجاع كائن من الكاش بعد فحص ETag

1-يقوم المستعرض بطلب logo.png إذا لم يجد أي شيء يطابق البصمة "ead145f" tag

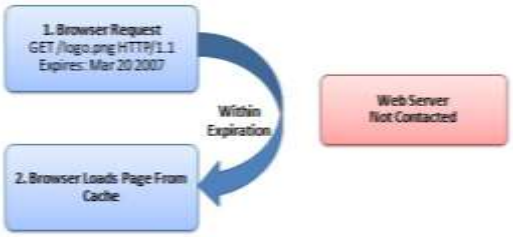
2-يقوم المخدم بفحص بصمة الكائن logo.png.

3-يجد المخدم أنّ الإصدار المطلوب من البصمة لم يتغير، ويخبر المستعرض بذلك.

4-يقوم المستعرض باسترجاع النسخة المخبأة في ذاكرة الكاش.

3- الطريقة الثالثة: الصلاحية Expires

إن تخبئة ملف والتفحص مع المخدم هو أمر جيد، ولكن يعاني من مشكلة واحدة وهي أننا نقوم بالتفحص مع المخدم في كل مرة. تحلّ هذه المشكلة بما يسمى "تاريخ انتهاء الصلاحية" expiration date. فمثلاً إذا علمنا متى ستنتهي صلاحية الكائن logo.png نستمر باستخدامها حتى انتهاء هذه الصلاحية، وحالما تنتهي هذه الصلاحية يقوم المستعرض بالاتصال مع المخدم لطلب نسخة حديثة بتاريخ صلاحية جديد، يصبح شكل الترويسة، والشكل (8) يوضّح ذلك



الشكل (8) استرجاع كائن من الكاش عن طريق تفحص تاريخ انتهاء الصلاحية

في هذه الحالة لا يتحدث المستعرض مع المخدم ولكنه يتفحّص تاريخ انتهاء الصلاحية للكائن فإذا تحقق ذلك يستعرض النسخة المخزنة في ذاكرة الكاش.

4-الطريقة الرابعة: العمر الأعظمي Max-Age

الطريقة السابقة جيدة، ولكن حساب التاريخ معقد في كل مرة. أمَّا في ترويسة العمر الأعظمي يبسّط الحساب وذلك مثلاً بتحديد عمر صلاحية ذي مدة محددة (أسبوع واحد مثلاً) بذلك تنتهي صلاحية الملف بعد أسبوع واحد بدءًا من اليوم وهذا أبسط بكثير من تحديد تاريخ معين.

6- DNS (Doman Name Server) caching: [5]

تؤدي عملية تخبئة معلومات الـDNS دوراً مهماً في تحسين أداء الشبكة، لذا كان لابدَّ من الخوض في هذا الموضوع واستعراضه بقليل من التفصيل.

6-1-استعلام نظام اسم النطاق DNS query:

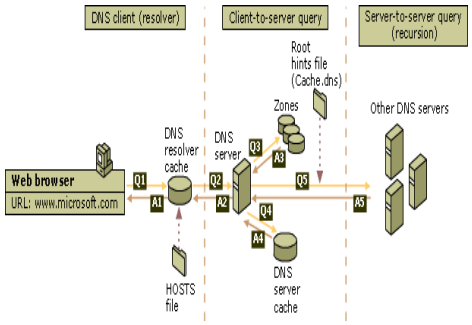
عندما يريد عميل DNS البحث عن اسم ما يقوم بالاستعلام من مخدم الـDNS لإيجاد الاسم، اذ تحتوي كل رسالة استعلام query يقوم العميل بإرسالها على ثلاث معلومات أساسية: اسم مجال DNS المؤهل بالكامل Fully qualified Domain Name (FQDN).

1- نوع الاستعلام المطلوب.
2- صنف محدد لاسم مجال DNS.

بشكل عام تحدث عملية استعلام DNS على جزءين:

1- تبدأ عملية الاستعلام عن الاسم عند حاسوب العميل ومن ثمّ تمرّر الى المحلل resolver ومنه الى خدمة عميل DNS من أجل عملية القرار (الحل) resolution.
2- عندما لاتحلّ الاستعلام محلياً يمكن الاستعلام من مخدمات DNS لحل الاسم عند الحاجة.

شرح السيناريو السابق في الشكل (9):

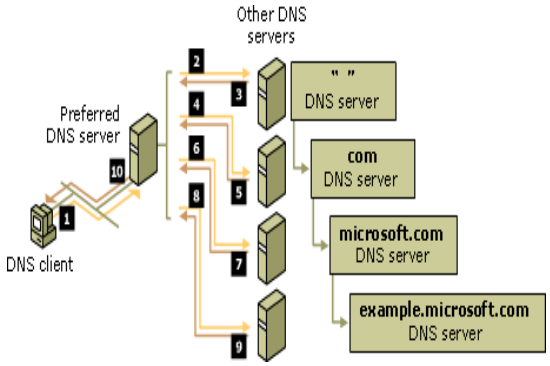


الشكل (9) طريقة حل الاسم محلياً local resolver [5]

يجري في الخطوات المبدئية من عملية الاستعلام استخدام اسم مجال الـDNS في البرنامج الموجود على الحاسب المحلي، ومن ثمّ يمرّر الطلب إلى خدمة عميل الـDNS (DNS client service)، حتى يتم التحليل باستخدام المعلومات المخزنة في ذاكرة الكاش، فإذا كان الاسم المستعلم عنه موجوداً يتم إجابة الاستعلام وتنتهي العملية.

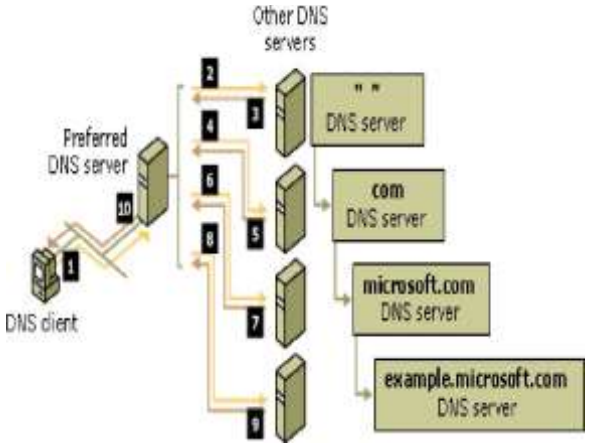
ويمكن أن تحصل ذاكرة كاش التحليل المحلية على معلومات الاسم المطلوبة من مصدرين أساسيين:

- إذا أعدّ ملف hosts محلياً: يمكن مقابلة كل عنوان مضيف host باسمه ضمن ملف معين يعاد تحميله إلى الكاش عندما تبدأ خدمة عميل DNS(DNS client service) بالعمل.
- تضاف سجلات المصادر التي تم الحصول عليها من الإجابات السابقة لاستفسارات DNS إلى ذاكرة الكاش حيث تبقى مدة من الزمن. إذا لم يطابق الاستعلام أي مدخل في ذاكرة الكاش تكمل عملية التحليل بالاستعلام من مخدم الـDNS لإيجاد الاسم المطلوب. عندما يستقبل مخدم DNS الاستعلام يحاول في البداية الاجابة عنه بالاعتماد على معلومات سجل المصدر الأصلية المحتواة في منطقة الاعدادات المحلية على المخدم، فاذا طابق الاسم المستعلم عنه سجل المورد المتعلق به في هذه المنطقة يقوم المخدم بالاجابة عن الطلب باستخدام هذه المعلومات لحل الاسم المطلوب. أمّا اذا لم تتوافر معلومات من اجل الاسم المطلوب في المنطقة فيقوم المخدم عندها بالبحث عن امكانية حل الاسم بالاعتماد على المعلومات المخزنة محلياً من قبل الاستعلامات السابقة. فاذا وجد تطابقاً يجيب المخدم على الطلب وفق هذه المعلومات. كذلك اذا استطاع المخدم المفضل(ذو الأولوية العليا) اجابة الاستعلام برد "تطابق ايجابي" positive matched response من ذاكرة الكاش المحلية الخاصة به عندها تنتهي عملية الاستعلام. أمّا إذا لم يجد الاسم المطلوب معلومات مطابقة في مخدم DNS المفضل يتم اكمال عملية الاستفسار باستخدام العودية recursion من أجل حل الاسم بشكل كامل، وهذا يتطلب مساعدة من مخدمات DNS أخرى للمساعدة في حل الاسم، كما هو موضّح بالشكل (10)



الشكل (10) طريقة حل الاسم بالطلب الى مخدم DNS[5]

بشكل افتراضي تقوم خدمة عميل DNS بالطلب الى المخدم استخدام عملية العودية من أجل حل الأسماء بشكل كامل بالنيابة عن العميل، وذلك قبل إعادة الإجابة، في الواقع في معظم الحالات يتم إعداد مخدم DNS افتراضياً ليدعم عملية العودية، كما هو موضّح في الشكل (11)الآتي:



الشكل (11) طريقة حل الاسم باستخدام عملية العودية [5]

حتى يتمكن مخدم DNS من إجراء عملية العودية اجراءَ صحيحاً يحتاج في البداية إلى بعض معلومات الاتصال المساعدة التي تتعلق بمخدمات DNS الأخرى ضمن فضاء تسمية مجال DNS(namespace)، حيث يتم توفير هذه المعلومات على شكل تلميحات الجذر (root hints) التي هي عبارة عن قائمة من سجلات الموارد الأولية التي يمكن استخدامها عن طريق خدمة DNS لتحديد مواضع مخدمات DNS الأخرى، والتي تكون موثوقاً بها بالنسبة الى جذر شجرة فضاء تسمية مجال DNS. وباستخدام تلميحات الجذر من أجل ايجاد المخدمات الجذرية يصبح مخدم DNS قادراً على استخدام عملية العودية. نظرياً تسمح هذه العملية لأي مخدم DNS بايجاد المخدمات الموثوق بها بالنسبة الى أي اسم مجال DNS المستخدم في أي مستوى في شجرة فضاء الأسماء.

6-2- آلية عملية التخبئة:[5]

نظراً الى أن مخدمات DNS تعالج استعلامات العميل باستخدام العودية أو التكرارية فإنّها تستطيع الحصول على معلومات مهمة عن فضاء أسماء DNS عندها تخزّن هذه المعلومات في ذاكرة كاش المخدم. تعدّ عملية التخبئة طريقة فعّالة لتسريع أداء عملية تحليل الـ DNS الخاصة بالاستعلامات الجزئية للأسماء المعروفة، وهي بذلك تقلل من الازدحام الخاص باستعلامات DNS على الشبكة.

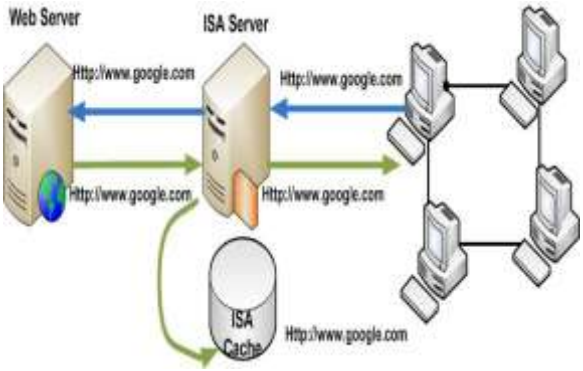
نظراً الى ان مخدمات الـDNS تجري الاستعلامات العودية عوضاً عن العملاء، فهي تقوم بعملية تخزين مؤقت لسجلات الموارد resource record (RR) التي تحتوي على معلومات تم الحصول عليها من مخدمات DNS والتي تكون موثوقاً بها بالنسبة الى أسماء مجال DNS والتي تم تعلمها عند القيام بالاستعلام العودي من أجل البحث وإجابة الاستعلام العودي المنفذ بالنيابة عن العميل.

لاحقاً عندما يطلب عميل آخر استعلاماً جديداً يطلب فيه معلومات تطابق الـRR المخبئة، يقوم مخدم الـDNS باستخدام المعلومات المخبئة في RR لاجابة العميل.

عند ما تخبّأ المعلومات تطبّق قيمة زمن الحياة Time-To-Live (TTL) على كل الـRR المخبئة، ونظراً الى أنّ TTL للـRR المخبئة لا تنتهي صلاحيته ابداً، يمكن لمخدم الـDNS اكمال تخبئة واستخدام RR مرة

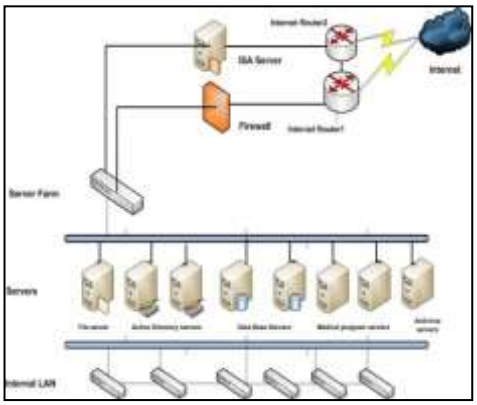
أخرى عند اجابة الاستعلامات المطلوبة من العملاء التي تطابق هذه الـRR.

7- بيئة التطبيق:



تم تطبيق تخبئة محتوى الويب باستخدام نموذج مخبأ الوكيل proxy cache على بيئتي شبكيتين مختلفتين، اذ تمت مراقبة الشبكة الأولى مدة 12 شهراً، واستخلصت البيانات ومقارنتها ضمن مخططات بيانية تستعرض تأثير أداء الكاش في أداء الشبكة كلّها. ومن ثم طبّق البحث على بيئة شبكية جديدة تحتوي على مخدم وكيل مطور عن المخدم الوكيل الموجود في الشبكة الأولى وجمعت البيانات الاحصائية الخاصة بحركة المرور (traffic) على الشبكة لتوضيح دور تخبئة محتوى الويب في تقليل الازدحام على الشبكة. يوضّح الشكل (12) المخطط الصندوقي للشبكة الأولى التي تتضمن مخدم ISA Server يحتوي ضمنياً على كاش داخلية.

- مبدلات switches: موزّعة على ثمانية طوابق بهدف التوصيل الفيزيائي للأقسام جميعها بالشبكة الداخلية فضلاً عن مبدل من الطبقة الثالثة لربط المخدمات مع الشبكة الداخلية.
- مخدمات servers: كل منها تخدم وظيفة معينة (متحكم بالمجال، وقواعد بيانات، وبرنامج مضاد الفيروسات،...).
- جدار ناري: يسمح برفض اتصالات معينة أو قبولها وفق قواعد موضوعة مسبقاً.
- نظم التخزين الاحتياطي backup systems: (أشرطة التخزين المغناطيسية Tapes و SAN storage..) هدفها التخزين الاحتياطي لاستعادة البيانات في حال فقدانها.
- موجهات الانترنتInternet Routers: تسمح باتصال الشبكة الداخلية مع شبكة الانترنت، حيث يتوفر موجهان أحدهما متصل بالشبكة الداخلية عن طريق الجدار الناري، والآخر عن طريق المخدم الوكيل.



الشكل(12) المخطط الصندوقي للبنية الشبكية الأولى

7-2- مخدم مايكروسوفت للتسريع وأمن الانترنت

Microsoft Internet Security and Acceleration (ISA) Server

يدعم مخدم ISA التخزين المؤقت لمحتويات الويب كطريقة لتحسين سرعة استرجاع المعلومات عن طريق الانترنت، وذلك من وجهتي نظر كل من: المستخدم للشبكة الداخلية –اذ يحسن الكاش من أداء الوصول إلى الانترنت– ومن وجهة نظر مسؤول الشبكة network administrator –حيث يقلل الكاش من استخدام عرض المجال المتوافر في الشبكة network bandwidth- حيث يتم تخزين نسخة من الطلبات القادمة في ذاكرة

التخبئة الخاصة بالمخدم ليتم الرجوع إليها لاحقاً عند طلبها مرة أخرى.

يتضمن مخدم ISA مرشح إعادة توجيه بروتوكول نقل النصوص الترابطية HTTP Filter الذي يسمح للجدار الناري ولعملاء secureNAT بالافادة من ميزات التخزين المؤقت، اذ يمكن عند تفعيل هذا المرشح تخزين طلبات الويب القادمة من الجدار الناري فضلاً عن الطلبات القادمة من عملاء secureNAT.

7-2-1- طريقة عمل مخدم ISA Server:

يقوم مخدم ISA بتحليل قواعد التوجيه وإعدادات التخبئة، ومحتويات ذاكرة التخبئة الموجودة حالياً، لتحديد هل سيقوم باستعادة الكائن من الكاش أم لا؟ في البداية إذا كان مسموحاً للمستخدم بطلب موقع ويب يتفحص مخدم ISA فيما إذا كان الكائن متوفراً في الكاش، فإذا لم يكن متوفراً عندها يتفحص المخدم الحدث المرتبط بالتوجيه routing rule's action ليحدد كيف سيوجّه الطلب. أمّا إذا كان الكائن موجوداً في الكاش يقوم المخدم بتنفيذ الخطوات الآتية:

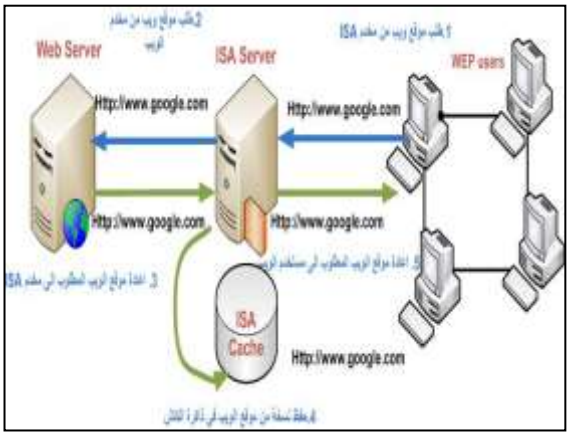
يتفحص المخدم صلاحية الكائن. فإذا كان الكائن صالحاً يستعيد المخدم هذا الكائن من ذاكرة التخبئة، ويعيده إلى المستخدم.

وإذا كان الكائن غير صالح عندها يتفحص المخدم قاعدة التوجيه المناسبة، فإذا تم إعداد قاعدة التوجيه بحيث يتم استعادة أي إصدار من الكائن، عندها يقوم المخدم باستعادة الكائن غير الصالح من الكاش. وإذا أعدّت قاعدة التوجيه بحيث يتم توجيه الطلب، عندها يحدد مخدم ISA إلى أين سيوجّه الطلب .هل هو إلى مخدم آخر، أم إلى مخدم الويب المطلوب من قبل المستخدم. إذا كان مخدم الويب غير قابل للوصول عندها يتفحص مخدم ISA إذا تم إعداد المخدم بحيث يعيد كائنات منتهية الصلاحية من ذاكرة الكاش عندها يمكن إعادة الكائنات من الكاش. أمّا إذا كان موقع الويب متوفراً، يحدد مخدم ISA هل كان الكائن قابلاً للتخبئة، وفيما إذا أعدّت قاعدة التوجيه ليتم تخبئة الاستجابة، فإذا كانت كذلك يخزن مخدم ISA نسخة من الكائن ومن ثم يعيد الكائن للمستخدم.

يمكن اختصار الخطوات السابقة بالآتي:

يقوم المستخدم بطلب موقع ويب من مخدم ISA وليكن مثلاً www.google.com، ثم يقوم مخدم ISA بطلب الموقع المطلوب من مخدم الويب Web server وبعدها يعيد مخدم الويب الصفحة المطلوبة إلى مخدم ISA، ثم يعيد مخدم ISA الصفحة المطلوبة للمستخدم النهائي مع تخزين نسخة منها في ذاكرة التخبئة (cache) الخاصة به

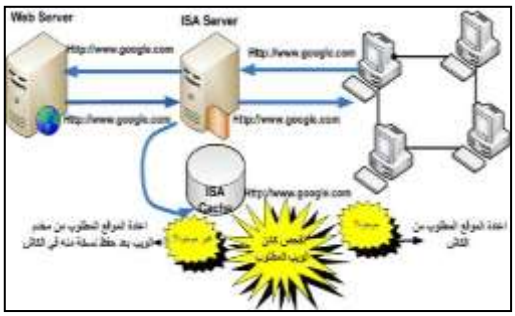
وعند طلب المستخدم للصفحة نفسها مرة أخرى يقوم مخدم ISA باسترجاعها من ذاكرة التخبئة دون الرجوع للمخدم الأساسي (وذلك وفق شروط معينة).



الشكل(13) طريقة عمل مخدم ISA

في الواقع يوجد نوعان من التخزين المؤقت في مخدم ISA وهما:1.التخزين الأمامي Forward caching:

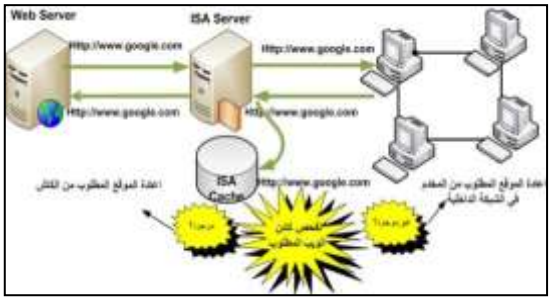
اذ تسرّع الاستجابة للطلبات الصادرة عندما يقوم مستخدم في الشبكة الداخلية بطلب كائن ويب من شبكة الانترنت كما هو موضّح في الشكل (14)



الشكل (14)التخزين الأمامي في مخدم ISA

2.التخزين العكسي Reverse caching:

اذ تسرّع الاستجابة القادمة من مستخدم على شبكة الانترنت يطلب فيها كائن ويباً موجوداً على مخدم ويب في الشبكة الداخلية. كما في الشكل (15)



الشكل(15) التخزين العكسي في مخدم ISA

7-2-2- تطبيق تخبئة المحتوى في مخدم ISA:

يقوم مخدم ISA بتوضيع الكائنات الأكثر استخداماً في الذاكرة RAM، في حين يخزن باقي الكائنات على القرص الصلب، وعندما يخبّأ كائن ما يقوم مخدم ISA في البداية بوضع الكائن في ذاكرة الكاش على RAM، ومن ثم يتم كتابتها إلى القرص الصلب، ويستخدم مخدم ISA 10% من الذاكرة RAM في المخدم لغرض التخزين المؤقت لمحتوى الويب بشكل افتراضي.

ينشأ مخدم ISA ملفاً لتخزين المحتويات على القرص المحدد للتخبئة لكل 10 غيغابايت محجوزة لأغراض التخبئة (أي إذا قام المسؤول بحجز 15 غيغابايت من القرص للتخبئة ينشا ملفان الأول بسعة 10 غيغابايت، والآخر بسعة 5 غيغابايت) وعند تخزين الكائن يقوم مخدم ISA بإلحاقه بملف تخزين المحتوى، فإذا كان الملف ممتلئاً يقوم مخدم ISA بحذف الكائنات الأقدم من ذاكرة الكاش باستخدام معادلة تأخذ بالحسبان عمر الكائن وحجمه.

7-2-3- قواعد عمل الكاش في مخدم ISA: ISA Cache Rules

تسمح قواعد الكاش بتخصيص أنواع المحتويات التي يجب أن تخزن في الكاش، وكيفية المعالجة عند طلب كائن مخزن في ذاكرة التخزين المؤقت. اذ تتضمن المحتويات التي من الممكن تخزينها الآتي: المكونات الديناميكية والمكونات الخاصة بالاستعراض دون اتصال والمكونات التي تحتاج إلى مصادقة من قبل المستخدم من أجل استرجاعها.

طبّقت في هذا البحث القواعد الآتية:

حجم ذاكرة التخبئة10 غيغابايت، ويستخدم 10% من الذاكرة الفيزيائية المتوافرة لغرض التخزين المؤقت وأيضا يخزّن الكائن في الكاش فقط إذا أشارت الترويسة بأن المكون قابل للتخبئة، فضلاً عن تخبئة المحتويات الديناميكية في بعض المراحل، ثم يسترجع الكائن من الكاش فقط في حال وجود إصدار صالح من هذا الكائن في الذاكرة كاش، وإذا لم يتوافر إصدار حديث يوجّه الطلب إلى المخدم بحيث يتم تخبئة استجابة SSL، تم تفعيل تخبئة HTTP.الحجم الأعظمي لطلبات URL المخبئة في الذاكرة هو 12800 بايت، وحدّد زمن بقاء الكائن (TTL) في الكاش صالحاً بـ 20%من عمر الكائن

object age وإذا لم يتم الوصول إلى موقع الويب الذي يحوي كائن الويب المنتهية صلاحيته يرجع الكائن فقط في حال كان زمن البقاء TTL للكائن في الكاش 20% من عمر الكائن اذ انّ زمن البقاء بين دقيقة واحدة وساعتين.

- البيئة الشبكية الثانية: طبّقت تخبئة محتوى الويب ضمن المخدم الوكيل المتوافر في الشبكة وهو (TMG) forefront Threat Management Gateway Server (الذي هو عبارة عن نسخة مطورة من مخدم ISA Server)، ويوضّح الشكل 3-5 المخطط الصندوقي لهذه الشبكة اذ تتكون من:

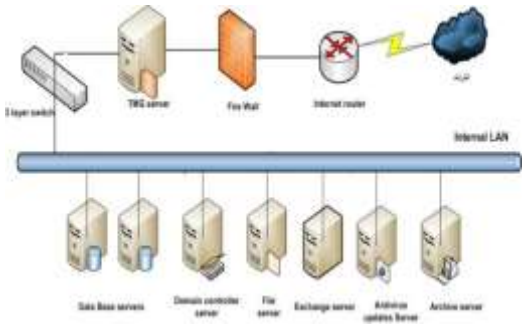
مخدمات Servers: تخدم وظائف معينة (مخدمات قواعد البيانات، متحكم مجال، مخدم تبادل الملفات، مخدم تبادل الرسائل، مخدم للأرشفة الالكترونية،...) فضلاً عن مخدم وكيل proxy server.

مبدلات Switches: عبارة عن مخدمات من الطبقة الثانية والثالثة تربط الشبكة الداخلية فضلاً عن مخدم من الطبقة الثالثة لربط المخدمات مع الشبكة الداخلية.

جدار ناري Fire wall: يستخدم لمراقبة العمليات التي تمر على الشبكة ويرفض أو يقر أحقية المرور ضمن قواعد معينة.

موجه انترنت Internet router: يسمح لمستخدمي الشبكة الداخلية بالنفاذ الى الانترنت عن طريق المخدم الوكيل.

1- نظم تخزين احتياطي: (NAS Storage) تسمح بتخزين نسخة عن البيانات المهمة في الشبكة.



الشكل (16) المخطط الصندوقي للشبكة الثانية

7-3- مخدم Threat Management Gateway Server TMG:[6]

هو نسخة مطورة من مخدم ISA يعمل ضمن بيئة نظام التشغيل Windows server 2008 ويتميز بالعديد من الميزات نذكر منها:

1- ميزات التوجيه والنفاذ عن بعد Routing and remote access

يمكن أن يعمل مخدمTMG كموجه، بوابة انترنت افتراضية، مخدم الشبكة الافتراضية الخاصةVPN، مخدم ترجمة عنوان الشبكة NAT فضلاً عن عمله كمخدم عميل proxy server.

2- ميزات أمنيةsecurity features: يعمل مخدم TMG كجدار ناري firewall الذي يمكّنه من تفقد حركة المرور على الشبكة مثل محتويات الويب ومحتويات الويب الآمنة والبريد الالكتروني، ومن ثم يقوم بفلترة البرمجيات الخبيثة التي تحاول استغلال الثغرات الأمنية الموجودة ، والمكونات التي لا تطابق السياسة الأمنية الموضوعة مسبقاً. بمعنى تقني يوفر مخدم TMG حماية لطبقة التطبيقات، وعملية فلترة بحالة stateful filtering، وفلترة المحتويات content filtering وحماية ضد البرمجيات الخبيثة anti-malware.

3- ميزات تحسين أداء الشبكة Network performance: يحسن مخدم TMG أداء الشبكة بعدة

طرائق، مثلاً يمكن أن يقوم المخدم بضغط حركة المرور على الشبكة لتحسين سرعة الاتصال، فضلاً عن تقديم خدمة تخبئة محتوى الويب التي تمكّن من تخزين محتويات الويب التي تم النفاذ إليها بشكل متكرر. وبذلك يمكن للمستخدم النفاذ إليها بطريقة أسرع عن طريق استرجاعها من ذاكرة الكاش المحلية. يمكن لمخدم TMG كذلك تخزين البيانات المستقبلة عبر خدمة نقل المعلومات الذكية Background Intelligent Transfer Service مثل تحديث البرمجيات المنشورة على موقع تحديث مايكروسوفت.

7-3-1- قواعد التخبئة المطبقة في مخدم TMG:

طبّقت قواعد التخبئة التالية على مخدم TMG (مع ملاحظة تطبيق نوعين من القواعد إحداها خاصة بالنفاذ إلى الانترنت والأخرى من أجل عملية تحديث البرمجيات): تم اعتماد حجم ذاكرة التخبئة10 غيغابايت ويستخدم 10% من الذاكرة الفيزيائية المتوافرة لغرض التخزين المؤقت.

يخزّن الكائن في الكاش فقط إذا أشارت ترويستا المصدر والطلب الى أنّ المكون قابل للتخبئة وتخبئة الكائن الذي لم تحدّد فيه زمن آخرو تعديل وتخبئة الكائنات حتى لو لم يكن لديها كود الحالة لبروتوكول HTTP هو 200.

كذلك الحجم الأعظمي للروابط URLs المخزنة في الذاكرة 12800 بايت وعند تعذر الوصول الى موقع الويب الخاص بالكائن منتهي الصلاحية يعاد الكائن من الكاش إذا كان زمن انتهاء الصلاحية أقل من 50% من عمر الكائن الأصلي TTL ولكنه لا يتجاوز 60 دقيقة

وتجري عملية الاستعادة من الكاش فقط عند توافر إصدار صالح من الكائن في الذاكرة، وإلا يوجّه الطلب إلى المخدم الأصلي ويتم تفعيل تخبئة HTTP ما لم يحدد المصدر زمن صلاحية، وتحدّث الكائنات في الكاش اعتماداً على زمن TTL الذي وضع مساوياً لـ20% من عمر الكائن، وبحيث لا يكون أقل من 15 دقيقة ولا أكثر من يوم واحد، وتم تفعيل تخبئة FTP وبحيث يبقى كائن الـFTP صالحاً في الكاش عندما لا يتجاوز عمره يوماً واحداً أي انّ(FTP- TTL= 1 day) وأخيراً يتم تخبئة إجابات SSL.

التطبيق العملي:

طبّق البحث على شبكة تحتوي على مخدم وكيل وهو مخدم ISA، واستخدمنا التجهيزات المتوافرة في الشبكة بمراقبة أداء هذه الشبكة بعد تطبيق قواعد التخبئة التي ذكرناها سابقاً(حيث طبّق التخزين الأمامي) وروقبت الشبكة مدة 12 شهراً لاستخلاص النتائج ورسم المخططات المناسبة، ويبيّن الجدول (1) والمخططات (1) و(2) النتائج التي تم التوصل اليها، وكيف تؤثر الكاش في تحسين سرعة النفاذ للإنترنت ومن ثمّ تحسن من أداء الشبكة كلّها، وقد قيست الأداء اليومي خلال اثني عشر شهراً، وتم أخذ 20 عينة عشوائية من هذه القياسات، مع حساب الوسطي لهذه النتائج، ومن ثم رسم مخطط يوضح النسبة المئوية لاسترجاع البيانات من الكاش مقارنة بالبيانات الكلية المسترجعة من الانترنت.

الجدول (1)

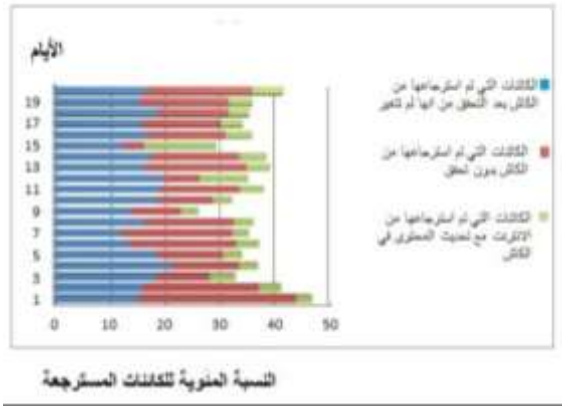
اليوم	النسبة المئوية للكائنات المسترجعة عن طريق الكاش بعد التحقق من أنها لم تتغير	النسبة المئوية للكائنات المسترجعة عن طريق الكاش دون تحقق من التغيير	النسبة المئوية للكائنات المسترجعة من الانترنت مع تحديث المحتوى في الكاش	المجموع الكلي
1	15.1	28.7	2.9	46.7
2	15.9	21.1	4.1	41.1
3	18.3	9.8	4.7	32.8
4	21.1	12.3	3.5	36.9
5	18.6	11.9	3.5	34
6	13.6	19.3	4.2	37.1
7	12.1	20.1	3	35.2
8	16.3	16.2	3.5	36
9	13.6	9.3	3.2	26.1
10	18.4	10.3	3.5	32.2
11	19.1	14.4	4.4	37.9
12	20	6.3	8.8	35.1
13	16.4	18.4	4.3	39.1
14	17.2	16.2	5	38.4
15	12.4	3.8	13.1	29.3
16	16.8	14.1	4.9	35.8
17	15.8	14.2	4.1	34.1
18	18.8	12.8	3.7	35.3
19	15.5	16.1	4.3	35.9
20	16.4	19.4	5.7	41.5
المتوسط	16.57	14.735	4.72	36.3

المخطط (2) الكائنات المسترجعة عن طريق الكاش يومياً مقارنة بالكائنات الكلية المسترجعة عبر الشبكة

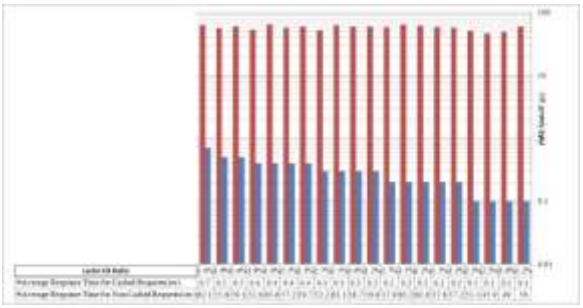
في الخطوة الثانية قورن زمن الاستجابة للمحتويات المسترجعة عن طريق الكاش والمحتويات المسترجعة عن طريق الانترنت عند قيم مختلفة لنسبة الإصابة في الكاش cache hit ratio بعد جمع عشرين قيمة عشوائية مدة أسبوع، وكانت النتيجة كما في الجدول(2).

الجدول(2)

Cache Hit Ratio%	Average Response Time for Non Cached Requests(sec)	Average Response Time for Cached Requests(sec)
1.2%	59	0.1
3.0%	49	0.1
2.3%	45.9	0.1
2.3%	51.1	0.1
1.2%	57.2	0.2
1.5%	57.8	0.2
1.2%	61.6	0.2
1.5%	63.2	0.2
1.7%	57.9	0.2
1.2%	59.6	0.3
1.6%	58.7	0.3
1.5%	63.1	0.3
2.7%	52.2	0.3
1.3%	59.7	0.4
1.5%	57.2	0.4
1.0%	63.6	0.4
1.9%	52.6	0.4
1.6%	59.1	0.5
1.9%	55.8	0.5
1.4%	62.1	0.7



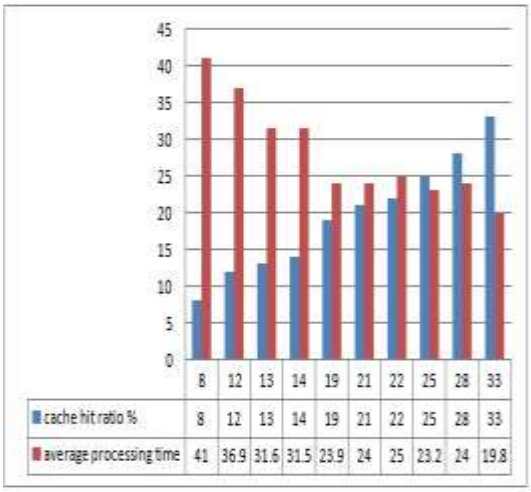
المخطط (1)أداء الكاش يومياً



المخطط (3) مقارنة زمن الاستجابة للمحتويات المسترجعة عن طريق الكاش والمحتويات المسترجعة عن طريق الانترنت

الجدول (3) مقارنة زمن المعالجة بالنسبة الى الزمن الاصابة في الكاش

cache hit ratio %	average processing time
%8	41
%12	36.9
%13	31.6
%14	31.5
%19	23.9
%21	24
%22	25
%25	23.2
%28	24
%33	19.8



المخطط(4) مقارنة زمن المعالجة بالنسبة الى الزمن الإصابة في الكاش

4-2-2- الخلاصة:

ممَّا سبق نستنتج دور استخدام الكاش كعامل مهم من عوامل تحسين أداء الشبكة الذي ظهر جلياً على أرض الواقع، اذ تتجاوز نسبة الكائنات المسترجعة من ذاكرة التخبئة أحياناً 45%، أي ما يقارب نصف مجموع الكائنات المسترجعة عبر الشبكة كلّها)، وهذا بدوره يقلل من الازدحام الموجود على الشبكة فضلاً عن تقليل استخدام عرض المجال المتوافر في الشبكة بهذه النسبة.

بالطبع لا نستطيع إنكار تأثير عوامل أخرى في تحسين أداء الشبكة، ولا نستطيع أيضاً إنكار سلبيات استخدام ذاكرة الكاش في بعض الأحيان، إلاَّ أنَّه من خلال تطبيق هذه الدراسة على أرض الواقع أظهرت تحسناً ملموساً في سرعة وصول المستخدمين الى بياناتهم على شبكة الانترنت والذي هو الغرض الأساسي من هذا البحث.

REFERENCES

[1] : **RFC 2756 ,Hyper Text Caching Protocol (HTCP/0.0), January 2000**

[2] **RFC 2324 , HTCPCP/1.0 , 1 April 1998**

[3] **RFC2186,Internet Cache Protocol(ICP), version2, September1997.**

[4] **RFC 2187, ICP, September 1997**

[5] **http://www.tutorialspoint.com/http/http_caching.htm retrieved from the internet on 28/10/2014**

[6] **http://blogs.technet.com/b/isablog/archive/2010/12/26/reasons-to-migrate-from-isa-server-2006-to-forefront-tmg-2010.aspx/ retrieved from the internet on 4/11/2014**

[7] **Squid internet object cache , http:// Squid.nlanr.net/squid/.**

[8] **http://wiki.squid-cache.org**

مسرد المصطلحات:

تخبئة محتوى الويب	Web caching
سياسات استبدال الكاش	Web caching olicies
محتوى قابل للتخبئة	Cacheable content
مخبأ المتصفح (مخبأ المستخدم)	Browser Cache(usercache)
مخبأ الوكيل	Proxy Cache
ذاكرة مخبأ الوكيل العكسية	Reverse proxy caching
ذاكرة مخبأ الوكيل الشفافة	Transparent proxy caching
التخبئة الأمامية	Forward caching
التخبئة العكسية	Reverse Caching
مخدم	Server
المخدم الأساسي	Origin server
مخدم ويب	Web Server
المخدم الوكيل	Proxy Server
بروتوكول تخبئة محتوى الويب	Internet Cache protocol
خوارزميات تبديل الكاش	Cache replacement algorithms

إيداع البحث	2017/08/21	Received
قبول البحث للنشر	2018/02/12	Accepted for Publ.