مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (4) العدد(7) 2020
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (4) No. (7) 2020

# تخفيف هجوم الحرمان من الخدمة الموزعة باستخدام العتبة الديناميكية

**د. علي أحمد***

**د. محمد محمد****

**م. ازدهار محمد*****

**(تاريخ الإيداع 6/17 /2020. قُبِل للنشر في 7 /10/2020)**

**□ ملخّص □**

هجوم الحرمان من الخدمة الموزعة (DDOS(Distributed Denial Of Service من أخطر الهجمات التي تتعرض لها الشبكات سواء كانت هذه الشبكات محلية مثل شبكات (LAN(Local Area Network أو شبكات الانترنت، وقد ازداد خطر هذا الهجوم في شبكات الانترنت بشكل كبير خاصةً بعد ظهور مفهوم الحوسبة السحابية Cloud Computing وانترنت الأشياء.

تم اعتماد العديد من الطرق من أجل التخفيف والكشف من هجوم الحرمان من الخدمة الموزعة، لما يسببه هذا الهجوم من خسارة مالية كبيرة لمزودي الخدمات (الشركات التي تقوم بتقديم خدمات الحوسبة عبر الانترنت). من أهم هذه الطرق خوارزميات الذكاء الصنعي مثل الخوارزمية الجينية والشبكات العصبونية التي تُدرب على كشف أنماط معروفة من الهجمات، اضافة الى منهجية التحكم بالازدحام التي تهدف إلى مراقبة الشبكة بشكل مستمر من أجل المحافظة على توافرية المخدم أكبر وقت ممكن (تجنب التحميل الزائد للمخدم بالطلبات الواردة من المستخدمين حتى لا يخرج عن الخدمة ).

الخوارزمية المقترحة الخوارزمية الجينية (Genetic Algorithm ) تخفف من هجوم الحرمان من الخدمة الموزعة اعتماداً على منهجية التحكم بالازدحام، تمت الدراسة العملية على محاكي CloudSim باستخدام لغة جافا، وتمت مقارنة النتائج مع نتائج لدراسات مرجعية.

**الكلمات المفتاحية:** الحوسبة السحابية Cloud Computing، هجوم الحرمان من الخدمة الموزعة DDOS، الخوارزمية الجينية Genetic Algorithm، CloudSim.

* أستاذ في قسم هندسة تكنولوجيا الاتصالات – كلية هندسة تكنولوجيا المعلومات والاتصالات – جامعه طرطوس – سوريا

** دكتور محاضر في الجامعة الافتراضية – كلية المعلوماتية والاتصالات – سوريا

***طالبة ماجستير في قسم هندسة تكنولوجيا الاتصالات – كليه هندسة تكنولوجيا المعلومات والاتصالات – جامعة طرطوس – سوريا

مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (4) العدد(7) 2020
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (4) No. (7) 2020

# Mitigate Distributed Denial of Service Attack Using Dynamic Threshold

***Dr.Ali Ahmad**

****Dr.Mohamed Mohamed**

*****Eng.Izdehar Mohamed**

**(Received 17/6/2020. Accepted 7/ 10/2020)**

**□ ABSTRACT □**

The DDOS is one of the most dangerous attacks that face networks, whether these networks are local such as LAN networks or internet networks. The danger of this attack increased very much especially after the concept "cloud computing" appeared and IOT. Many ways are depended to reduce and detect of DDOS, and this because of enormous financial loss that attack caused to services providers. The most important ways in the factitious, intelligence algorithm such as genetic algorithm and neuron networks that works on detect known pattern of attacks. In addition to congestion control methodology that aims to keep server's availability much time, avoid the overload server. The suggested algorithm is the genetic algorithm, which is dependent on congestion control methodology to mitigate DDOS. This practical study is from cloudsim and these results were compared with those of the previous studies.

**Key Words:** cloud computing, DDOS, genetic algorithm, CloudSim.

*Professor, Communication Technology Engineering Department, Information and Communication Technology Engineering, Tartous University, Syria.

** Lecture doctor at Syrian Virtual University, Faculty of Information and Communications, Syria

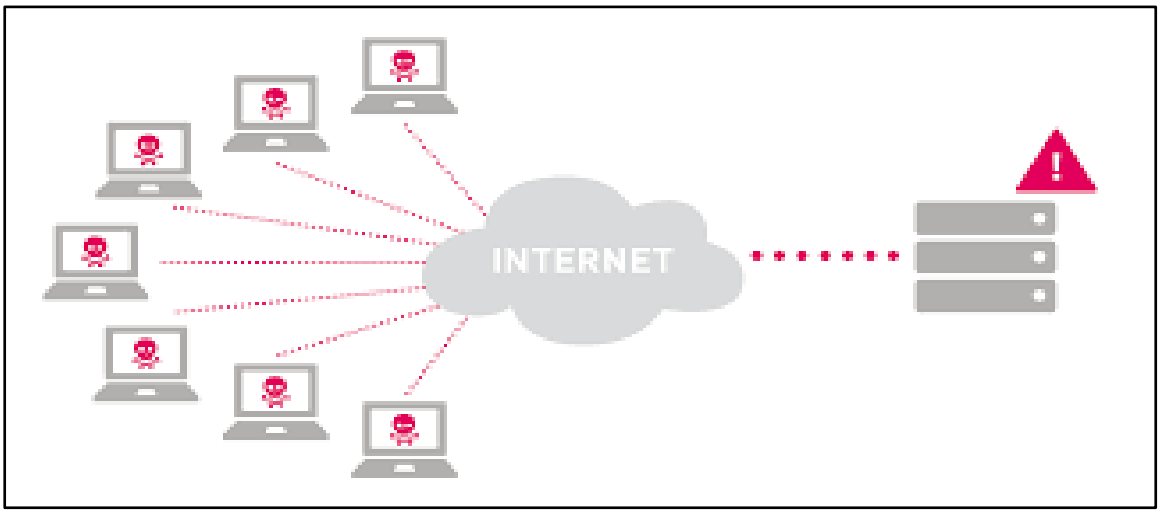
***Master Student at Communication Technology Engineering Department, Information and Communication Technology Engineering, Tartous University, Syria.

## 1- مقدمة:

أصبح الأمن في الانترنت الجانب الأكثر أهمية، وذلك لأنه يتم استخدام عدد كبير من المخدمات لتزويد الخدمات عبر الانترنت ومنها خدمات الحوسبة السحابية [2].

الحوسبة السحابية: هي نموذج لتقديم الموارد الحاسوبية (compute , storage ,...) للعملاء عند الطلب، وذلك عن طريق الانترنت، حيث يتم الدفع مقابل كل استخدام، ويمكن الوصول إلى هذه الخدمات من أي مكان وفي أي زمان ومن أي جهاز يحتوي نظام تشغيل وقادر على الوصول إلى الانترنت، وقد أتاحت هذه الخدمة للعميل عدم استنزاف موارد جهازه، حيث أن بعض المعلومات فقط تخزن بشكل مؤقت في جهاز العميل [2,3,8,6].

أحد أهم مقومات الحوسبة السحابية هي التوافرية، وهجوم الحرمان من الخدمة الموزعة ليس بجديد ولكنه مازال يهدد توافرية خدمات الشبكات السحابية، يهدف هذا الهجوم إلى جعل الهدف خارج الخدمة من خلال التحميل الزائد له بكمية هائلة جداً من الطلبات وذلك من عدة مصادر لذلك سمي بالهجوم الموزع لأن ارسال الطلبات يتم من عدة مصادر وليس مصدر واحد، وبالتالي فإن أي مستخدم يريد الوصول إلى الهدف سوف ينتظر كثيراً وهذا يخالف بنود اتفاقية مستوى الخدمة SLA(Service Level Agreement)، التي تنص على تأمين الخدمات عبر الانترنت ضمن معايير محددة لكل من زمن الانتظار وجودة الخدمة، وفي حال خالفت الشركات المزودة للخدمة معايير الأداء التي تحددها هذه الاتفاقية ستخسر عملائها وبالتالي ستخسر الكثير من الأموال [1,2,3,11,7].




**الشكل(1) : هجوم الحرمان من الخدمه الموزعة [1] .**

ظهرت في السنوات الأخيرة طرق لكشف اختراقات الشبكة، وذلك بجمع معلومات عن الأنواع المعروفة من الهجمات، واستخدام هذه المعلومات لكشف أي محاولة لمهاجمة الشبكة تمثل تهديداً على البيانات أو الموارد، ولكن مهما تكن هذه الطرق مجدية ستفشل في النهاية بسبب ظهور أنواع جديدة (غير معروفة مسبقا) من الاختراقات [1,8].

قامت الدراسات المرجعية بالكشف والتخفيف من الأنماط المعرفة مسبقاً لهجوم الحرمان من الخدمة، أي الأنماط المخزنة مسبقاً في قواعد البيانات، وذلك باستخدام عدة طرق منها: العتبة الثابتة (أي استخدم قيمة واحدة للكشف، في

كل مرة يقوم المستخدمون بطلب الخدمات تبقى قيمة هذه العتبة ثابتة)، خوارزميات الذكاء الصنعي التي تم تدريبها على أنماط معرفه مسبقاً في قواعد البيانات [1,5].

بسبب ظهور أنماط جديدة للهجوم بشكل مستمر وفشل أي نظام في الكشف والتخفيف من الهجوم تم من خلال هذا البحث باستخدام خوارزمية عتبة ديناميكة (حيث في كل مرة يرسل المستخدمون طلباتهم الى السحابة ستقوم الخوارزمية بحساب قيمة عتبة لهذه الطلبات يتم على أساسها التخفيف)، وبالتالي تجنب التحميل الزائد للمخدمات بالطلبات.

## 2- هدف البحث:

يهدف البحث إلى تقديم اطار عمل متكامل يتكون من منهجية وتقنيات للتخفيف من خطورة هجوم الحرمان من الخدمة الموزعة، من خلال الحرص على عدم تحميل مركز البيانات بالطلبات الزائدة، وذلك لتحقيق أعلى درجة من التوافرية لخدمات الشبكات السحابية.

## 3- أهمية البحث:

أصدرت الشركة الرائدة عالميا Netscout Systems المتخصصة في خدمات الأعمال الرقمية المتعلقة بالتوافرية والأداء والأمن، في عام 2018 تقريرها السنوي المتعلق بأمن الحوسبة السحابية.

أوضح التقرير بأن معدل تواتر وتعقيد هجمات الحرمان من الخدمة الموزعة في تزايد عالمياً. وخصوصاً مع ازدياد تبني خدمات الحوسبة السحابية وانترنت الأشياء في بيئة الأعمال.

كان هناك 7.5 مليون هجوم حرمان من الخدمة في عام 2017، وفقاً للبيانات الواردة من نظام ATLAS، وهذا يغطي مايقارب ثلث حركة الانترنت العالمية.

نلاحظ من الاحصائيات السابقة أن هجوم الحرمان من الخدمة الموزعة يهدد أمن الانترنت وحتى الآن لم يتم اقتراح أي نظام قادر على التخفيف من أثره بشكل فعال.

## 4- مواد وطرق البحث:

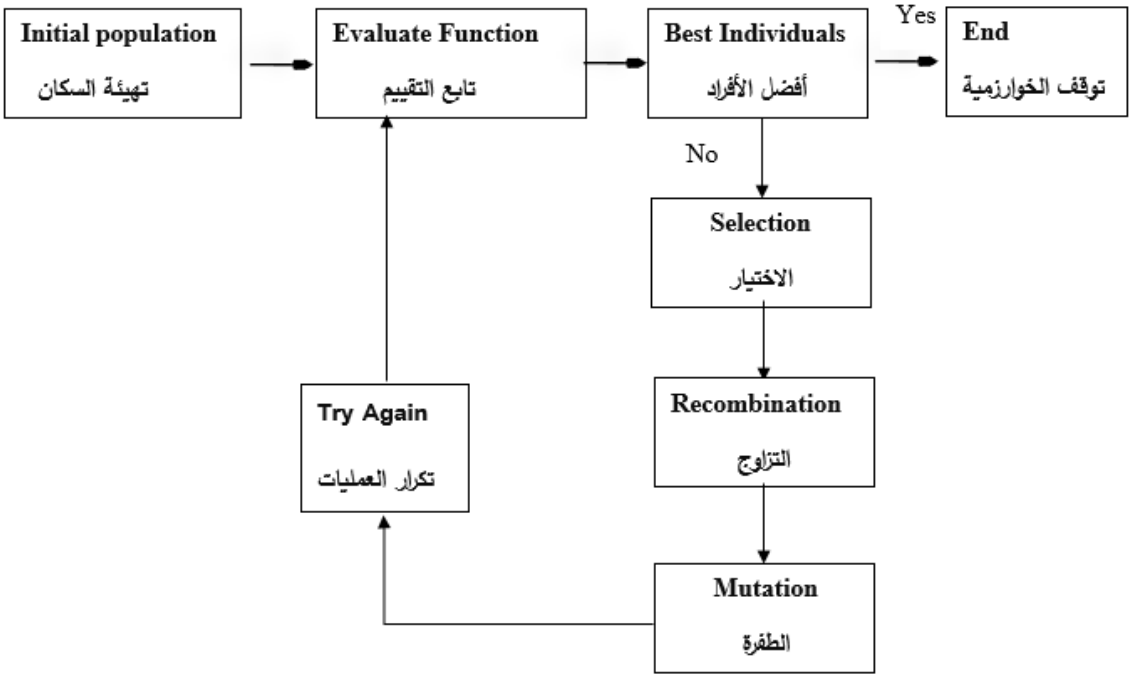
أُنجز هذا البحث اعتماداً على دراسات ومراجع علمية حديثة وعديده تختص في هذا المجال، والتي اعتبرت أن أحد أهم الوسائل للتخفيف من هجوم الحرمان من الخدمة الموزعة بالنسبة للأنماط الغير معرفة مسبقاً هو التخفيف من الازدحام على مراكز البيانات، وقد أُخذت نتائجها و توصياتها بعين الاعتبار[1,5,6,8].

تمت الدراسة العملية على محاكي CloudSim وهو عبارة عن اطار عمل مفتوح المصدر يستخدم في الأبحاث التي تختص بالسحابة.

### 4-1- تعريف الخوارزمية الجينية:

الخوارزمية الجينية هي طريقة من طرق البحث ويمكن تصنيف هذه الطريقه كإحدى طرق الخوارزميات التطورية. تعتبر الخوارزمية الجينيه من التقنيات الهامة في البحث عن الخيار الأمثل من مجموعة حلول متوفره لتصميم معين. وتعتمد مبدأ داروين في الاصطفاء [9].

تم استخدام الخوارزمية الجينية في تطبيقات الأمن بشكل أساسي لايجاد الحلول المثلى لمشكلة معينه، وعادةً ماتبدأ الخوارزمية الجينية بمجموعة عشوائية من الكروموسات، هذه الكروموسات تمثل المشكلة التي يتوجب حلها وذلك وفقاً لخصائص المشكلة، مثلا في مسألة البائع المتجول كانت الكروموسات تعبر عن المسافة بين كل مدينتين، وفي مسألة الثماني ملكات كانت الكروموسات تعبر عن موقع كل ملكة، أما في هذه المسألة تعبر الكروموسات عن عدد الطلبات التي يرسلها كل مستخدم [9].




**الشكل(2): خطوات الخوارزمية الجينية.**

## 4-2- خطوات الخوارزمية الجينية:

يوضح الشكل(2) خطوات الخوارزمية الجينية، وكيف تم توصيفها من أجل الكشف والتخفيف من هجوم الحرمان من الخدمة الموزعة. وقد تم اعتبار متوسط عدد طلبات المستخدميين عتبة مرجعية يتم على أساسها تحديد العتبة الديناميكية استناداً على الدراسة المرجعية [3,10]، حيث اعتبرت مبدأياً أن المتوسط الحسابي هو عتبة مرجعية يتم على أساسها حساب البارامترات الأخرى.

### 4-2-1- تهيئة السكان (Initial Population):

يتم توليد n عدد من السكان بشكل عشوائي، والسكان هم مجموعة من الكروموسات التي تمثل عدد الطلبات الواردة من كل مستخدم.

تـم اسـتخدام التمثيـل الثنـائي (binary)لهـذا الغـرض وذلـك لأن جميـع الدراسـات المرجعيـة التـي تخـتص بالخوارزميـة الجينيـة أكدت أنهـا تعطـي أفضـل النتـائج فـي حـال كـان التمثيـل ثنـائي[9]، وذلك لسـهولة حـدوث التزاوج والطفرة. الجدول(1) يوضح بنية الكروموسات.

**الجدول(1): بنية الكروموسات.**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Chromosome1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| Chromosome2 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| Chromosome3 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

### 4-2-2- تابع التقييم (Evaluate Function):

يتم حساب تابع التقييم Fitness Function لكل كروموسوم من أجل اختيار الكروموسوم الأفضل، ليكون هذا الكروموسوم أب في الخطوات التالية من أجل انتاج جيل جديد(أي حلول جديدة أي عدد طلبات جديدة عشوائية). وتابع التقييم هو أساس عمل الخوارزمية يتم اختياره حسب المشكلة المدروسة، وتم تجريب العديد من التوابع للوصول الى التابع الأفضل، اعتماداً علي التجريب وعلى توابع تقييم في دراسات سابقة[3,9,10] تم استنتاج تابع التقييم الذي يعطى وفق المعادلة(1):

$$F(X) = \text{Min } (-)* B_i) \qquad (1)$$

حيث أن F(X): تابع التقييم من أجل كل كروموسوم، و الدليل i رقم الكروموسوم.

(: الانتروبيا السريعة استخدمت في الدراسة [3] حيث يتم من خلالها التنبؤ بالمستخدم الذي سيكون مسيطر بعدد طلباته في الفترات اللاحقة من ارسال الطلبات، تُحسب لكل مستخدم والمستخدم الذي تكون قيمة الانتروبيا السريعة له صغيرة هذا يعني أنه المستخدم المسيطر بعدد طلباته، أي أنه يرسل عدد طلبات كبيرة، والمستخدم الذي تكون قيمة الانتروبيا السريعة له كبيرة يرسل عدد طلبات قليلة هذا مايوضحه الشكل (3)، واشارة الناقص – وُضعت من أجل تجنب النتائج السلبية بحيث تكون جميع قيم الانتروبيا موجبة.

**الجدول(2): الإنتروبيا السريعة[3].**

| Connection | Flow Count | Fast Entropy | Flow Count | Fast Entropy | Flow Count | Fast Entropy |
|---|---|---|---|---|---|---|
| $C_1$ | 2605 | 1.387 | 2367 | 1.776 | 1094 | 2.497 |
| $C_2$ | 1831 | 1.726 | 1956 | 2.047 | 751 | 2.782 |
| $C_3$ | 1786 | 1.747 | 1890 | 2.071 | 743 | 2.799 |
| $C_4$ | 1743 | 1.776 | 1865 | 2.203 | 559 | 2.984 |
| $C_5$ | 1023 | 2.814 | 299 | 4.304 | 48 | 5.377 |
| $C_6$ | 295 | 3.748 | 495 | 3.465 | 172 | 4.086 |
| $C_7$ | 204 | 4.159 | 377 | 3.946 | 81 | 5.025 |

$$-) = /  \qquad (2)$$

حيث أن N: عدد السكان (عدد المستخدمين)، : الكروموسوم وهو عبارة عن عدد الطلبات من أجل كل مستخدم.

$B_i$: الوزن الخاص بكل كروموسوم ويحسب من خلال المعادلة (3).

$$B_i = abs\ ( -\sigma) \qquad (3)$$

$\sigma$: الانحراف المعياري الذي يعبر عن مدى تشتت القيم بالنسبة للمتوسط الحسابي (الذي هو متوسط عدد الطلبات الواردة الى السحابة) يعطى بالمعادلة (4).

$$\sigma = \qquad (4)$$

$\mu$ : متوسط عدد الطلبات الكلية الواردة إلى السحابة وتحسب وفق المعادلة (5):

$$\mu = / N \qquad (5)$$

**ملاحظة:**

المتوسط الحسابي لا يكفي لتعريف مجموعة من البيانات تعريفاً دقيقاً، بل نحتاج لمعيار اضافي يوضح مدى تشتت هذه البيانات حول الوسط الحسابي، ولذلك تم ادخال مفهوم الانحراف المعياري الذي يعبر عن مدى تشتت البيانات عن المتوسط الحسابي.

**4-2-3-الاختيار (Selection):**

يتم حساب المتوسط الحسابي لعدد الطلبات الواردة وفق المعادلة(5)، و يتم مبدأياً قبول هذه القيمة واعتبارها عتبة مرجعية، ولا نستخدمها كعتبة للكشف والتخفيف من هجوم الحرمان من الخدمة الموزعة بسبب تأثر هذه القيمة بالقيم المتطرفة (المقصود في حال استخدمنا المتوسط الحسابي كعتبة للتخفيف من هجوم الحرمان من الخدمة الموزعة سيتم منع المستخدمين الذين عدد طلباتهم أكبر من المتوسط من الوصول الي الخدمة وقد لايكون أحد هؤلاء المستخدمين من المهاجمين وبالتالي ستكون نسبة الخطأ في الخوارزمية كبير، لذلك سنستخدم أيضا الانحراف المعياري حتى يكون المجال أكبر لقبول عدد أكبر من طلبات المستخدمين والتقليل من خطأ الخوارزمية) لأن المشكلة التي تصادف أنظمة الكشف والتخفيف من الهجمات هي صعوبة التمييز بين المستخدم الشرعي والمهاجم في حال أرسل كلاهما عدد كبير من الطلبات، وذلك لأن هناك بعض المستخدمين الشرعيين(ليسوا مهاجمين) يرسلون عدد كبير من الطلبات الى السحابة. لذا نقوم بحساب الانحراف المعياري للطلبات الواردة وفق المعادلة (4)، ومن ثم نقوم بحساب الوزن الخاص بكل كروموسوم وفق المعادلة(3)، و الانتروبيا السريعه تحسب وفق المعادلة(2) وتكون ذات قيمة صغيره في حال كان عدد طلبات المستخدم كبير لأن هذا المستخدم يكون المهيمن بعدد طلباته، وذات قيمة كبيره في حال عدد طلبات المستخدم صغير، نقوم بحساب تابع التقييم وفق المعادلة(1)، وذلك بجداء بين الانتروبيا السريعه والوزن، حيث ستكون قيمة التابع كبيرة من أجل أي عدد من الطلبات، لذلك نأخذ القيم الصغيره للتابع وهذا هو المقصوج ب Min ونعتبرها آباء لأجيال لاحقة، لأن القيم الصغيره هي لعدد طلبات متوسطة بين عدد الطلبات الكبيرة والصغيرة.

باختصار يتم حساب تابع التقييم لكل كروموسوم وفق المعادلة (1) واختيار الكروموسوم الأفضل الذي يمتلك أقل قيمة لتابع التقييم، يختلف اختيار الكروموسوم الأفضل حسب تابع التقييم المقترح لكل مشكلة.

هناك عدة أنماط لاختيار الكروموسوم الأفضل وأحد هذه الأنماط هي النمط Tournament Selection حيث يتم في هذه الطريقه اختيار الكروموسوم الذي يمتلك قيمة تابع تقييم أقل(حسب تابع التقييم الذي اقترحناه يكون الكروموسوم الأفضل هو الذي يمتلك قيمة تابع أقل) والكروموسوم الذي يليه تماماً ليكونو آباء للخطوة الآتية، لأننا نحتاج الى أبوين حتى يتم التزاوج والطفرة بينهما لذلك أحد الأبوين من يمتلك أقل قيمة تابع تقييم (هو أفضل وفق التابع) والأخر هو الذي يلية تماما (الذي يمتلك قيمة تابع تقييم أعلى منه قليلاً).

### 4-2-4- التزاوج (Recombination):

في هذه الخطوة يتم التزاوج بين أفضل فردين، أفضل كروموسومين تم اختيارهم وفق تابع التقييم السابق وذلك من أجل انتاج سكان جدد (عدد طلبات جديدة عشوائية) وتسمى هذه الخطوة أيضاً Crossover وتتم بعدة طرق، جميع هذه الطرق تستخدم في الخوارزمية الجينية وجميعها يعطي نتائج ولكن كل باحث يختار الطريقة الأنسب حسب مشكلته [4]:

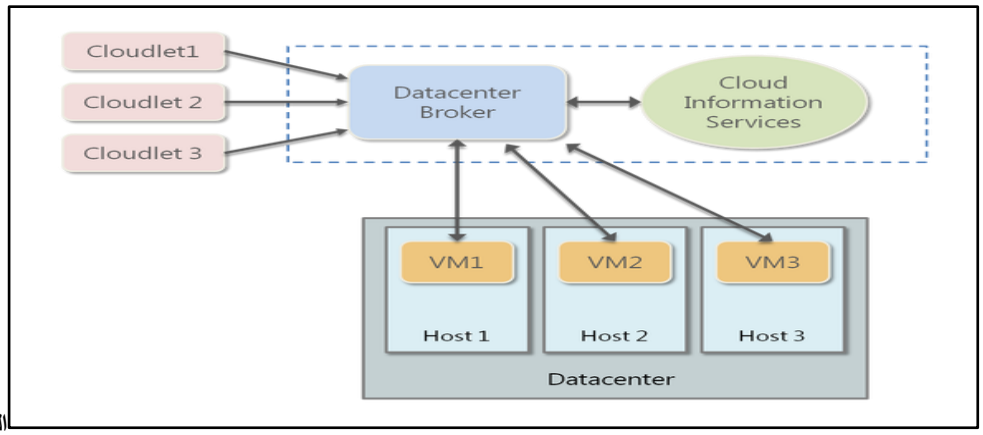
- العبور باستخدام نقطة واحدة Single point crossover: ويتم بشكل عشوائي اختيار نقطة واحده يحدث عندها التزاوج بين الأبويين، أي يتم تبادل البتات( البتات هي الأصفار والواحدات الموجودة في بنية الكروموسوم) بين الأبويين التي تقع على يمين هذه النقطة.
- العبور باستخدام نقطتين Two point crossover: يتم بشكل عشوائي اختيار نقطتين يحدث عندهما التزاوج بين الأبويين، أي يتم تبادل البتات بين الأبويين بين هاتين النقطتين.
- العبور العشوائي Uniform crossover: يتم التزاوج بين الأبويين بالاعتماد على قيمة معينة تسمى احتمال العبور crossover probability، وقد اعتمدنا هذه الطريقة، وبعد التجريب بعدة قيم لـ $P_c$ تبين أن أفضل قيمة تعطي أفضل نتيجة (أي أفضل عتبة وتخفيف أكثر من هجوم الحرمان من الخدمة الموزعة) هي القيمة $P_c = 0.7$.

### 4-2-5- الطفرة (Mutation):

يتم في هذه الخطوة احداث تغيرات طفيفه في بنية الكروموسوم والحصول على حل جديد مختلف عن الحل السابق، اعتماداً على قيمة الاحتمال $P_m$ يتم تبدلات في مواقع البتات (الأصفار والواحدات الموجودة في بنية الكروموسوم) ويتم الحصول على قيمة جديدة (عدد طلبات)، ويفضل أي تكون قيمه هذه الاحتمال منخفضة في حال كانت قيمة احتمال الطفرة $P_m$ كبير سيتحول البحث الى بحث بدائي عشوائي [4]، وبعد التجريب بعدة قيم تبين أن أفضل قيمة للطفره (أفضل قيمة تعطي أفضل عتبة) هي $P_m = 0.001$.

يتم تكرار الخطوات السابقة حتى يتم الحصول على أفضل الحلول ويكون الحل الناتج هو عباره عن العتبة الديناميكية (في كل مرة يتم ارسال طلبات من المستخدمين سيتم تكرار الخطوات السابقة حتى نحصل على أفضل عتبة والعتبة هي قيمة عدد طلبات، يتم رفض عدد الطلبات التي تتجاوز هذه القيمة، هذه العتبة تتغير عند كل ارسال، في حال العتبة الثابتة تعني استخدام قيمة واحدة ثابتة في كل ارسال يتم على أساسها التخفيف من هجوم الحرمان من الخدمة الموزعة) التي سيتم اعتمادها في الكشف والتخفيف من هجوم الحرمان من الخدمة الموزعة.

### 4-3- آلية معالجة طلبات مستخدمي السحابة [6]:




**الشكل(3): كيفية معالجة الطلبات الواردة الي السحابة [6] .**

يعبر الشكل(3) عن كيفية معالجة الطلبات الواردة الى السحابة وفق المحاكي Cloudsim وذلك وفق الخطوات [6]:

1- يرسل المستخدم الطلبات الخاصة به إلى مراكز البيانات في السحابة، يعبر عن هذه الطلبات بالمصطلح Cloudlet.

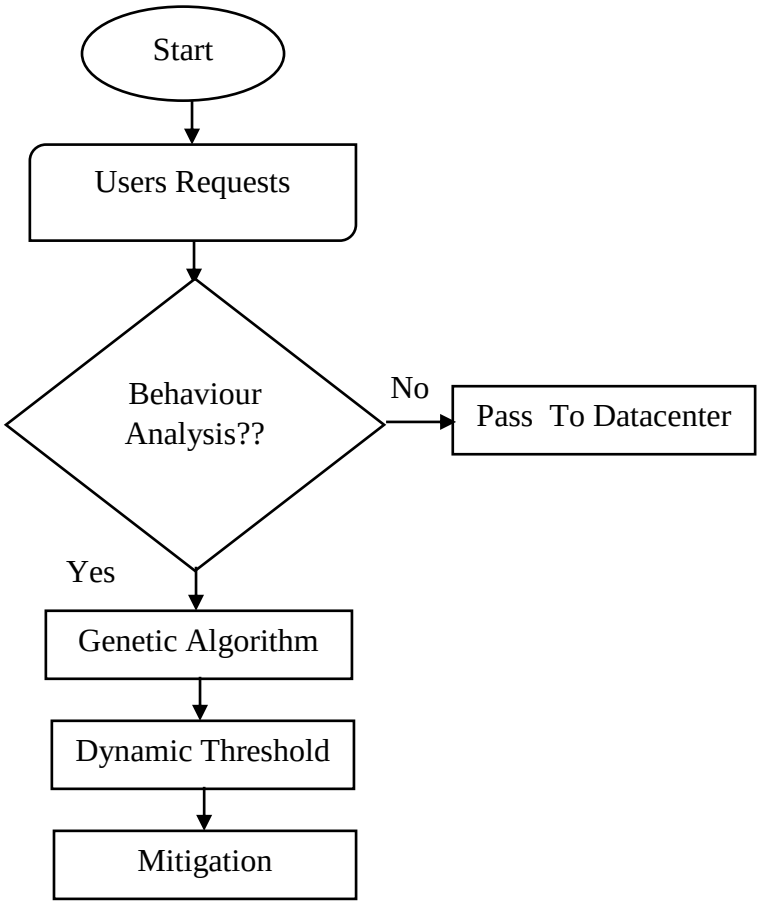
2- تمر هذه الطلبات أولا إلى بروكر مركز البيانات Datacenter Broker وهو بمثابة الحوسبة ذات الحواف Edge Computing، ويكون البروكر هو عبارة عن وسيط بين مزود خدمة السحابة والمستخدمين، يقوم البروكر بدراسة طلبات المستخدمين.

3- يتصل البروكر مع خدمات معلومات السحابة Cloud Information Services التي تحتوي معلومات عن جميع مراكز البيانات Datacenter الفعالة ومواصفات هذه المراكز من حيث (الذاكرة، المعالج، Number Of Host....).

4- يختار البروكر مركز البيانات القادر علي تلبية طلبات المستخدمين، يتم اختيار مركز البيانات الذي يحتوي قيمة الذاكرة والمعالج التي يطلبها المستخدم ، ويقوم بالاتصال معه وانشاء ألة افتراضية Virtual Machine لكل مستخدم.

**4-4- النظام المقترح :**

يوضح الشكل (4) خطوات المنهج المقترح من أجل الكشف والتخفيف من هجوم الحرمان من الخدمة الموزعة وفق الخطوات الآتية:




**الشكل(4) : خطوات المنهج المقترح.**

I. تأتي تدفقات البيانات (طلبات المستخدمين) من مستخدمي السحابة باتجاه مراكز البيانات، ستمر في البروكر Datacenter broker أولاً وهنا ستتم معالجة الطلبات في حال وجد نظام لتحليل السلوك (أي دراسة عدد طلبات المستخدم وتحديد فيما اذا كان مهاجم أو لأ اعتماداً على قيمة عتبة معينة، يعني مقارنة عدد الطلبات الواردة بقيمة عدد الطلبات المسموح بها ).

II. في حال وجد نظام للتحليل في البروكر ستتم دراسة سلوك المستخدمين من حيث عدد الطلبات لكل مستخدم من خلال تطبيق الخوارزمية الجينية بالخطوات التي ذكرت سابقاً، من أجل ايجاد عتبة ديناميكية (تتغير هذه العتبة عند كل ارسال) يتم من خلالها التخفيف من الطلبات التي سيتم تنفيذها في مراكز البيانات وستكون بداية للتخفيف من الهجوم.

III. يوضع كود النظام المقترح في البروكر(يتم وضع الكود المقترح في البروكر حتى يتم التقليل من عدد الطلبات قبل ارسالها الي مركز البيانات حتى لا يحدث تحميل زائد على مركز البيانات، ترسل طلبات المستخدمين التي تكون

عددها أقل من العتبة المسموحة إلى مركز البيانات ليتم تنفيذها، بينما طلبات المستخدمين التي يكون عددها أكبر من العتبة المسموحة تُمنع من الوصول إلى مركز البيانات ) من أجل معالجة الطلبات قبل وصولها إلى مراكز البيانات.

## 5- النتائج والمناقشة:

تم تقسيم العمل الى ثلاث مراحل:

1- **المرحلة الأولى:** تطبيق الهجوم وبيان أثره على كل من زمن الانتظار الخاص بالمستخدم الشرعي (الغير مهاجم) وزمن التنفيذ الخاص بكل ألة افتراضية VM.

2- **المرحلة الثانية:** تطبيق الخوارزمية المقترحة للتخفيف من زمن التنفيذ الخاص بكل ألة افتراضية VM، أي التخفيف من هجوم الحرمان من الخدمة الموزعة، وذلك من أجل ثلاث سيناريوهات مطبقة مختلفة من حيث عدد المستخدمين وعدد الطلبات الخاصة بكل مستخدم.

3- **المرحلة الثالثة:** مقارنة نتائج التخفيف والكشف للخوارزمية المقترحة مع نتائج تخفيف وكشف لدراسات سابقة وذلك من أجل عدة سيناريوهات مطبقة مختلفة (يتم في كل سيناريو تغير عدد المستخدمين حتى يتم تعميم النتائج على n مستخدم) كما هو موضح في الجدول (3).

**الجدول(3): جدول يوضح عدد المستخدمين في كل سيناريو.**

| السيناريو | عدد المستخدمين |
|---|---|
| السيناريو الأول | 10 |
| السيناريو الثاني | 20 |
| السيناريو الثالث | 30 |

**المرحلة الأولى:**

تحتوي السحابة على مراكز بيانات Data Centers وكل مركز بيانات له مواصفات محددة من حيث (سعة التخزين، سعة المعالجة، عدد الألات الافتراضية التي يمكن تزويدها لكل مستخدم)، فعند تعرض مركز البيانات إلى هجوم الحرمان من الخدمة الموزعة فإن جميع قدراته تكون مستهلكة من قبل المهاجمين، ولا يمكن لأي مستخدم شرعي الوصول إلى هذا المركز لتنفيذ طلباته، وبالتالي عندما يرسل المستخدم الشرعي طلب إلى هذا المركز سوف ينتظر وقت طويل جداً ولن يستطيع الوصول اليه.

بالتالي فإن أحد أهم المؤشرات الدالة على أن مركز البيانات يتعرض لهجوم الحرمان من الخدمة الموزعة هو أن زمن انتظار المستخدم الشرعي أكبر من زمن الانتظار المسموح به.

في هذه المرحلة مهاجم واحد يرسل 5 طلبات كلٌ على حده إلى مركز البيانات وكل طلب يحتوي كمية بيانات مختلفة AmountOfData وفق الشكل(5)، (كمية البيانات تدل على حجم الطلب، كلما كانت كمية البيانات كبيرة تدل على أن حجم الطلب كبير وبالتالي سيستغرق وقت أطول في تنفيذه)، و مستخدم شرعي يحاول الوصول إلى مركز البيانات هذا وفق الشكل(6)، بحساب زمن الانتظار الذي ينتظره المستخدم الشرعي من أجل كل طلب وفق المعادلة (5)، تم الحصول على الشكل(8).

```
cloudletList1 =new  ArrayList<Cloudlet>
   ();
   long cloudletLength1 = 40000;
   int pesNumber1 = 1 ;
   long cloudletFileSize1 = 100;
   long cloudletOutputSize1 = 100;
   UtilizationModelFull utilize1 = new UtilizationModelFull();
   Cloudlet clU1 = new Cloudlet(1, cloudletLength1, pesNumber1, cloudletFileSize1, cloudletOutputSize1, utilize1
   clU1.setUserId(brokerId1);
   clU1.setVmId(1);
   cloudletList1.add(clU1);
   broker1.submitCloudletList(cloudletList1);
```

**الشكل(5): طلب المستخدم المهاجم.**

```
long cloudletLength2 = 1000;
int pesNumber2 = 1 ;
long cloudletFileSize2 = 100;
long cloudletOutputSize2 = 100;
UtilizationModelFull utilize2 = new UtilizationModelFull();
    Cloudlet clU2 = new Cloudlet(2, cloudletLength2, pesNumber2, cloudletFileSize2, cloudletOutputSize2, utilize2
    clU2.setSubmissionTime(0.1);
    clU2.setUserId(brokerId1);
    clU2.setVmId(1);
    cloudletList1.add(clU2);
    broker1.submitCloudletList(cloudletList1);
```

**الشكل(6): طلب المستخدم الشرعي.**

من الشكلين (5) و(6) : Cloudlet Length يعبر عن حجم كل طلب وذلك ضمن المحاكي CloudSim.

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Exec Time       Start Time    Finish Time
    1        SUCCESS        2             1          400             400.1             800.1
    2        SUCCESS        2             1          10              800.1             810.1
BUILD SUCCESSFUL (total time: 0 seconds)
```

**الشكل(7): تنفيذ الطلبات الواردة من المستخدمين.**

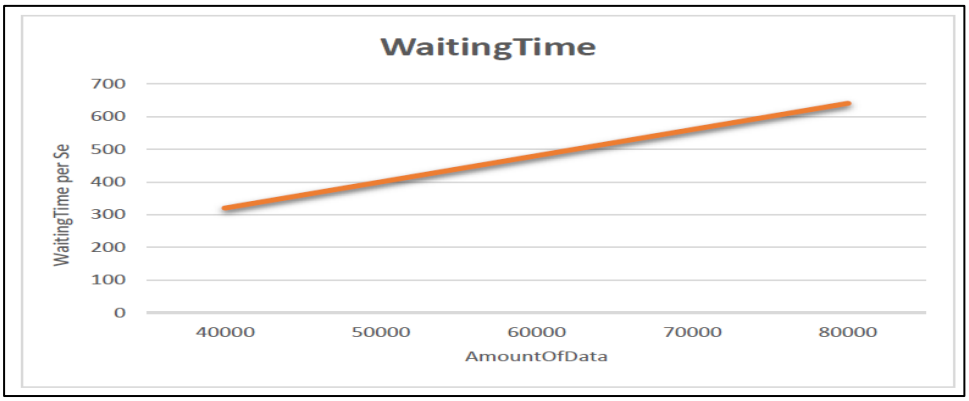
نلاحظ من الشكل(7): أن المستخدم الشرعي انتظر زمن قدره 800s حتى بدأ تنفيذه، حيث استغرق تنفيذه 10s، بينما المهاجم استغرق زمن تنفيذ 400 ثانية وكل زاد حجم الطلب الذي يرسله زاد زمن تنفيذه وبالمقابل زاد زمن انتظار المستخدم الشرعي.

يعطى زمن الانتظار لكل مستخدم شرعي وفق المعادلة (6):

$$WT = ST - ET \qquad (6)$$

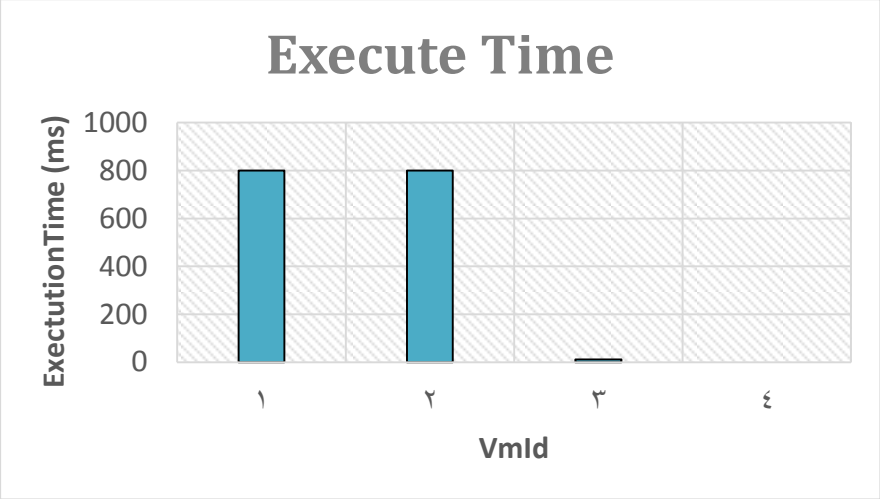
حيث أن WT: Waiting Time زمن الانتظار الخاص بالمستخدم و ST: Submission Time زمن الارسال الخاص بالمستخدم.

ET: Execution Time زمن المعالجة الخاص بالمستخدم.




**الشكل(8): زمن الانتظار الخاص بمستخدم شرعي أثناء تطبيق الهجوم.**

تبين لنا من الشكل(8) أن زمن الانتظار الخاص بالمستخدم الشرعي يزداد كلما ازدادت كمية البيانات التي يرسلها المهاجم، ويزداد أيضاً كلما ازداد عدد الطلبات التي يرسلها المهاجم، لأن زيادة عدد الطلبات مثل زيادة حجم الطلب تزيد من زمن التنفيذ وبالتالي تزيد من زمن انتظار المستخدم الشرعي، وبشكل تدريجي سيُحرم المستخدم من الوصول الى الهدف وذلك من خلال استمرار المهاجم بارسال كميات كبيرة من البيانات( سواء كبيره بحجم الطلب أو عدد الطلبات). هذا المخطط من أجل مهاجم واحد فقط، فماذا سيحدث اذا كان هناك ملايين من المهاجمين والهدف واحد؟ بالتأكيد سيسقط الهدف المطلوب ويخرج عن الخدمة.




**الشكل(9): أثر الهجوم على زمن التنفيذ الخاص بكل ألة افتراضية VM .**

تبين لنا من الشكلين (7)و(9): زمن التنفيذ الخاص بكل ألة افتراضية VM يتأثر بالهجوم تأثير كبير ويمكن كشف الهجوم من خلاله (زمن التنفيذ يتناسب طرداً مع عدد الطلبات وحجم كل طلب)، حيث يكون زمن التنفيذ الذي تستغرقه الألة الافتراضية VM التي تتعرض للهجوم أكبر بكثير من زمن التنفيذ للألة الافتراضية الأخرى VM الأخرى، ويزداد هذا الزمن طبعاً بازدياد عدد طلبات المهاجم. وبالتالي التقليل من هذا الزمن هو تخفيف من الازدحام والضغط الذي تتعرض له الألة الافتراضية VM وبالتالي هو تخفيف من هجوم الحرمان من الخدمة الموزعة.

**المرحلة الثانية:**

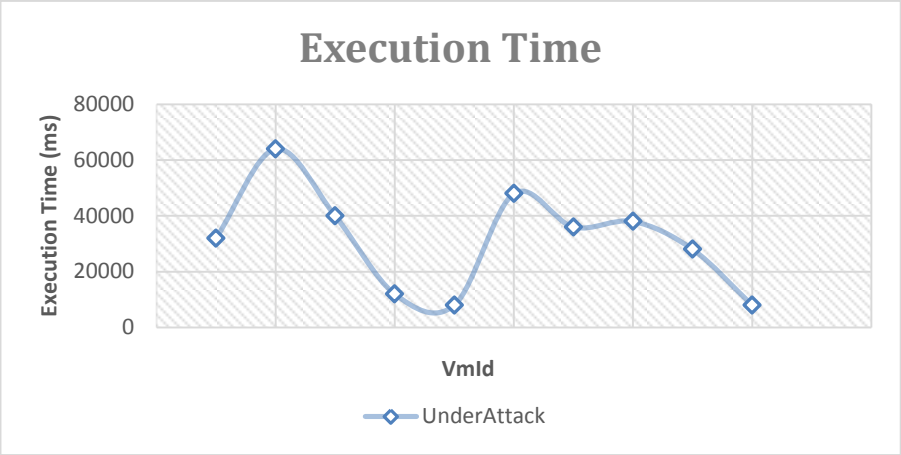
تطبيق النظام المقترح من أجل التقليل من زمن التنفيذ الخاص بكل ألة افتراضيةVM، أي التخفيف من هجوم الحرمان من الخدمة الموزعة، وذلك من أجل السيناريو الأول الذي يتألف من 10 مستخدمين كل مستخدم يرسل عدد محدد من الطلبات، وينقسم هؤلاء المستخدمين بين مهاجمين ومستخدمين شرعيين، يوضح الشكل(10) جزء من الكود المستخدم.

```
//cloudlet user5
        cloudletlist5 = new ArrayList<Cloudlet>
();
long cloudletLength5 = 80000;
int pesNumber5 = 1 ;
long cloudletFileSize5 = 100;
long cloudletOutputSize5 = 100;
UtilizationModelFull utilize5 = new UtilizationModelFull();
for (int cloudletId5 = 0; cloudletId5 < 10; cloudletId5++){
Cloudlet cl5 = new Cloudlet(cloudletId5, cloudletLength5, pesNumber5, cloudletFileSize5
cl5.setUserId(brokerId5);
cl5.setVmId(5);
cloudletlist5.add(cl5);
}
broker5.submitCloudletList(cloudletlist5);

//parameter for user6 attacker
cloudletlist6 = new ArrayList<Cloudlet>
    ();
    long cloudletLength6 = 80000;
    int pesNumber6 = 1 ;
    long cloudletFileSize6 = 100;
    long cloudletOutputSize6 = 100;
    UtilizationModelFull utilize6 = new UtilizationModelFull();
    for (int cloudletId6 = 0; cloudletId6 < 60; cloudletId6++){
    Cloudlet cl6 = new Cloudlet(cloudletId6, cloudletLength6, pesNumber6, cloudletFileSize
    cl6.setUserId(brokerId6);
    cl6.setVmId(6);
    cloudletlist6.add(cl6);
    }
```

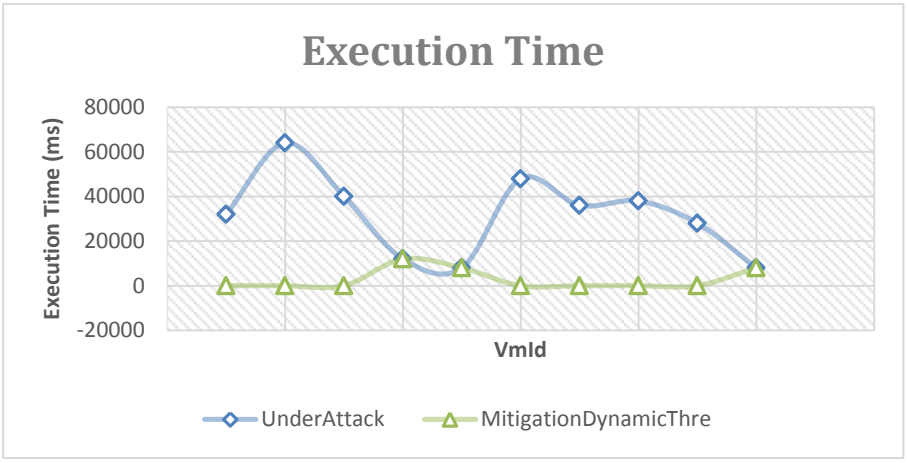
**الشكل(10): طلبات المستخدمين 5 و6 .**

تبين لنا من الشكل (11) أن: زمن التنفيذ الخاص بكل ألة افتراضية وذلك من أجل 10 مستخدمين قبل تطبيق الخوارزمية المقترحة.




**الشكل(11): زمن التنفيذ لكل ألة افتراضية Vm من أجل السيناريو الأول قبل استخدام العتبة.**

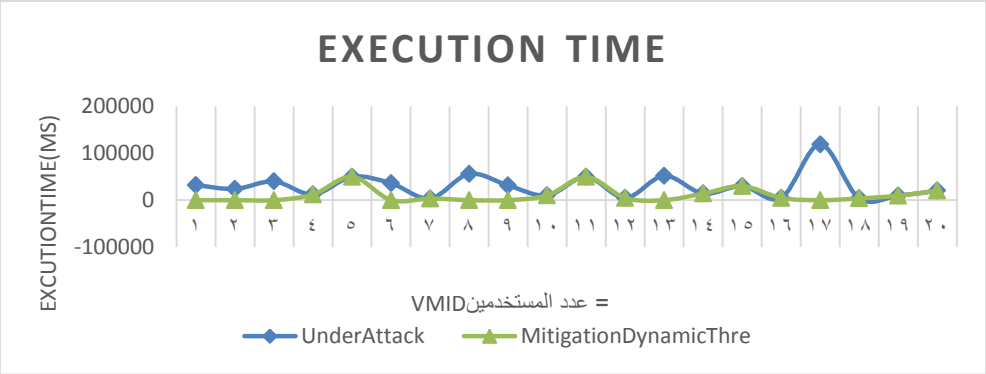
تصل طلبات المستخدمين إلى البروكر وعندها يتم تطبيق الخوارزمية الجينية على عدد طلبات المستخدمين القادمين، وتكون نتيجة هذه الخوارزمية عتبة يتم على أساسها التخفيف، تتم مقارنة عدد طلبات كل مستخدم بهذه العتبة، المستخدم الذي تتجاوز عدد طلباته هذه العتبة يتم منعه من الوصول إلى مركز البيانات، أي منعه من تنفيذ طلباته، وبالتالي زمن التنفيذ الخاص بهذا المستخدم سيصبح صفر(لأنه لم يصل مركز البيانات أصلا) مقارنةً بزمن تنفيذه قبل استخدام الخوارزمية.




**الشكل(12): زمن التنفيذ لكل ألة افتراضية Vm من أجل السيناريو الأول بعد استخدام العتبة.**

تبين لنا من الشكل(12) الذي يعبر عن مقارنة بين زمن التنفيذ الخاص بكل ألة افتراضية VM أثناء تطبيق الهجوم، وزمن التنفيذ الخاص بكل ألة افتراضية VM بعد تطبيق الخوارزمية المقترحة، وبالتالي نلاحظ أن الخوارزمية المقترحة تقلل من زمن التنفيذ لكل ألة افتراضية كانت عدد الطلبات التي تنفذها أكبر من العتبة الناتجة عن تنفيذ الخوارزمية وبالتالي تخفف من هجوم الحرمان من الخدمة الموزعة وذلك من أجل السيناريو الأول أي من أجل 10 مستخدمين فقط (كل مستخدم يخصص له ألة افتراضية وبالتالي VMID هي نفسها عدد المستخدمين).

في حال زيادة عدد المستخدمين الى 20 (كل مستخدم يخصص له ألة افتراضية وبالتالي VMID هي نفسها عدد المستخدمين) مستخدم وتكرار خطوات السيناريو الأول نحصل على الشكل(13).

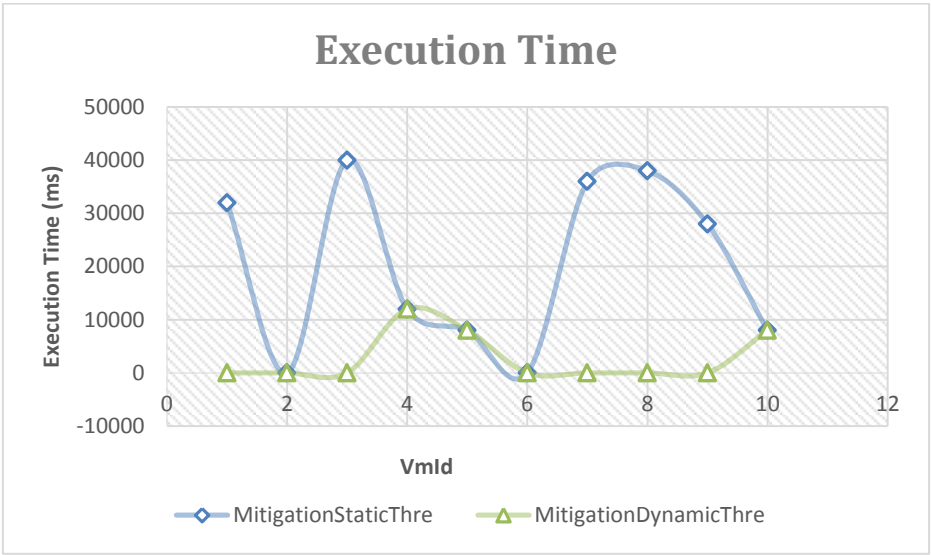



**الشكل(13):زمن التنفيذ لكل ألة افتراضية Vm من أجل السيناريو الثاني في حال استخدام الخوارزمية.**

تبين لنا من الشكل(13) الذي يعبر عن: مقارنة بين زمن التنفيذ الخاص بكل ألة افتراضية VM أثناء تطبيق الهجوم وزمن التنفيذ الخاص بكل VM عند تطبيق الخوارزمية المقترحة، وبالتالي نلاحظ أن الخوارزمية المقترحة تقلل من زمن التنفيذ وبالتالي تخفف من الهجوم وذلك من أجل السيناريو الثاني، وفي حال زيادة عدد المستخدمين الى n مستخدم سنحصل على نفس النتيجة، أن الخوارزمية تخفف من هجوم الحرمان من الخدمة الموزعة.

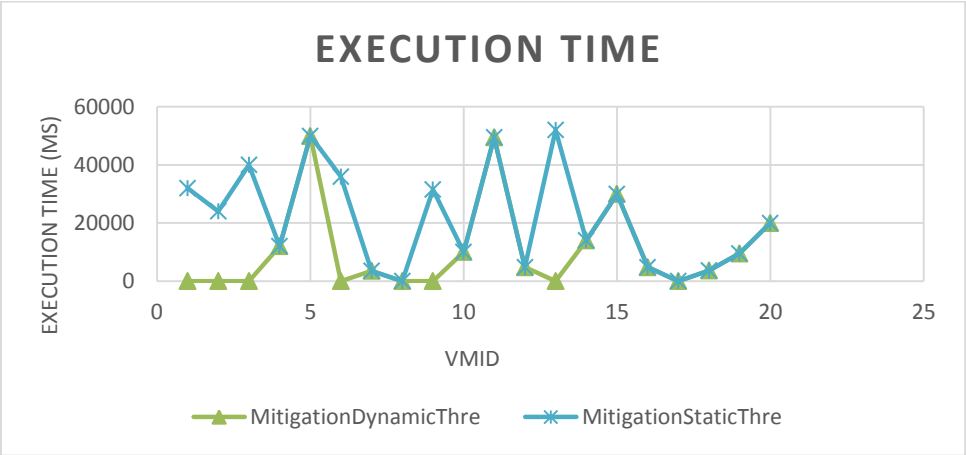
**المرحلة الثالثة:**

مقارنة النتائج التي توصلنا اليها مع نتائج الدراسة [10]، والتي تعتمد العتبة الثابتة (أي استخدام عتبة واحدة فقط عند كل ارسال إلى السحابة، أي هذه العتبة ثابتة في كل مرة يتم ارسال الطلبات إلى السحابة) في التقليل من زمن التنفيذ وفي الكشف عن هجوم الحرمان من الخدمة الموزعة.




**الشكل(14):زمن التنفيذ لكل ألة افتراضية Vm من أجل السيناريو الأول.**

أظهر لنا الشكل(14) الذي يعبر عن مقارنة بين زمن التنفيذ عند استخدام الخوارزمية المقترحة وزمن التنفيذ عند استخدام العتبة الثابتة في الدراسة [10] وذلك من أجل السيناريو الأول، ونلاحظ أن نظامنا أكثر فعالية في التقليل من زمن التنفيذ وبالتالي التخفيف من الهجوم.




**الشكل(15): زمن التنفيذ لكل ألة افتراضية virtual machine من أجل السيناريو الثاني.**

تبين لنا من الشكل(15) الذي يعبر عن مقارنة بين زمن التنفيذ عند استخدام الخوارزمية المقترحة وزمن التنفيذ عند استخدام العتبة الثابتة في الدراسة [10] وذلك من أجل السيناريو االثاني، ونلاحظ أن نظامنا أكثر فعالية في التقليل من زمن التنفيذ وبالتالي التخفيف من الهجوم.


**الشكل(16): زمن التنفيذ لكل virtual machine من أجل السيناريو الثالث عدد المستخدمين 30.**

تبين لنا من الشكل(16) الذي يعبر عن مقارنة بين زمن التنفيذ عند استخدام الخوارزمية المقترحة وزمن التنفيذ عند استخدام العتبة الثابتة في الدراسة [10] وذلك من أجل السيناريو االثالث، ونلاحظ من أجل جميع السيناريوهات

الأعلى(يعني بزيادة عدد المستخدمين إلى n مستخدم) سنحصل على نفس النتائج السابقة، التي تؤكد فعالية النظام المقترح في التخفيف من هجوم الحرمان من الخدمة الموزعة.

تحسب دقة الكشف $D_R$ وفق المعادلة[7].

$$D_R = TP / (TP+TN) \qquad (7)$$

حيث أن: TP عدد المستخدمين المهاجمين الذين تم كشفهم بشكل صحيح، TN: عدد المستخدمين المهاجمين

الذين لم يتم كشفهم.

خلال تقييم النظام المقترح من حيث دقة الكشف، لوحظ أن النظام المقترح يُحسن من دقة الكشف مقارنة مع الدراسة

[10] وهذا مايوضحه الجدول (4).

**الجدول(4): تقييم النظام المقترح من حيث دقة الكشف.**

| السيناريو | دقة الكشف في حال العتبة الثابتة | دقة الكشف في استخدام الخوارزمية المقترحة |
|---|---|---|
| السيناريو الأول | 73% | 85% |
| السيناريو الثاني | 80% | 88% |
| السيناريو الثالث | 74% | 100% |

## 6- الاستنتاجات والتوصيات المستقبلية:

تكمن أهمية البحث بكونه تطرق على مشكلة حتى الأن لم يتم ايجاد حل لها، ومن خلال الدراسة العملية السابقة نستنتج:

1- المرحلة الأولى من العمل أظهرت تأثير هجوم الحرمان من الخدمة الموزعة على زمن انتظار المستخدميين الشرعيين، حيث تبين لنا أن زمن الانتظار وزمن التنفيذ يناسبا طرداً مع عدد وكمية الطلبات الواردة، وهذا بدورة يؤدي الي تراجع الخدمة وخسارة العملاء.

2- المرحلة الثانية أظهرت مدى فعالية النظام المقترح في التخفيف من الهجوم مقارنةً مع الدراسة المرجعية [10].

3- المرحلة الثالثة أوضحت أن النظام المقترح أفضل من النظام في الدراسة [10]، الذي يعتمد العتبة الثابتة في الكشف والتخفيف حيث أن دقة الكشف في السيناريو الأول 85% وفي السيناريو الثاني 80% وفي السيناريو الثالث 74% وهذا يدل على أن النظام المقترح أفضل من الدراسة المرجعية.

يفيد نظامنا المقترح في كشف الأنماط ذات معدل الطلبات العالي والتخفيف منها، وبالتالي نوصي الباحثين بتطوير هذا النظام بحيث يكون قادراً على الكشف و التخفيف من هجمات الحرمان من الخدمة الموزعة المتخفيه. هذا الهجوم الذي يبدأ بمعدل طلبات قليل وضمن الحدود الطبيعية ويسلك سلوك المستخدم الشرعي[4].

## 1- المراجع:

[1] Srinivasan, K., Mubarakali, A., Alqahtani, A. S., & Kumar, A. D. (2017, February). *A Survey on the Impact of DDoS Attacks in Cloud Computing: Prevention*, Detection and Mitigation Techniques. In Intelligent Communication Technologies and Virtual Mobile Networks (pp. 252-270). Springer, Cham.

[2] Balobaid, A., Alawad, W., & Aljasim, H. (2011, December). *A study on the impacts of DoS and DDoS attacks on cloud and mitigation techniques*. In 2011 International Conference on Computing, Analytics and Security Trends (CAST) (pp. 411-421). IEEE

[3] David, J., & Thomas, C. (2015). DDoS attack detection using fast entropy approach on flow-based network traffic. *Procedia Computer Science*, *50*(4), 30-36..

[4] Goutham, D. V., & Tejaswini, M. (2011). *A Denial of Service Strategy To Orchestrate Stealthy Attack Patterns In C loud Computing*. International Journal of Computer Engineering and Technology, 7(3).

[5]Kumar, G. (2014). *Evaluation metrics for intrusion detection systems-A study. Evaluation*, *2*(8), 8-7.

[6] http://udemy.com/course/learn-basics-of-cloudsim.

[7] Ali, L., Mathieu, H., & Biennier, F. (2006, April). *Monitoring and Managing a Distributed Networks using Mobile Agents*. In 2006 2nd International Conference on Information & Communication Technologies (Vol. 2, pp. 3377-3382). IEEE.

[8] Bahaweres, R. B., & Alaydrus, J. S. M. (2011, April). *Building a private cloud computing and the analysis against DoS (denial of service) attacks*: Case study at SMKN 6 Jakarta. In 2011 4th International Conference on Cyber and IT Service Management (pp. 1-6). IEEE..

[9] Mizukoshi, M., & Munetomo, M. (2010, May). *Distributed denial of services attack protection system with genetic algorithms on Hadoop cluster computing framework. In 2010 IEEE Congress on Evolutionary Computation (CEC) (pp. 1075-1080)*. IEEE.

[10] Bakshi, A., & Dujodwala, Y. B. (2014, February). *Securing cloud from ddos attacks using intrusion detection system in virtual machine*. In 2014 Second International Conference on Communication Software and Networks (pp. 260-264). IEEE.

[11] Mathieu, H., Ali, L., & Biennier, F. (2006, May). *A distributed management system: In 12th IFAC Symposium on Information Control Problems in Manufacturing* (pp. 657-665).