Int. J. Advance. Soft Comput. Appl., Vol. 6, No. 1, 2014
ISSN 2074-8523; Copyright ©SCRG Publication, 2014

Virtualization Technique for Port Knocking in Mobile Cloud Computing

Laleh Boroumand[1], Muhammad Shiraz[1], Abdullah Gani[1], Rashid Khokhar[2]

[1] Mobile Cloud Computing Research Lab, Faculty of Computer Science and Information Technology, University of Malaya

e-mail: Lalehborouamnd@siswa.um.edu.my
muh_ shiraz@ siswa.um.edu.my
abdullah@um.edu.my

[2]School of Computing and Mathematics, Charles Sturt University,

e-mail: rkhokhar@csu.edu.au

Abstract

The rise in adoption of mobile cloud computing (MCC) has increased the security concerns and this fact has motivated security practitioners to consider novel lightweight authentication approaches in order to improve the authentication as a service (AaaS). Indeed, the authentication process has a direct effect on client's perception of trust and therefore the authentication method is required to be robust and easy to use simultaneously. In this paper, we have proposed virtual port knocking authentication method which uses port knocking to authenticate users by using the closed port. Moreover, it employs the virtualization concept taken into account as a constructive concept in MCC to enhance the performance which is affected by long length of knock sequence in the basic port knocking. The statistical results indicate that the proposed method offers an acceptable level of security calculated based on the probability of hacking. The empirical results indicate that the imposed buffering load to the cloud gateway is decreased around 4% for the short length knock sequence while, the enhancement is about 12% in case of using the long knock sequence.

Keywords: *Authentication, portknocking, Virtualization, Mobile Cloud computing*

1 Introduction

Nowadays, most of the daily operations which contain personal information are done through online applications. These applications make the operations more facile during the short period of time. However, at this moment in time, security has a high priority due to irreversible losses which are occurred by disclosing the personal information especially in the financial operations. The security concerns would be more highlighted in the networks which offer services such as storage, computational operations and platforms sharing in the server side to cover the restrictions of the user's devices. One of these networks is mobile cloud computing (MCC) [1] which is adopted by organizations and individuals in last few years. With growing the number of users, security turns to a considerable challenge for cloud provider [2], [3].

To overcome the security problems and reduce the negative effect on them, authentication process counts as the first important applied stage [4]. Although user name and password is the most common way to authenticate the users, there are lots of attempt to make these kind of authentication more complicate to reach the higher security. One of the approaches to this end is an interactive authentication which means using the second password (usually is a sent code to the phone number of user) during the performing operations that utilizes by the most of the online bank systems. These kind of authentication is known as two-factors authentication method [5],[6],[7]. Nonetheless, two-factors authentication method suffer from communication limitation in some cases for instance delayed received SMS or weak mobile network coverage in some areas can bring on the timeout error and terminate the session middle of the process. Therefore, the lightweight authentication method which is a possible way to make balance between security and performance is still required.

Recent studies try to apply port knocking authentication model for the MCC which show that it has potential to use as an antidote of fears about security issues in authentication stage [8]. In port knocking method, authenticating process is performed on closed ports. Indeed, clients (port-knocker) send non-reply synchronization (SYN) packets to the specific closed ports of server's FW [9],[10] consecutively that is called knocked process. Meanwhile, the server logs the incoming packets through storing information in its buffers. Therefore, port sequence is used as a password. In fact, this method uses secret key for authentication that is a sequence of ports and this secret key is defined statically or dynamically. Whenever the valid sequence is determined, an appropriate port is opened for port-knocker (who generates the sequence) to use require the services [11]. In another word, port-knocking is a strategy used to grant remote access without requiring left the port open constantly. Hence, port in this system is kept open based on user's demand for a period of time. Consequently, the vulnerability of the system is reduced in comparison

to the situation in which ports always remain unclosed for authorized users to offer services [11]. Also the basic port knocking model needs some changes to be suitable for MCC which are considered hereinafter.

The basic port knocking method has undergone several changes by previous studies to overcome issues which are Plain text port sequence, Network Address Translate (NAT) Knock [12], Denial of Service (DoS) knock attacks [13], Appropriate length of knock sequence [14], Out of order packet delivery [15] and Lack of association between authentication and connection [15], [16]. During this transformation the port sequence has been changed from static to dynamic mode [17] and plain knock sequence was converted to the encrypted sequence [18], [19], [20], as it makes port knocking more secure. Lack of attention to the length of the knock sequence needs to be addressed by port knocking authentication to make it suitable for MCC [[21], [22] and [11]]. Indeed, length is a factor that has an effect on the performance if it is not chosen properly. Theoretically, if the selected length is too short, sniffers can find the range easily with capturing the traffic. On the other hand, the long length increases the possibility of DoS knocking attack. Providing balance between the length of knock sequence and prevention of attack is an issue that is expected to be solved in an improved version of port-knocking [14].

In this paper, the impact of selecting inappropriate length with the implementation is analyzed. Moreover, as a main contribution, we propose a new technique to solve these issues which is known as virtual port knocking technique. Moreover, we direct the researchers who are interested to solve this problem through suggesting the applicable solution. The rest of this paper is organized as follows; Section 2 presents the analysis of length issues in knock sequence in terms of security and performance, followed by virtual port knocking authentication in Section 3. Section 4 discusses about the security and performance results which are gathered from implementation of virtual port knocking and compare them with basic port knocking's result. Finally, Section 5 concludes the paper with some future directions to overcome these issues which can make the port knocking suitable for MCC environment.

2 Analyses the Issue of knock sequence length

One of the constraints of port knocking is the selection of appropriate length for knock sequence due to its effect on both security and performance. Befit length of knock sequence can be weighed by a level of security and performance that cloud users are expected. In overall, short length for knock sequence refers to a sequence which has more than one but less than four knock packets. Selecting the short knock sequence which is determined statically increases the probability of identification of port sequence by malicious users who are monitoring the traffic. For example, consider a scenario in which each user

needs to send three packets to predefined ports. Although these three ports are closed during the knocked process, hackers through monitoring and sniffing the packets can figure out the port numbers. Even if they do not know the correct order of them, they can find out, with changing the order of knock sequence less than seven times. To decrease the chance of hackers in disclosure the knock sequence, two options exist. First we can use dynamic knock sequence in which random ports are utilized for knocking process. It would be more difficult for hackers to capture the knock sequence. Second option concerns about using long knock sequence that it covers the security issue, but it leads to performance challenges.

Long knock sequence decreases the performance in two aspects, which are authentication times and buffering load. Long sequence takes more time in comparison with short length to be completed and then server compares with predefined one in static knock sequence. The time which is wasted in a dynamic long sequence is almost two times of static short sequence. Because port knocking module in the server side waits to receive completed sequence, then spend time to generate the sequence based on the random key. In the final stage, a generated sequence is compared with received one. Hence, long length of the knock sequence imposes latency to the authentication process. On the other hand, for each port which is attending to the port knocking authentication a buffer is allocated to store the information during the knocking process. The buffer is expected to be flushed after finishing the knock process regardless the result. Therefore, in case of using long knock sequence, buffer depletion occurs later than the situation in which short knock sequence is used. Moreover, size of the buffer and number of users who knock the certain ports in a same time could be led to the buffer overflow. Buffer overflow leads to miss the correct sequence and failure of authentication, although the user is in the authenticated group.

Table 1 shows the theoretical comparison between short and long lengths of knock sequences, which are evaluated in the following through using the mathematical and experimental results. Regards to table, short knock sequence has a high probability to be disclosed by the hacker in comparison with long knock sequences, while, in terms of performance short knock sequence provides fast authentication. Furthermore, the load which imposes to the server due to monitoring and allocating the buffers in short knock sequence is cost-effective in contrast with the long one. Moreover, short knock sequence is preferable for high-traffic networks due to a higher chance that it has to be completed. Generally, short knock sequence is an acceptable option in scenarios that concern about the performance while the long knock sequence is expected to choose by high-security scenarios.

Table 1: Comparison of short and long knock sequence

	Identification probability by hackers	Time of authentication	Load of monitoring	High traffic network suitability
Short knock sequence	High	Low	Low	Yes
Long knock sequence	Low	Low	High	No

2.1 Impact of the knock sequence length on the security

Security issue which may arise due to select the short knock sequence is evaluated here through using the mathematical concept. In the rest of this paper, X_{HP} and X_{TP} parameters indicate the number of hacked packets and number of transferring packets, respectively. Mathematically, the number of ways to have X_{HP} among X_{TP} are calculated through using Equation (1) in which "NDWHP" indicates the number of different ways to hack a packet.

$$NDWHP = \binom{X_{TP}}{X_{HP}} = \frac{X_{TP}!}{X_{HP}!(X_{TP} - X_{HP})!} \tag{1}$$

Assuming that the length of the sequence is "3" then the number of ways that one of the packet could be hacked is equal to :

$$NDWHP = \binom{3}{1} = \frac{3!}{1!2!} = 3 \tag{2}$$

Nevertheless, in case of port knocking authentication, all the ports are required. That means all the packets are expected to hack by sniffers to have a prosperous authentication. Logically, there is one way to have a successful hack of sequence in which all packets should be sniffed. Additionally, the mathematical concepts support the noticed logical statement through using Equation (3).

$$NDWHP = \binom{3}{3} = \frac{3!}{3!} = 1 \tag{3}$$

Each packet has an equal chance of being hacked or not, so all eight possible outcome which is illustrated by following graph (Fig 1) are equally probable. The probability of a different number of hacked packets calculates through using equation (4) in which L_{KS} indicates the length of knock sequence, and

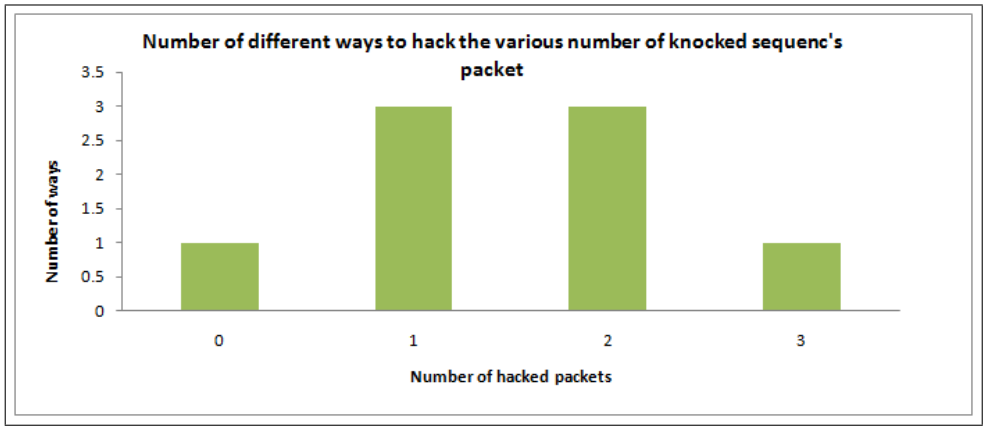

Figure 1: Number of ways which led to different number of hacked packets in 3-knocked sequence

P(SP) shows the probability of success sniffing for each packet.

$$P(SP) = \frac{NDWHP}{2^{L_{KS}}} \tag{4}$$

Table 2 shows the probability of a different number of hacked packets in the three knocked sequence primary column determines the number of hacked packets while the second column belongs to a number of ways that the requirement of the first column could be occurring. Finally, the third column shows the probability. In the port knocking that all packets should be hacked

Table 2: Probability of hacked packets in the 3 knocked sequence

Number of hacked packets in the 3 knocked sequence	Number of ways	Probability
0	1	1/8
1	3	3/8
2	3	3/8
4	1	1/8

by a hacker to allow them to pass the authentication process the number of ways is always "1" then the probability of hacked a knock sequence is calculated through using following Equation (5) in which P (HS) indicates the

probability of hacking a sequence.

$$P(HS) = \frac{1}{2^{L_{KS}}} \tag{5}$$

Therefore, Table 3 shows the probability of hacked sequence for various lengths which includes short and long knocked sequence.

Table 3: Probability of hacked sequence for various lengths

Length of knock sequence	Probability
3	$1/8 = 0.125$
4	$1/16 = 0.062$
5	$1/32 = 0.031$
7	$1/128 = 0.007$
10	$1/1024 = 0.0009$

Fig 2 shows the hacking probability of a knock sequence. The horizontal axis indicates different lengths while the vertical axis depicts the probability of hacking. The down ward trend can figure out from the graph meanwhile the length of sequence is increasing. Although the graph has downward slope, gradient descent among short lengths is sharper than the long ones.

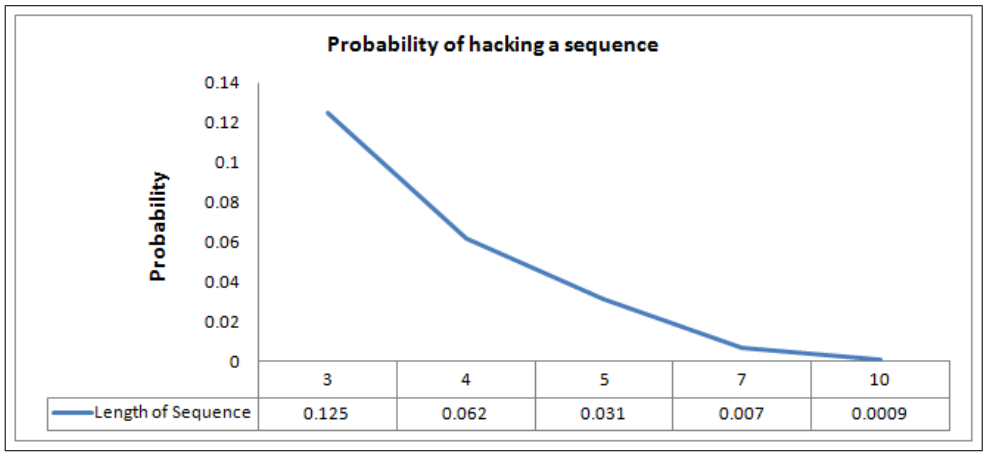

Figure 2: Probability of hacking a sequence for different lengths

Still the achievement is not enough for malicious users to claim that they can misuse the sequence, because the order of the ports plays a key rule in the port knocking authentication. Therefore, the hacker needs try different combinations, $L_{KS}!$ times to find a correct order. Accordingly, the exact

probability of hacked Sequence P (HS) could come out from Equation (6) for one user and single attempt to knock the server in the certain time.

$$P(HS) = \frac{1}{2^{L_{KS}} \times L_{KS}!} \tag{6}$$

Now if considering to a real system in which multiple users try to send an authentication request to the server the probability is calculated by Equation (7) in which "U" shows the number of users who are in the authentication phase in a specific time.

$$P(HS) = \frac{U}{2^{L_{KS}} \times L_{KS}!} \tag{7}$$

It is necessary to point that the note this equation is true in case of static sequence.
Fig 3 illustrates the hacking probability of a sequence given 100 parallel authentications. The vertical axis represents the hacking probability, and the horizontal axis depicts the knock sequence's length. It can be noted from the figure that the hacking probability exponentially decreases by the increase of the knock sequence's length.

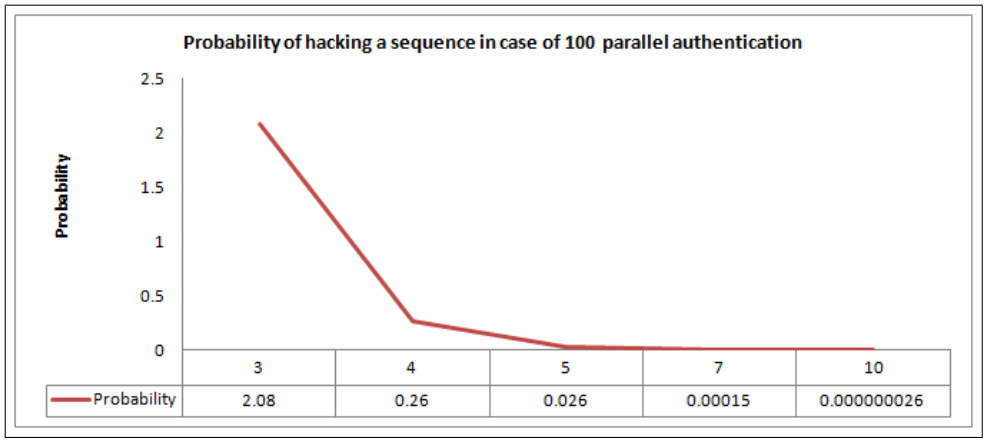

Figure 3: Probability of hacking for various length of sequence in case of 100 simultaneous authentication request

Fig 4 presents the hacking probability of a 5-knocked sequence. Accordingly, the vertical axis depicts the hacking probability, and also the horizontal axis represents the number of users. As it is shown in the figure, the probability of hacking increases gradually until having 1000 users and after that point, the hacking probability starts to rise moderately by adding the number of users.

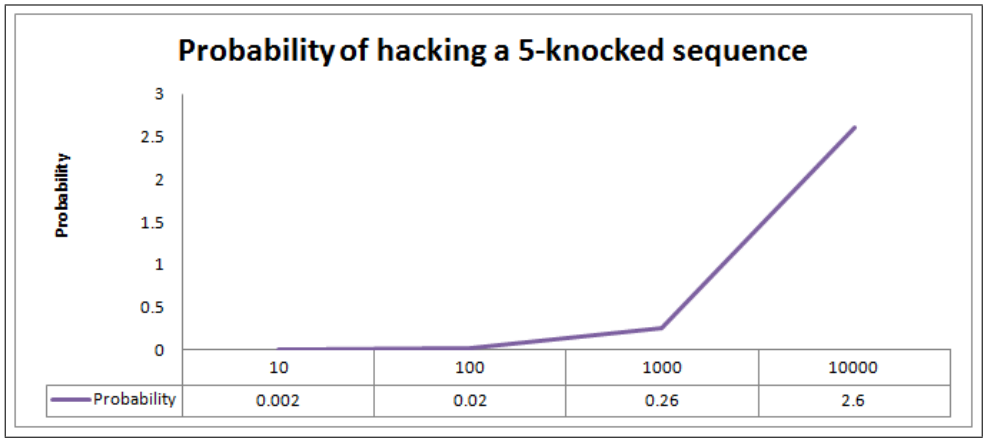

Figure 4: Probability of hacking for 5-knocked sequence with different number of users

2.2 Analyses the impact of length of the knock sequence on performance

Performance of port knocking authentication evaluates through using two following parameters;
1. Time of port knocking authentication (ToPKA): which start counting time when a port knocker sends the authentication request from to end of the authentication process.
2. Load of buffering (LoB): This parameter computes the load which is imposed to the CGW due to monitoring the ports and creating dynamic address lists to buffer the packets of a port knocker. Indeed, the LoB estimates by the memory space which is occupied for buffering. Hereinafter, two mentioned parameters are measured to examine the effect of the length on the performance.

2.2.1 Measuring the time of basic port knocking authentication

One of the factors which are involving evaluating the performance is time of authentication. The theoretical discussion asserted that using long knock sequence consume more time in comparison with the short length. Table 4 shows the run time for single client through using different length of knock sequence. Table 4 clearly shows that increasing the length of knocked sequence the time

Table 4: Run time of basic port knocking for single attempt

L_{ks}	3	4	5	7	10
ToPKA	3.034	4.079	5.030	7.040	10.036

of authentication take more seconds for performing authentication. Moreover, it would be worse in case of parallel authentication. The gathered data from 100 concurrent 5-length port knocking authentications, show the minimum run time is about "5020.89" milliseconds while the maximum run time is "5222.8" milliseconds. Time increases to "5079.26" and "18359.07" milliseconds in case of having 500 simultaneous authentication. That determines growing the number of users has a negative effect in run time of the basic port knocking.

2.2.2 Measuring the load of buffering in basic port knocking

In the port knocking each port needs a list to track the port knocker's operations. The order of sequence would be checked by port knocking module in the server through storing IP address of the port knocker in the list of each port. Hereinafter, the load of buffering for each port knocking is investigated. The LoB can evaluate by memory space that is occupied for each authentication process. First, the LoB of 3-knocked sequence examines then it expands for other lengths of knock sequence. Dynamic address lists which create to keep the track of port sequences in the 3-knocked port knocking are known as dynamic lists because they are created during the knock process, and they are elevating the port knocking. Each list stores the IP address of the port knocker. Then, for individual user and single attempt of sending an authentication request, each buffer needs 32 bits. Hence, the required space for each port knocking authentication computed by the equation (8).

$$LoB = L \times 32bits \tag{8}$$

now if the number of users who send a request simultaneously equal to "U" the LoB calculates through using Equation (9).

$$LoB = L \times U \times 32bits \tag{9}$$

Above two equations clearly indicate that the increase in LoB has a direct relationship with the increase the length of knock sequence and number of users. Table 5 shows the LoB for 100 simultaneous authentication requests and various lengths. The LoB values are in kilobytes. Graphs in Fig 5 reveal

Table 5: Load of buffering of 3-knocked basic port knocking for 100 attempts

L_{ks}	3	4	5	7	10
LoB	1.1718	1.5625	1.9531	2.7343	3.9062

the memory space which is occupied by each length of knock sequence per number of request for authentication. The horizontal axis belongs to a different

number of users who are in the authentication stage in the same time while, the vertical one shows the occupied memory space in Kilobytes. The upward trend of all graphs determines that increment of the number of request lead to more required memory. Moreover, the important point can figure out from the graphs is stunning difference between occupied memory in short and long knocked sequence, especially when the amount of users increases dramatically.

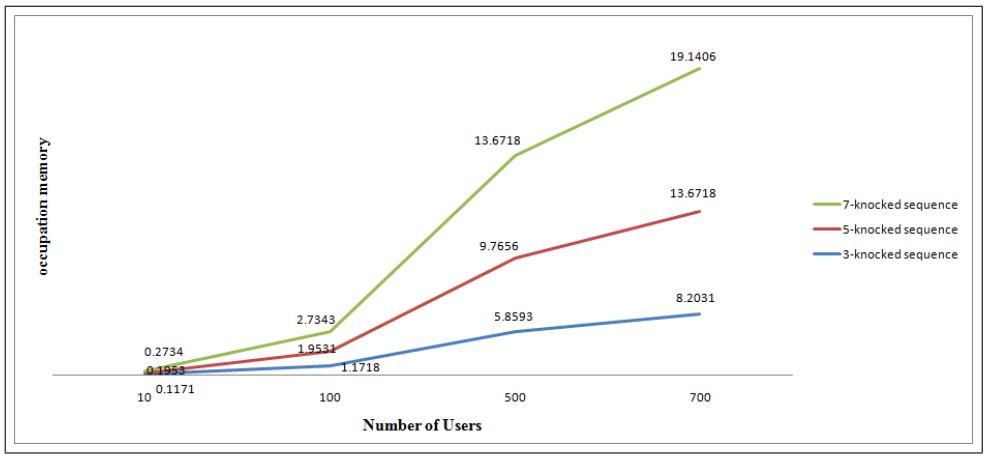

Figure 5: Occupation memory(KB) for different number of users and lengths of port knocking

This difference is about more than two times in 700 parallel authentications that shows the increment of users and length of knock sequence impose negative effect on performance. Therefore, appropriate method which can cover the short and long lengths of knock sequence limitation is highlighted especially in the MCC environment that needs high security and acceptable performance due to the shared nature and number of users.

3 Virtual port knocking authentication

Virtuallization in port knocking can be applied in two different phases; the first one that is the main focus of this subsection, employ the virtualization concept during the port knocking. While the second one use virtualization after confirming a success knock process to protect the open connection. For instance, in the SPKT method [23] in the post port knocking phase, VPN connection held between port knocker and knock server. Distributing the port knocking authentication to the virtual machines and analyzing how virtualization impacts on port knocking are the main goals of this subsection.

Knocked Virtual Machines (KnVMs): Set of virtual machines, which are allocated for port knocking authentication is known as KnVMs. These virtual machines cooperate with the cloud gateway and each other based on the defined

policy to monitor, track and record the related information about the knock packets until the knock sequence be completed.

Knocked Virtual Machines Manager (KnVMMs): When a complex policy for collaborating the KnVMs is utilized, the KnVMMs module responsible to make a connection between related KnVMs.

Initial Port Knocking Packet (IPKP): That is the primary packet, which port knocker sends to CGW. It has information about the port knocker as its content. Furthermore, IPKP brings information about key of random generation function in case of dynamic port knocking.

Knocked Virtual Machine List (KnVMList): It refers to a list which is used in KnVM to store the information of knock packets for checking the trustworthy of ordering the knocked packet. Besides, time out parameter which indicates how long that information would be remained on the list, is using to manage the time of the authentication process.
Hereinafter, virtual port knocking is defined with 4 knock sequences. In this scenario,one CGW and three KnVMs are cooperating with each other. In fact, a port knocker sends an IPKP to the CGW and the information of packet and port knocker store in a list (buffer). The CGW makes a decision about the distribution of the rest of knocked packet to which KnVM based on the port number on which received the IPKP.

If the sequence matches with the predefined knock sequence in static port knocking or with the server-generated in the dynamic one, result of success authentication returns to the CGW. Then the post port knocking phase is started in which an appropriate port will open for the specific port knocker. Moreover, based on the predefined rules in the CGW further security policies like VPN, IPSeC could be applied. Apart from that, all the buffers which are occupied for that specific port knocker would be flushed. There is necessary to point that all these processes are done only when the initial packet sent to the correct closed port in the CGW, otherwise the packet is dropped without sending any notification to the packet initiator.

Fig 6 indicates the steps of virtual static port knocking, which is completed through sending four (4) knocked packets. In the first stage, a port knocker sends IPKPto CGW further it sets a timer (T) that is used for terminating port knocking process in case of failure. Three remain knocked packets send to CGW as well. Then the port knocker checks the timer if its value is greater than "Number of packets of knock sequence $\times 10$ Sec" port knocker audits acknowledge packet that is sent via the server. If the port knocker received the acknowledge packet that means authentication done successfully.Otherwise, the process of authentication failed and the port knocker requires to start it again. The following flowchart (7)describes the steps during the process of VSPK. To make the process understandable, consider three ports for port knocking, which are P1, P2 and p3 in the CGW. Furthermore, It is assumed

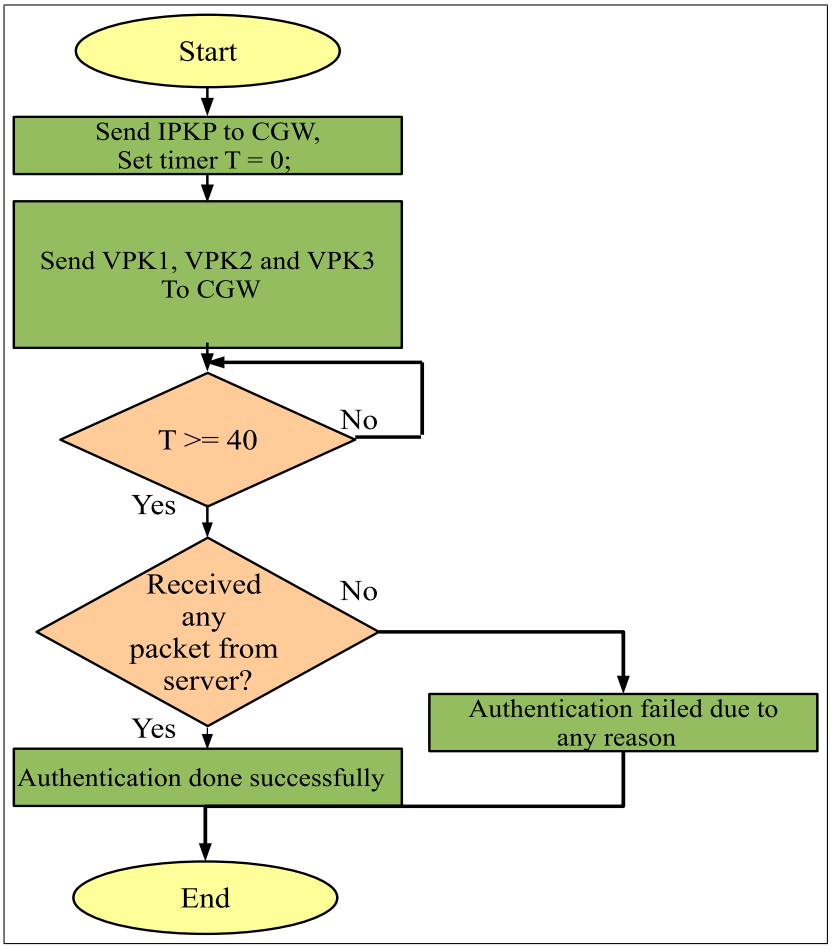

Figure 6: Virtual Static Sequence Port Knocking Authentication in client side

that three KnVMs are involved the knocking process. In fact, in this scenario for each port which is involved in the port knocking,a KnVM is considered. It is obvious that this kind of distribution policy is unsuitable for case studies in which a range of port numbers are allocated to knocking process for the CGW. Then, decision-making process requires modification based on each scenario. The first step of flowchart indicates that CGW monitors the ports.

Fig 8 determines the trend of client in virtual dynamic sequence port knocking. Port knocker generates dynamic sequence through using random generator module. Then it sends IPKP to the first generated port number of CGW. The IPKP has a key of random function as well. Further client sets a timer (T) that is used for terminating port knocking process in case of failure. Three remain knocked packets send to CGW . Then port knocker checks the timer if its value is greater than "Number of packets of knock sequence $\times$ 10 Sec" port knocker audits acknowledge packet that is sent via the server. If the port knocker received the acknowledge packet that means authentication done

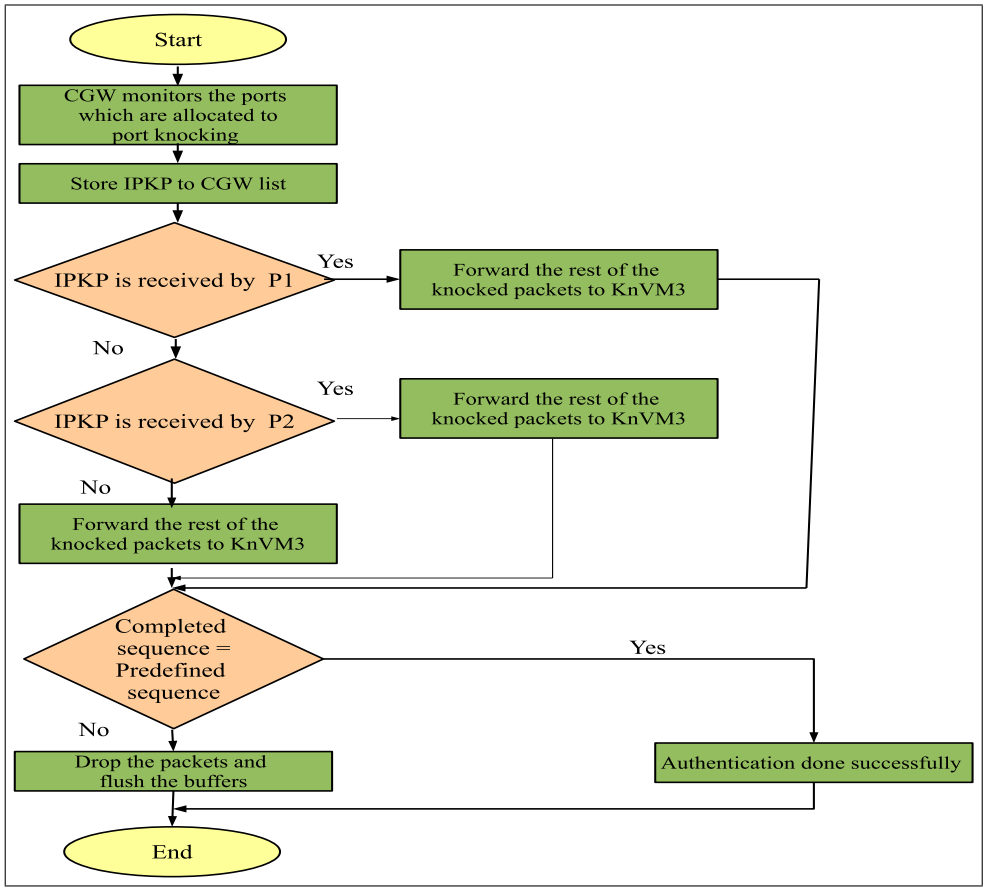

Figure 7: Virtual Static Sequence Port Knocking Authentication in server side

successfully. Otherwise the process of authentication failed and port knocker requires to start it again.

Fig 9 describes the steps during the process of VSPK. To make the process understandable let's consider three ports for port knocking in the CGW which are P1, P2 and p3. Also. It is assumed that three KnVMs are involved the knocking process. In fact, in this scenario for each port which is involved in the port knocking, a KnVM is considered. It is obvious that this kind of distribution policy is not suitable for case studies in which a range of port numbers are allocated to knocking process for the CGW. Therefore decision making needs modification based on each scenario. The first step of flowchart indicates that CGW monitors the ports. Afterwards, it stores the IPKP's information into the CGWList. If IPKP received by pn subsequently the rest of the port knocking process assigns to the KnVMn. For instance, the CGW received IPKP in the P1 then the rest of the packet forwards to the KnVM1. After that, KnVM1 monitors the ports and keeps the primary packet on the

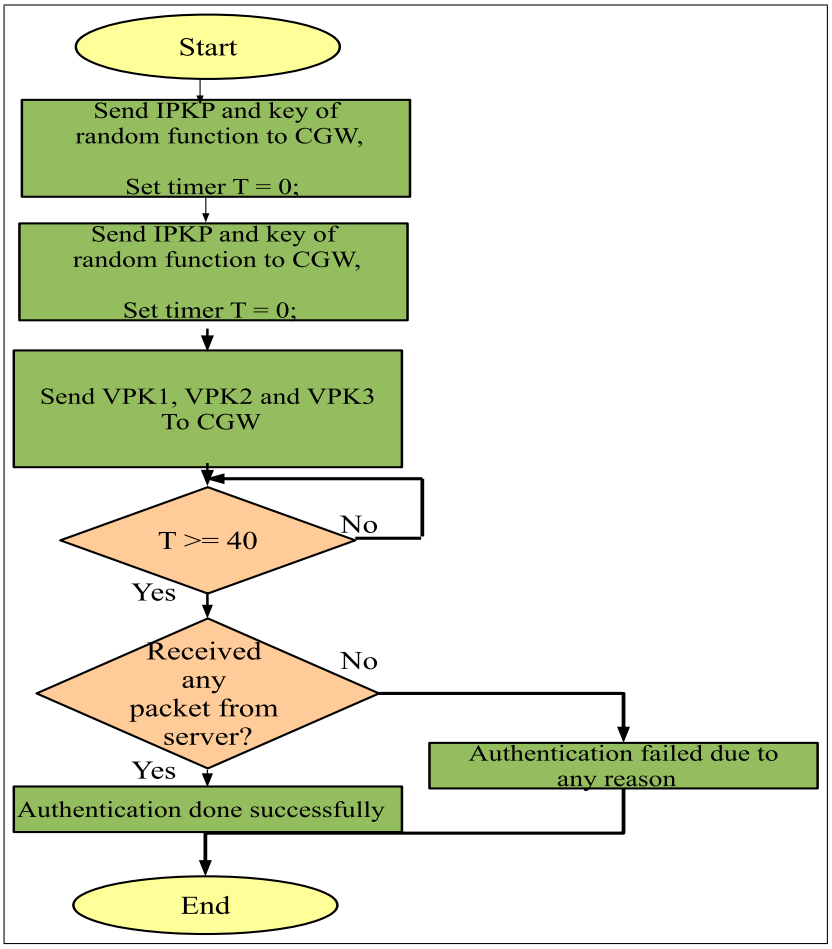

Figure 8: Virtual Dynamic Sequence Port Knocking Authentication in client side

KnVMList1. When it receives the second packet, it stores that packet on the KnVMlist2 if the preliminary one is valid. The validity of the first packet is checked by fetching the content of KnVMList1. The same operation is expected to happen in third packet. Then as a final step, the server compares the predefined knock sequence and the received one. If they are same that means authentication done successfully otherwise all involved buffers are released.

Through using virtual port knocking, which is interactive, the load which is imposed on CGW in the normal port knocking is elevated due to spread the authentication between several KnVMs Now to solve the length problem of the knock sequence, dynamic port knocking is presented in the following subsection.

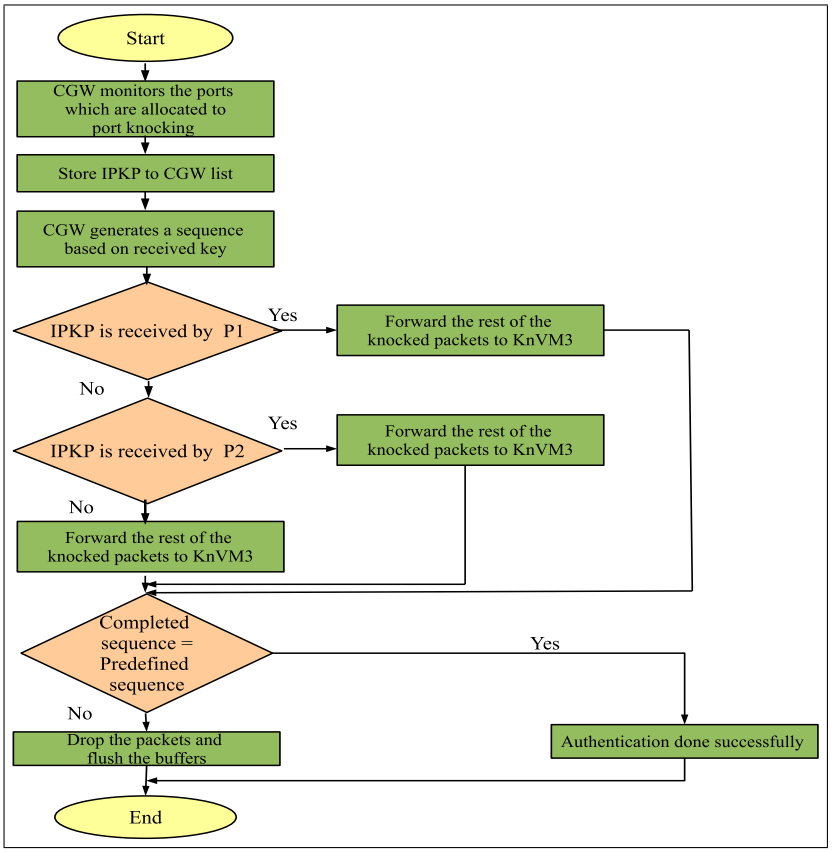

Figure 9: Virtual Dynamic Sequence Port Knocking Authentication in server side

3.1 Collected Data for virtual Port knocking

In this section data which are related to the security and performance are collected from mathematical equations and experimental computation.

3.1.1 Security measurement for the virtual port knocking

In the virtual port knocking, all the packets are sending to the CGW then it distributes to the related KnVM. Therefore, probability of hacking the sequence is same as the basic port knocking. That means the P(HS) for the virtual machine calculates through using equation (10) for the static knock sequence.

$$P(HS) = \frac{U}{2^{L_{KS}} \times L_{KS}!} \tag{10}$$

3.1.2 Performance measurement for the virtual port knocking

Efficiency of the performance could weighted in several aspects which are discussed in this section. The parameters which are used to evaluate the performance are ToPKA and LOB. As mentioned before, the first factor indicates the time of port knocking authentication in millisecond. The second factor works on the load which is imposed on the CGW and KnVMs due to the monitoring and buffering the received packets.

Time of virtual port knocking authentication: This section examines the effect of distributing the authentication among several KnVMs, on the ToPKA. Theoretically, the virtual port knocking with short length of knock sequence is expected to need less time in comparison with long knock virtual port knocking. Hereby we aim to indicate if the experimental data which is acquired supports the hypothesis presented within the theoretical discussion. To this end Table 6 shows the time which is spent for virtual port knocking, while in the cloud side just the CGW and one KnVM server the port knocker request. The values in table (6) determines the authentication time is approximately four times when the length of knock sequence is increased from three to ten. Also, comparing the time value indicates that the time difference between the short lengths sequence is more than long one. For instance, the time which is taken to have a 4-knocked virtual port knocking is roughly more than 2 times of 3-knocked one. While, this amount is just about three (3) seconds when the length of knock sequence increase from 7 to 10.

Table 6: Time of virtual port knocking authentication for single attempt

Length of Knock Sequence	3	4	5	7	10
ToPKA	31.009	64.003	85.004	127.019	130.019

Then generally the experimental time result argue that to have a secure authentication with considering time, the three-knocked authentication is more cost effective than the four knocked virtual port knocking. Whereas, in case of long length it would be completely reverse. that means ten- knocked sequence is more preferable than seven-knocked.
Also, the achieved results certifies the theoretical statements in which the ToPKA have an upward trend meanwhile the number of packets which are involved in the port knocking are increasing.

Up to now, the impact of length increment on the ToPKA is investigated. The next stage for choosing an appropriate length when time is considered, is evaluation the effect of raising the number of users which apply for authentication in a same time. Table 7 contains the ToPKA of 10 users, who use

5-knocked virtual port knocking method when the server allocates the CGW and three KnVMs to handle the authentication process.

Table 7: Time of 5-knocked virtual port knocking authentication for 10 simultaneous users

85019.42	85040.75	85008.03	85001.19	85014.05
85017.4	85043.2	85024.38	85011.18	85030.47

The collected time data shows the authentication time for 10 users is vary from "85001.19" to "85043.2" milliseconds. The average time for each user equal to 85.021 seconds. While, the gathered data for 100 parallel authentication's attempts indicate that minimum run time is "85000.43" and the maximum is "85118.96" milliseconds. The average time for each user is about "85.028". The facts and stats shows clearly each port knocker requires to spend additional 0.007 seconds when the amount of users turn to 10 times rather than before.
In the real cases MCC serves the numerous number of users therefore 10 or 100 parallel authentications is not close to the fact. Hence, the time of authentication for 500 users who run the port knocking process in the same time without even milliseconds delay is gathered to evaluate a real case study. The achieved data determines the ToPKA changes between "85020.74" and "100868.93". Although the time difference between minimum and maximum time of authentication for 500 users is not as little as before, the average time for each user equal to "89.1160" seconds. It shows that each user need more four seconds to finish the authentication process regardless the result of it while the number of users increase dramatically from 10 to 500.

The smooth slow upward trend of ToPKA for virtual port knocking indicates that this method potential to be considered as a suitable method in case of having numerous concurrent authentications. But this statement is correct when performance evaluate by only time factor.

Measuring the load of buffering in virtual port knocking: In the virtual port knocking, the load of buffering distributes among the virtual machines as mentioned before. The practical observation shows only information about the first packet of authentication is stored by the CGW. While the rest of information forward to the KnVMs. Hence, the LoB for the CGW in the virtual port knocking is equal to:

$$LoB_{CGW} = U \times 32bits \tag{11}$$

That U indicates the number of users who apply for authentication in the same time. Also, the load which is imposed to each KnVM computes by Equation

(12)

$$LoB_{KnVM} = U \times (L_{ks} - 1) \times 32bits \tag{12}$$

The above equations indicate that the LoB which is impose to the CGW is completely independent to the length of the knock sequence. In fact, the mentioned load increase directly through raising the number of users.
Table 8 shows the space which is expected to occupy for monitoring the various number of users. The values are in KiloByte. The mentioned space is only for

Table 8: Load of buffering of the CGW in virtual port knocking

Number of users	10	100	500
LoB	0.0390625	0.390625	1.953125

authentication process and occupied space for post-phase is not count in this calculation. Also, Table 9 indicates the allocated space through using KiloByte unit, for tracking the authentication in each KnVM. We assume that the whole system works based on the normal and equal distribution, which means if the number of users who use the sequence which is assign to the KnVM1 to be processed equal to X. The same number of port knockers use the sequences which are assigned to KnVM2 and KnVM3.

Table 9: Load of buffering of each KnVM in virtual port knocking

Number of users	10	100	500
LoB for 3-knocked sequence	0.078125	0.78125	3.90625
LoB for 5-knocked sequence	0.15625	1.5625	7.8125
LoB for 7-knocked sequence	0.234375	2.34375	11.71875

4 Results and Discussion

Fig 10 shows the average time which is spent by each user to complete the basic or 5-knocked virtual port knocking. The horizontal axis represents the number of simultaneous attempts of authentication whereas the vertical one belongs to the time of authentication. According to the figure, run time rises dramatically from 5.03 to 85.02 seconds in case of ten parallel authentications when users utilize virtual port knocking instead of basic one. Which means time is roughly seventeen times when port knockers use the virtual port knocking authentication.
The upward trend of consumed time remains constant when the number of users who want to be authenticated in the same time, increases from 10 to 100 users. While authentication of huge number of users for instance 500 parallel

users consumes 14-fold time in comparison with the basic port knocking. The main reason of a feasible difference is back to the usage of KnVMs during the virtual port knocking and distributing the authentication. The negative effect of distribution in the time of authentication give high opportunity to basic authentication to be selected by port knockers in contrast with the virtual port knocking. The huge difference indicates that distributing the authentication among KnVMs is a time-consuming process. Afterward, through comparing the imposed load on the CGW, a proper result is figured out that shows using the virtual port knocking is cost effective or not?

Fig 11 and Fig 12 are comparing the amount of the allocated memory space

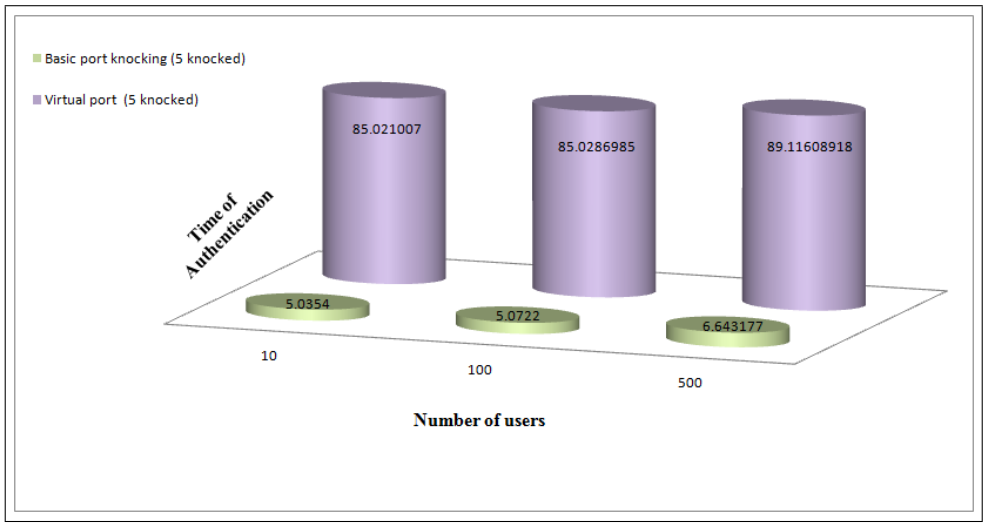

Figure 10: Comparison of ToPKN of virtual port knocking and Basic port knocking

in case of using basic and virtual port knocking. The horizontal axis belongs to the various number of users who apply for port knocking authentication in a same time. While, the vertical axis reveals the amount of the occupied memory space through using the kilobyte unit.

In Fig 11 first three bar charts in each column represent the required memory for 3, 5 and 7 length of basic port knocking, respectively. Whereas, the fourth chart is representative of the memory space which is granted for the authentication process in the CGW of virtual port knocking. Take a quick look to this figure makes it clear that in virtual port knocking, the memory space for buffering would be increased through expanding the length of knock sequence and number of port knockers as well. Whereas, in the virtual port knocking in case of examine the imposed load on the CGW, the length of knock sequence do not have any effect on increasing the amount of memory. Indeed, the required memory space is rising gradually by adding the number of users.

The bar charts show clearly that the allocated memory space for all methods

is less than 0.3 KB for 10 concurrent authentications. Although the largest amount of space allocates to the 7-knocked basic port knocking, the required memory space has not been significant difference for all methods in this case. Growing the number of simultaneous authentications emphasizes the benefit

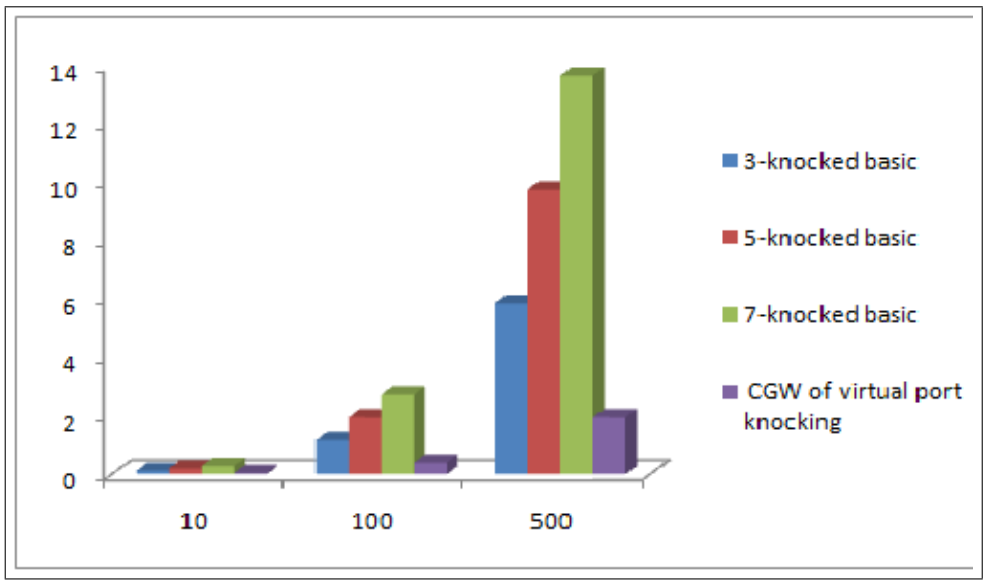

Figure 11: Allocated memory space to port knocking process for basic and CGW of virtual port knocking

of using the virtual port knocking rather than the basic port knocking in terms of memory utilization. For instance, the memory space that is required for 100 users in the virtual port knocking is 10 times in comparison with before, but it is still below the 0.5 KB. While basic port knocking needs "L $\times$ 10-fold" memory in the same condition. Therefore, the amount of required memory for 7-knocked basic port knocking is near to 4 KB.
When the number of users reaches to 500, the occupied memory is around to 6 KB, 3 KB and 14 KB for 3, 5 and 7-knocked basic port knocking in order whereas, the fourth bar chart still does not reach to 3 KB. Hence, in case of 500 concurrent users, virtual port knocking enhances the performance with reducing allocated buffering space in the CGW around 4% for short length knock sequence and decreases it around 12 % in case of using long length.
The main reason behind these results is the distribution process, which is performed in the virtual port knocking and reduces the load on the CGW.

The comparison results of the imposed load on each KnVMs and basic port knocking is shown by Figure 12. Both basic and virtual port knocking, have a same trend in memory occupation when we discuss about the load of KnVMs in virtual knocking method. That means both has a dramatics upward trend when the number of port knockers increased from 10 to 500. Despite

the mentioned similarity, still the amount of memory which is required by the KnVMs is less than the required amount in the basic port knocking. This

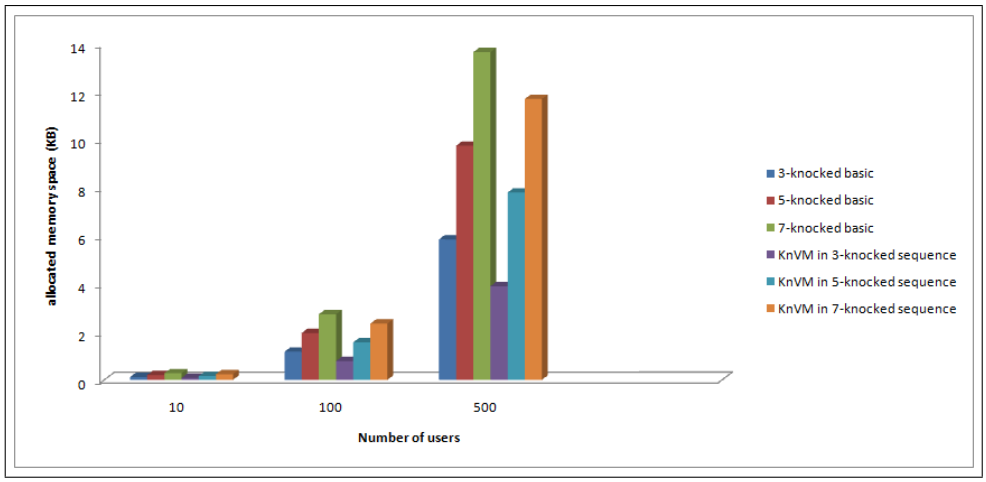

Figure 12: Allocated memory space to port knocking process for basic and virtual port knocking

differentiate reaches to 2 KB in case of 500 parallel authentications. This difference would be high if the number of KnVMs grow from one to three because the number of request, which is sent by users would be divided into the number of KnVMs.

5 Conclusion and Future Direction

To conclude, this paper argues that short length of knock sequence makes authentication vulnerable by increasing the chances of hackers to capture and decrypt the packets and find the Knock sequences. Moreover, selecting the long knock sequence requires more buffers to monitor them and imposes buffering load due to occupation more memory space to the server side. Besides, long knock sequence makes the authentication process slower in comparison with using the short length knock sequence. Moreover, a new technique is presented by this paper which is called virtual port knocking that could be consider as solution to mitigating the security and performance's issues. in decision-making step to choose between virtual and basic port knocking, the administrators decide based on the performance's requirements because both methods have same P(HS). Therefore, if the network does not face limitation of memory space, virtual port knocking would be a better option due to less process time that it need. Otherwise, the time could be placed in the second level because overflowing of buffers lead to drop the authenticate request beside the unauthenticated ones.

Although the load of buffering is reduced through using the virtual port knocking, the time of authentication is raising up dramatically due to the authentication distribution between virtual machines. The increased time has a reverse effect on the performance enhancement specially in case of users enlargement. Therefore, hereinafter we point to the two methods which could be considered as solution to optimize the port knoking method for MCC in the near future.

1. Using dynamic concept to have dynamic length during the authentication. That means each client use different length in each attempts. Dynamic lengths are selected randomly. This method theoretically would help both security and performance. Because through using various lengths, hacker needs more effort to find the correct sequence. Also, the average time which is spent for authentication would be reduced.

2. Combining virtual method and dynamic one to generate new method which enhances buffering load, time of authentication and security.

References

[1] S. Weiguang and S. Xiaolong, "Review of Mobile cloud computing," in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, 2011, pp. 1–4.

[2] M. D. Ryan, "Cloud computing security: The scientific challenge, and a survey of solutions," *Journal of Systems and Software*, no. 0. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121212003378

[3] A. N. Khan, M. L. Mat Kiah, S. U. Khan, and S. A. Madani, "Towards secure mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1278–1299, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X12001598

[4] E. Cohen and S. A. Cohen, "Authentication: Hot and cool," *Annals of Tourism Research*, vol. 39, no. 3, pp. 1295–1314, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0160738312000333

[5] B. Lakshmiraghavan, "Two-factor authentication," in *Pro ASP. NET Web API Security*. Springer, 2013, pp. 319–343.

[6] D. Coffin, "Two-factor authentication," in *Expert Oracle and Java Security*. Springer, 2011, pp. 177–208.

[7] F. Aloul, S. Zahidi, and W. El-Hajj, "Two factor authentication using mobile phones," in *Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on.* IEEE, 2009, pp. 641–644.

[8] R. Support, "Port Knocking," 2013. [Online]. Available: http://www.rackspace.com/knowledge

[9] L. J. White and H. K. N. Leung, "A firewall concept for both control-flow and data-flow in regression integration testing," in *Software Maintenance, 1992. Proceerdings., Conference on*, 1992, pp. 262–271.

[10] O. Nurika, A. H. B. Muhamad Aminz, A. S. B. A. Rahman, and M. N. B. Zakaria, "Review of various firewall deployment models," vol. 2, 2012, pp. 825–829. [Online]. Available: http://www.scopus.com/inward/record.

[11] Krzywinski, "Port Knocking: Network Authentication Across Closed Ports," *SysAdmin Magazine*, vol. 12, p. 6, 2003.

[12] S. Prowell, R. Kraus, and M. Borkin, "Chapter 1 - Denial of Service," in *Seven Deadliest Network Attacks.* Boston: Syngress, 2010, pp. 1–21. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9781597495493000018

[13] M. Doyle, "IMPLEMENTING A PORT KNOCKING SYSTEM IN C," Ph.D. dissertation, 2004.

[14] L. Jiun-Hau, S. Lee, I. Ong, L. Hoon-Jae, and L. Hyotaek, "One-Time Knocking framework using SPA and IPsec," in *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, vol. 5, 2010, pp. V5–209–V5–213.

[15] "Port Knocking: Beyond the Basics." 2005.

[16] M. Krzywinski, "Port knocking from the inside out," *Communication*, 2005. [Online]. Available: http://www.portknocking.org

[17] E. Vasserman, N. Hopper, and J. Tyra, "SilentKnock: practical, provably undetectable authentication," *International Journal of Information Security*, vol. 8, no. 2, pp. 121–135, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10207-008-0070-1

[18] T. Popeea, V. Olteanu, L. Gheorghe, and R. Rughinis, "Extension of a port knocking client-server architecture with NTP synchronization," in *Roedunet International Conference (RoEduNet), 2011 10th.* IEEE, 2011, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5993704

[19] D. D. Dhobale, V. R. Ghorpade, B. S. Patil, and S. B. Patil, "Steganography by hiding data in TCP/IP headers," in *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, vol. 4, 2010, pp. V4–61–V4–65.

[20] W. Du and L. Wang, "Context-aware application programming for mobile devices," Montreal, Quebec, Canada, pp. 215–227, 2008.

[21] A. Manzanares, J. Márquez, J. Estevez-Tapiador, and J. Castro, "Attacks on port knocking authentication mechanism," *Computational Science and Its Applications–ICCSA 2005*, pp. 1292–1300, 2005. [Online]. Available: http://www.springerlink.com/index/LRAB9TF4EF199AJG.pdf

[22] V. Srivastava, A. K. Keshri, A. D. Roy, V. K. Chaurasiya, and R. Gupta, "Advanced port knocking authentication scheme with QRC using AES," in *2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*. IEEE, Apr. 2011, pp. 159–163. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?

[23] P. Mehran, E. A. Reza, and B. Laleh, "SPKT: Secure Port Knock-Tunneling, an enhanced port security authentication mechanism," *2012 IEEE Symposium on Computers & Informatics (ISCI)*, pp. 145–149, Mar. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6222683