# AN IMPROVED PORT KNOCKING AUTHENTICATION FRAMEWORK FOR MOBILE CLOUD COMPUTING

***Muhammad Shiraz[1], Laleh Boroumand[2], Abdullah Gani[3], Suleman Khan[4]***

[1]Department of Computer Science,
Federal Urdu University of Arts, Science and Technology, Islamabad, Pakistan
[2,3]Centre for Mobile Cloud Computing Research (C4MCCR)
Department of Computer Systems and Technology, Faculty of Computer Science and Information Technology
University of Malaya, 50603 Kuala Lumpur, Malaysia
[4]Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE18ST, UK

Email: drmuhammadshiraz@fuuastisb.edu.pk[1], laleh.b127@gmail.com[2], abdullah@fsktm.um.edu.my[3], suleman.khan@northumbria.ac.uk[4]

DOI: https://doi.org/10.22452/mjcs.vol32no4.2

## *ABSTRACT*

*The latest developments in mobile cloud computing (MCC) have changed user's priorities for computing. However, the change towards MCC brings new challenges to cloud service providers and administrators. Authentication is one among the challenges categorized in the classification of security  issues for MCC. Port knocking authentication method eliminates user's collaboration during the authentication process. Thus, such technique has the potential to be applied on the MCC environment which can ensure reliable communication. However, current port knocking authentication techniques lack of addressing the issue of knock-sequence length. It is challenging to deploy appropriate length sequence for port knocking authentication for the reason that shorter length knock-sequence degrades security, whereas, deploying longer length sequence involves performance issues in terms of time and buffer management. This paper proposes a dynamic length port knocking authentication framework which addresses the issue of security degradation and optimizes performance in terms of time up and buffer managment. We employ MikroTik RouterOS for the evaluation of the proposed technique. Analysis of the results shows that dynamic length port knocking authentication technique improves performance in terms of time up to 23% and buffer management up to 28% by reducing the imposed load. Furthermore, by deploying dynamic length (DL) and pool of length (PoL), the proposed method reveals high security, which decreases the probability of hacking knock-sequence near to zero for a number of parallel authentication requests. Hence, dynamic length port knocking authentication technique provides an optimal solution for reliable communication in MCC.*

**Keywords:** *Mobile Cloud Computing (MCC), Authentication, Port Knocking, Security, dynamic port-knocking*

## 1.0 INTRODUCTION

The latest developments in mobile computing technology have promoted Smart Mobile Devices (SMDs) worldwide. SMDs are employed in various domains including health-care, disaster recovery, education, crowd management, and IT business. However, despite of considerable technological advancements, full functionality and adoption , mobile devices are still hindered by resource poverty problems. Therefore, mobile cloud computing (MCC) leverages resource rich computational clouds for mitigating resources limitations in SMDs. However, a shift from traditional devices as a client (Like PCs) towards the SMDs [1]involves security challenges to MCC ranging from authentication to resource isolation [2]. To reduce these challenges, cloud security service which is known as security-as-a-service is deployed [3,4] which provides various levels of security which are divided into normal security service (NSS) and critical security service (CSS) [5]. One of the essential security service in security-as-a-service model is authentication which ensures only authorized users access the SaaS applications through utilizing seamless, single-sign on granted access [2, 6]. Authentication process provides different levels of security based on the requirements of the applications which are classified based on their factors. For instance, username and password, multi-factors, knowledge factor, possession factor and inherent factor authentication techniques.

Despite high demand for username and password to authenticate the users, a number of attempts are made to make these kind of authentications more complicated for reaching the higher security beside the reduction of management cost. First familiar authentication method which utilizes username and password is single sign-on (SSO), which allows users to authenticate once and gain access to multiple, authorized systems [6, 7]. However, SSO includes security flaws, for the reason that breach of the password security results in unauthentic access to all the system resources. Commonly, SSO is deployed along with multi-factors authentication [8] which is an interactive authentication technique. It uses third party authentication (for instance, sent code to the phone number of the user) for ensuring trusted party in performing operations [9-13]. Multi-factors authentication methods is among the popular methods, for the reason of controlled budgets and providing convenient access to the data and services for users [14]. Each of the knowledge [15] possession and inherent factor [16, 17] authentication can be involved in multi-factors authentication technique. Nonetheless, multi-factors authentication method suffers from communication limitation. For instance, in some cases delayed SMS or weak mobile network coverage in some areas can bring timeout error and terminate the session in the middle of the process. Moreover, some of these techniques require complex computation and they are user-dependent [18].

It is expected to consider resource management issues in case of applying severe authentication technique. In addition, network has potential to face security breaches in case of using authentication techniques in which third parties are involved [19, 20]. Simple authentication methods employ simple username and password [21] and consume less resources; however, they are not reliable enough to be deployed on the sharing networks with a large number of users. Therefore, MCC needs a lightweight inter-operable authentication method [22].

Port knocking authentication technique uses sequence of ports as its password [23] in the authentication process. Each client sends the authentication packets to the specific ports of server's gateway sequentially. Port sequence is expected to be defined statically [24, 25] or dynamically [26] depending upon the severity of the required security level. Static port sequence means ports and their order are defined for both client and server. User sends knock packets to the predefined ports, in the meanwhile, server monitors the ports and buffers the received knock packets without sending any notification message to client as delivery confirmation [24, 25]. When the knock sequence is completed, the server compares it with the predefined one. Then, if the result of comparison is true, one port is opened for client. Since all the ports involving to the port knocking authentication method are closed and server does not send any notification (ACK packets) to the client during the authentication process, therefore, this method is known as silent authentication method. However, a major drawback of static port knocking is about detection of valid knock sequence by attacker through sniffing enough traffic [27]. To address this issue, dynamic port knocking authentication is introduced. Dynamic knock sequence refers to the sequence of ports which is generated randomly [26]. In this situation, knock packets contains a key which is used by a server to generate knock sequence after the client completes the sequence [28-31]. Similarly, the static knock sequence [24, 25] server compares the received and generated knock-sequence and upon similarity it navigates further the process to the post authentication procedure.

Knowledge center of Rackspace [32], which is the open cloud company, reported the testing process of port-knocking authentication on their cloud servers is ongoing, port-knocking needs more consideration depending upon public or private cloud. Knock sequences are like passwords which usually remains private. However, unlike a password, knock sequence cannot be easily encrypted. Moreover, regardless using the static or dynamic Knock-sequence, its length is a factor that affects on the performance and security if it is not chosen properly. If the selected knock sequence is too short, sniffers can find the range easily by capturing the traffic [33, 34]. Conversely, the long knock sequence imposes overhead to server due to the requirement of large size buffer for monitoring the ports. Furthermore, long knock sequence makes the authentication process longer than before [33].

According to the results which are obtained from [35], hacking probability of knock-sequence exponentially decreases by increasing the length of knock-sequence. Moreover, effect of length on the performance is evaluated in [35] by considering two parameters; time of authentication and load of buffering. The achieved results determine that consumed time for authentication is increased via using the long knock-sequence. Furthermore, increasing the length of knock-sequence leads to impose more load to the cloud gateway due to allocating more buffers to monitor the ports. Therefore, deploying an appropriate sequence length for dynamic port knocking authentication method in MCC is still challenging.

This paper as its significant contribution proposes an optimal port knocking authentication framework which is capable to switch between various length of knock sequences in each authentication's attempt. The framework employs dynamic length port knocking authentication technique which addresses the trade off between security and performance. We employ MikroTik RouterOS for the evaluation of the proposed technique. Analysis of the results shows that the proposed technique enhances performance in terms of time up to 23% and buffer management up to 28% by reducing the imposed load. In addition, by deploying dynamic length (DL) and pool of length (PoL), the proposed method reveals high security, which decreases the probability of hacking knock-sequence near to zero for a number of parallel authentication requests. The rest of this paper is organized as follows.

Section 2 discusses the existing authentication methods. Section 3 explains the algorithms and parameters of the proposed technique. Section 4 describes the methodology to evaluate of the proposed technique. Section 5 presents and discusses the results of the analysis. Finally, section 6 concludes the paper and suggests some future directions.

## 2.0 RELATED WORK

The following section reviews the advanced port-knocking methods which are classified based on the static or dynamic knock-sequence. Advanced port-knocking methods include the methods that combine basic port-knock with other security methods or attach additional features to basic port-knocking to make it more secure.

TCP stenography [36] and fast cryptographic Message Authentication Code (MAC) [37] are merged in Silent knock [24]. It offers lightweight authentication that imposes minimal computational overhead on the system. MAC contains information about both authentication and integrity of the message. Block ciphers generates MAC, which means it operates on *n*-bits blocks of data and symmetric secret key is used which for modifying the message [38]. Silent knock includes sknockd and knockProxy. The server side deploys sknockd between the TCP/IP stack and client. SYN packets sent by the client are verified by the sknockd based on the IP address and stenogotext is retrieves by using a a counter which belongs to the IP address and stenographic algorithm ,. If the result of the verification is successful, then the packet passes the sknockd and reaches the TCP/IP stack; otherwise it is dropped. The second program reads a configuration file on the client side and to figure out the server which offers the silent knock to different services. In addition, a shared secret key and counter is collected from this file. Then, a MAC is computed and the above information is encoded and sent to the sknockd by the knockproxy

The silent knock is a lightweight method, which protects the network against a reply attack of the global active adversary through an incremental counter for each attempt to access. The experimental results show that the average response time is increased by using sknockd instead of SSH [39]. Therefore, administrators need to figure out whether or not the cost of delay imposed by the silent knock to the network is affordable. Secure Port-knock-Tunneling (SPKT) [25] contains two phases for authenticating users. The first one is a modified port-knocking which overcomes the DoS knock attack; whereas the second one is a connection based on tunneling that prevents the network from a NAT knock attack. Each packet has a pass-phrase; therefore, besides the sequence of ports the pass-phrase is checked. If the pass-phrase is similar to the predefined one and the sequence is valid, then the second phase is started, thereby triggering a VPN connection. Hence, unique username and password are required for users and there is no concern about the NAT knock attack. If the pass-phrase is identical for all packets that belong to the knock process, it could be hacked through packet capturing. Otherwise, the server needs to keep different pass-phrases for packets and in the case of a large length sequence, it imposes additional load on the server.

The advanced port-knocking authentication scheme with QRC using Advanced Encryption Standard (AES) [29] method randomizes the source IP address and port number during the port-knocking process through the help of Quadratic Residue Ciphers (QRC) [40, 41]. The port-knocker sends an SMS that requests the One Time Password (OTP) to the SMS server. The OTP is generated based on a time factor and therefore, there is no chance of having a duplicate of the OTP for the same user ID and is protected from DoS attacks. The SMS server replies to the knocker with the message that contains a timestamp that is used for authenticating the OTP. The second field of the SMS is a One Time key that is 256 bits, which is used by the AES [42, 43] to encrypt the knock sequence.

The last field is a random number "R" that is generated by the Pseudo Random Number Generator (PRNG) [44, 45] which is used as a key in the QRC. This SMS is also forwarded to the server, which performs authentication in the next step. In this method, the client IP address is changed dynamically to mitigate the risk of attack. In fact, the OTP and the last 8 bits of the IP address (last octet) are employed by the AES encryption algorithm to generate the knock sequence. Each time, these bits are XORed with R to generate a new IP address. This process continues until the completion of the knocking sequence. On the other side, the server has R, OTP and other values, which are used in the encryption and it generates the knock sequence and compares it to the received one. The one time knocking framework employs the SPA and IPsec [30] method and SMS server hosts a random number generator (RNG). In the initial step, the user sends an SMS to the server and the server identifies it by the user’s number, which was previously registered in the server. This phase helps in the prevention of a DoS attack. Then, the server generates an OTP and sends it back to the client through the same Out of Band (OoB) channel that is used by the client. This message contains a time stamp, random port as well as OTP. The random port and OTP are transferred to the SPA client as an input value that is used by the Key Derivation Function (KDF) [46]. Four keys are prepared by this function – *k1; k2; k3; k4* – which are used in that order for data encryption, MAC calculation for SPA and the two last for IPSec VPN connection [47-49].

The basic port knocking method has undergone several changes to overcome issues including Plain text port sequence, Network Address Translate (NAT) Knock [50], Denial of Service (DoS) knock attacks [51, 53], appropriate length of knock sequence [30], out of order packet delivery and lack of association between authentication and connection [27]. During this transformation, the knock-sequence is changed from static to dynamic mode and plain knock sequence is converted to the encrypted sequence, as it assists in making port knocking secure [24, 52]. However, such techniques lack of deploying appropriate length for the knock sequence which needs to be addressed by port knocking authentication to make it suitable for MCC [29].

Static knock-sequence [25,39] is replaced by dynamic knock sequence to achieve higher security by utilizing random knock-sequence instead of the predefined knock-sequence [29, 30, 53]. Although dynamic knock-sequence reduces the probability of hacking the knock-sequence, it still faces security and performance challenges during the authentication process for the reason of employing fixed length of knock-sequence [35]. Based on the results in [35], 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, whereas, selecting the long knock sequence requires more buffers to monitor them and imposes buffering load for the reason of additional memory space requirment on the server side. Besides than that, long knock sequence makes the authentication process slower in comparison with using the short length knock sequence. Thus, an optimal length port knocking authentication technique is required which is able to make a balance between security and performance.

## 3.0 PROPOSED OPTIMAL PORT KNOCKING AUTHENTICATION FRAMEWORK FOR MCC

To address the the trade off between security and performance, we propose an optimal framework for port knocking authenticationin MCC. The framework employs dynamic length port knocking authentication method where in each attempt of authentication, port knocker uses various knock-sequence in terms of length and port numbers. Therefore, both security and performance issues which are raised due to using short and long knock-sequence, are addressed. Fig. 1 shows the proposed framework which is composed of two modules, port knocker and port knocking. Port knocker module is deployed on the mobile device and it includes knock-sequence manager and knock-packet transfer manager. Port knocking module is empoyed on the cloud gateway and composes monitoring/buffering manager and comparing module. Details of the framework are as follows:

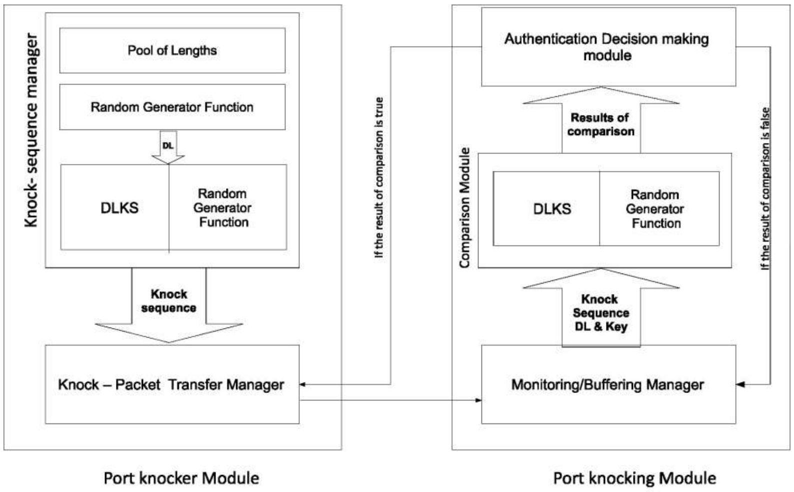
Knock- sequence manager
Pool of Lengths
Random Generator Function
DL
DLKS
Random Generator Function
Knock sequence
Knock – Packet Transfer Manager
If the result of comparison is true
Authentication Decision making module
Results of comparison
Comparison Module
DLKS
Random Generator Function
If the result of comparison is false
Knock Sequence DL & Key
Monitoring/Buffering Manager
Port knocker Module
Port knocking Module

**Fig. 1:** Proposed Framework for Dynamic Port knocking

***3.1 Knock-sequence Manager***: Knock sequence manager is responsible for generating the knock-sequence. To this end, it works with the following components: (1) Pool of Length (PoL) is maintained in the form of a linear list that keeps various lengths which are valid to be used by port knocker. Dynamic Length (DL) is selected among the members of PoL; (2) Random length selection function accepts the PoL as an input value and randomly selects the DL value from the PoL; (3) Dynamic Length Knocked Sequence (DLKS) is a two dimensional array which stores knock sequence according to the length of the sequence; and (4) Random knock-sequence generator function is utilized by DLDKS method, which accepts port ranges and DL which is fetched from PoL as its input value and generate knock-sequence. Knock-sequence manager selects the length of sequence randomly among the available lengths of

PoL by using the random length selection function. DL is used as an index to fetch the knock sequence from DLKS array in the static scenario whereas, in the dynamic knock sequence process the fetched DL along with port range are transferred to the random knock-sequence function.The knock-sequence is sent to the knock-packet transfer manager regardless it is fetched from DLKS or generated by random function.

***3.2 Knock-packet Transfer Manager****:* It is responsible for sending the knock-packets to the server and waiting for the authentication results based on the timer which is set by the knock-packet Transfer Manager when it sends the first knock packet to the server. Valid interval time is calculated based on DL ×10 milliseconds. On server side, the authentication process requires port knocking module which works with Monitoring/Buffering Manager and Comparison Module. Monitoring/Buffering Manager module is responsible for monitoring the ports which are involved in the port knocking process. It fetches the dynamic length (DL) from the first packet and buffer the received packets. When the authentication is completed, the received knock-sequence and fetched DL is sent to the comparison phase. Comparison Module fetches the pre-defined knock-sequencein the DLSKS based on the received DL and compares it with the received one.The comparison output enters into the authentication decision making module. In DLDKS, comparison module has random generator function for generating random knock-sequence based on the received DL and random key which are sent by port knocker.

***3.3 Authentication Decision Making Module*****:** True result of the comparison indicate legitimate user and therefore, an acknowledge packet is sent by server and the post authentication phase is started to establish a safe connection between port knocker and server. Otherwise, the buffers which are occupied for this port knocker is flushed. Dynamic length framework shows the way dynamic concept is employed in port knocking to have various length of knock-sequence. It also states the effect of using dynamic length on both security and performance.Two scenarios are considered for the dynamic length port knocking for further illustration. In the first scenario the length of knock sequence is dynamic, however the sequence is defined statically. This method is known as dynamic length static knock sequence (DLSKS) which is the main focus of this paper. It means by using the dynamic length, we can apply static knock-sequence with less risk in comparison with static length static knock sequence [24, 35]. In the meanwhile, this method eliminates the complexity of randomly generating the knock-sequence on network insight. The second scenario supports dynamic knock sequence besides the dynamic length (DLDKS). In the DLDKS, port knocker module is capable to switch between different lengths and the knock-sequence is determined by a random generator function.

***3.4 Dynamic Length Static Knock Sequence (DLSKS) Approach:*** Fig. 2 indicates a port knocker's trend during the DLSKS port knocking. First, DL is selected from PoL. Then the DLKS is searched based on the fetched DL to find the predefined knock sequence. After that, the port knocker sends the knocked packets and sets the timer. Whenever, the timer exceeds time limit, which is predefined for authentication, port knocker expects to receive a notification packet, otherwise, the authentication process is reinitiated. The notification message is sent only to the legitimate port knocker. For illegitimate user, the server remains silent.

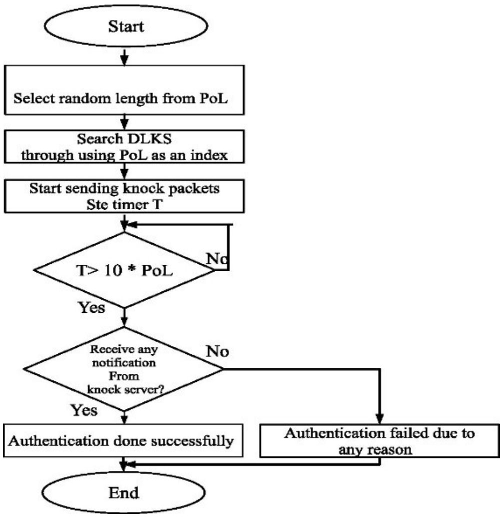
Start
Select random length from PoL
Search DLKS through using PoL as an index
Start sending knock packets Ste timer T
T> 10 * PoL
No
Yes
Receive any notification From knock server?
No
Yes
Authentication done successfully
Authentication failed due to any reason
End

**Fig. 2:** Dynamic Length Static Knock Sequence (DLSKS) Port Knocking Authentication on Client Side

Fig. 3 indicates the steps required on the server side to pass DLSKS port knocking authentication. Server fetches the DL from the first packet and monitors and records all the ports involved in port knocking authentication. Finally, the received sequence is compared with the existing one in the DLKS. If they are matched, then user passes the authentication phase successfully.

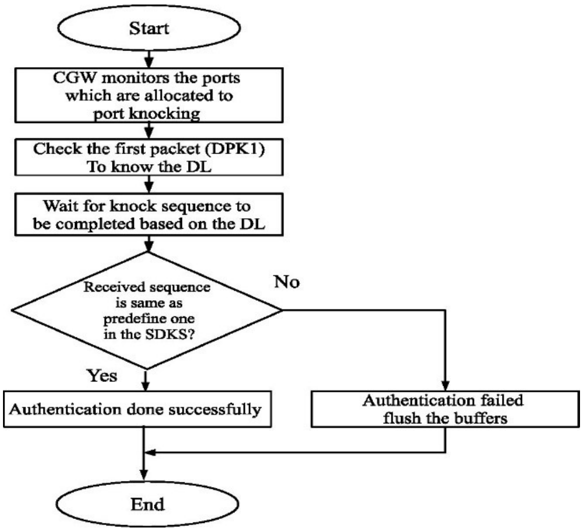
Start
CGW monitors the ports which are allocated to port knocking
Check the first packet (DPK1) To know the DL
Wait for knock sequence to be completed based on the DL
Received sequence is same as predefine one in the SDKS?
No
Yes
Authentication done successfully
Authentication failed flush the buffers
End

**Fig. 3:** Dynamic Length Static Knock Sequence (DLSKS) Port Knocking Authentication on Server

***3.5 Dynamic Length Dynamic Knock Sequence (DLDKS) Approach:***The authentication process is different in the DLDKS for both client and server in terms of its random knock-sequence. Fig. 4 presents the step inolved in authentication of each port knocker. Port knocker module selects an arbitrary length from PoL and assigns it as an input parameter to the random generator function. Therefore, the dynamic sequence is generated based on the selected length. Then, a timer is set in the port knocker's device. Furthermore, it sends the knocked packets along with the random key to the closed ports of server. After passing the specified time period, if the port knocker receives any notification that means server recognized it as an authorized user. Otherwise, the port knocker needs to restart the authentication process.

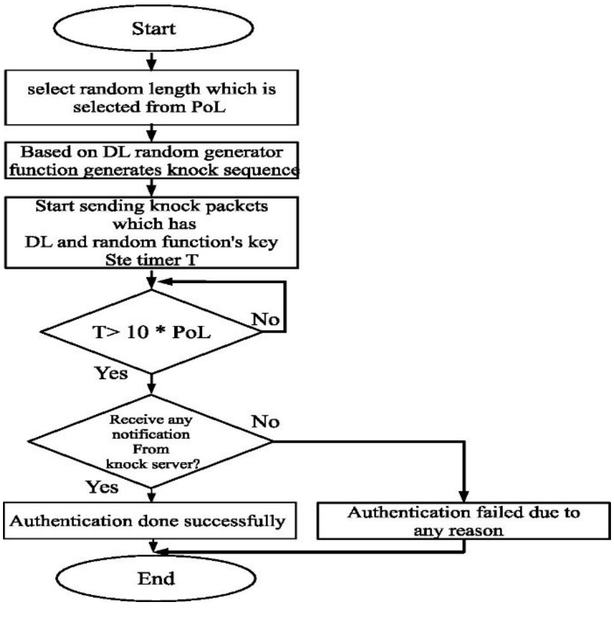
Start
select random length which is selected from PoL
Based on DL random generator function generates knock sequence
Start sending knock packets which has DL and random function's key Ste timer T
T> 10 * PoL
No
Yes
Receive any notification From knock server?
No
Yes
Authentication done successfully
Authentication failed due to any reason
End

**Fig. 4:** Dynamic Length Dynamic Knock Sequence (DLDKS) Port Knocking Authentication in client side

Port knocking module in the server retrieves PoL and random key after receiving the first dynamic port knocking packet which is shown by Fig. 5. It generates knock-sequence based on the received PoL and a key value. In the meanwhile, it sets a timer for ensuring that the authentication is performed in the specified time. If the knock sequence is completed during that interval, server compares it with the generated sequence. Otherwise, all the lists which are involved in the authentication process for that port knocker are flushed.

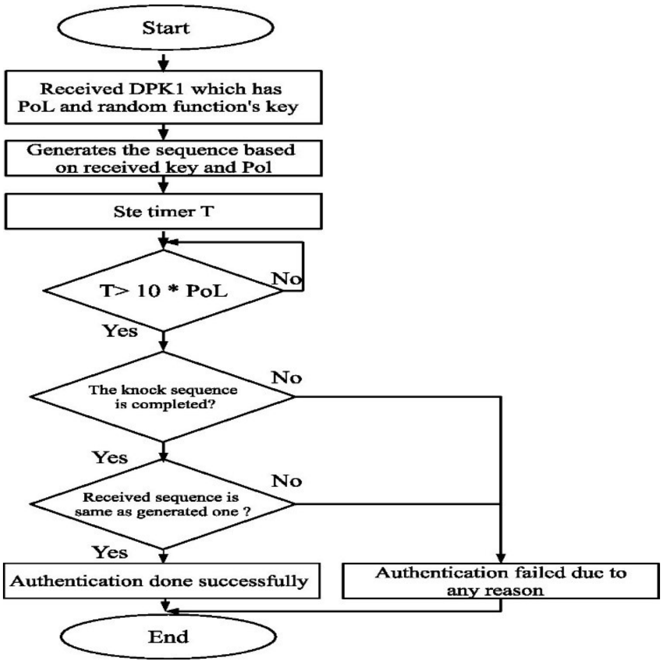
Start
Received DPK1 which has PoL and random function's key
Generates the sequence based on received key and Pol
Ste timer T
T> 10 * PoL
No
Yes
The knock sequence is completed?
No
Yes
Received sequence is same as generated one ?
No
Yes
Authentication done successfully
Authentication failed due to any reason
End

**Fig. 5:** Dynamic Length Dynamic Knock Sequence (DLDKS) Port Knocking Authentication on Server

Performance is optimized and the security level is improved by employing the dynamic length sequence for port knock authentication. Using the dynamic length port knocking, it is hard to find the correct knock sequence which can easily be determined in short length knocking. Additionally, performance is optimized due to reduction of delay, which occurs in long knock sequences. As an illustrative instance, previous authentication process [1] took 100 seconds in port knocking, which is completed with 10 knocked packets (10 packets × 10 Sec) in each attempt, whereas by employing the proposed dynamic length port knocking, the time of authentication is variable in each request and the maximum time is determined by the maximum length which is existed in the PoL.

Analysis shows that dynamic length port knocking addresses the issues of both security and performance. In [35], security of basic port knocking is evaluated based on the probability of hacking the knock-sequence through two parameters which are the number of users who request for authentication simultaneously, *U*, and the length of knock-sequence, $L_{ks}$. In the proposed method, besides the mentioned parameters, PoL is involved to consider that port knockers utilize various lengths in the dynamic length port knocking method. Thus, the way of computing the probability of hacking sequence is different from basic port knocking. The basic port knocking security is evaluated based on the number of users using the same sequence, *U*, length of the knock-sequence. $L_{ks}$, and probability of use the correct order of knock-sequence, $L_{ks}!$, as shown in the following equation.

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

Based on the Equation (1), an increase in *U* leads to a higher probability of the knock-sequence being hacked. In the meantime, an increase in $L_{ks}$ causes a lower amount for the probability as it tends to be more difficult for the hacker to acquire the proper port sequence. Dynamic length port knocking is expected to consider both best and worst cases based on the utilized length of knock-sequence. In terms of security, the best case refers to the condition in which all port knockers use the longest sequence. Also, the worst case occurs when the port knockers utilize the shortest length among the sequences which are existing in the PoL. However, the mentioned scenarios occur rarely, due to the use of random generation function for selecting the length. Therefore, we assume that the number of users which apply for authentication with *Lksi*-knocked sequence are equal to each other. Therefore, probability of hacking sequence in the dynamic length port knocking is computed by the following equation.

$$P(HS)_{dynamic\ length} = \prod_{i=1}^{number\ of\ POL'sentries} (P(HS)_{L_{KS_i}} \qquad (2)$$

According the aforementioned assumption, P(HS) is computed as follows.

$$P(HS) = \frac{\frac{U}{PoL}}{L_{KS}! \times 2^{L_{KS}}} \qquad (3)$$

The following expression represents the $P(HS)_{\text{dynamic length}}$ for 10 concurrent authentications:

$$P(HS)_{dynamic\ length} \approx \frac{\frac{10}{3}}{3! \times 2^{3}} \times \frac{\frac{10}{3}}{5! \times 2^{5}} \times \frac{\frac{10}{3}}{7! \times 2^{7}} \qquad (4)$$

The probability of hacking the sequence in the dynamic length in this method is near to "0" when the round function was used. In addition to the security, performance is an important factor that impact on both client and server. We evaluate the performance of the dynamic length port knocking authentication through using the following two parameters*: (1) Time of port knocking authentication (ToPKA):* which starts counting time when a port knocker sends the authentication request to end of the authentication process; and *(2) Load of buffering (LoB):* shows the load which is imposed to the cloud gateway in monitoring the ports and creating dynamic address lists to buffer the packets of a port knocker. LoB is estimated by the memory space which is occupied to buffer the received knock-packets during the authentication process. In the dynamic port knocking method, a PoL is introduced that keeps various lengths of knock sequence. The PoL is used in the port knocker script in the implementation section. Afterwards, LoB is computed based on the number of members in the PoL.

Assume that the number of members in PoL is equal to "m", Therefore, the "input" rules which are defined for the dynamic port knocking is calculated by the following equation.

$$Number\ of\ input\ rules = \sum_{i=1}^{M} DL_{KS_i} \qquad (5)$$

For each input rule, a dynamic address list is created. Indeed, in a single authentication, the LoB is equal to the L × 32 bits. Generally, for a single attempt in dynamic port knocking, the minimum space that is required is equal to $L_{min} \times 32$ bits and the maximum required space is $L_{max} \times 32$ bit. When multiple users request for authentication at the same time, the minimum and maximum spaces which are occupied are equal to $U \times L_{min} \times 32$ bits and $U \times L_{max} \times 32$ bits. The experimental results which are gathered from 10,100 and 500 parallel authentication attempts are discussed in the following section. We deployed the proposed authentication method on the MikroTik RouterOs as a cloud gateway and used Autoit to program the mobile device as a port knocker.

## 4.0 RESULTS AND DISCUSSION

Fig. 6 shows the probability of hacking sequence in basic port knocking and dynamic port knocking. In this figure, three graphs belong to the basic port knocking, which illustrate the P (HS) of three different lengths that are common with lengths which are used in the dynamic port knocking. The fourth graph reveals the P (HS) of the dynamic port knocking method. The horizontal axis depicts the number of users who send the authentication request to the server at the same time, whereas the vertical axis determines the probability of hacking the knock sequence.

The figure shows that 3- knocked basic port knocking (short length basic port knocking) is unreliable and hackers can find the sequence easily especially with the growing of the number of users. Indeed, short length basic can only be used in cases that security is prioritized. Other three graphs determine that 7-knocked and dynamic port knocking is strong enough to be trusted for the reason that the probability of being hacked is almost near to zero, regardless the number of concurrent authentication's requests. While the 5-knock basic port knocking in case of numerous simultaneous authentications faces security issues because of it's potential to be hacked. Hence, the long length of basic port knocking is secure enough for the reason that probability of hacking does not exceed the 0.00001%, even if the number of parallel port knockers increases drastically. Conversely, dynamic length port knocking is preferable, for the reason that it employs different knock sequence in each attempt, which reduces the possibility of hacking to zero percent.

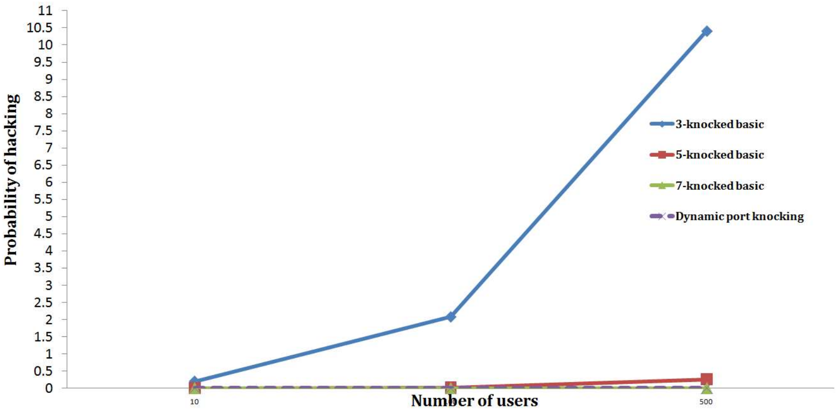

**Fig. 6:** Probability of Hacking Sequence in Basic Port Knocking and Dynamic Port Knocking

Time of authentication is an important parameter especially in high traffic network. It shows the time interval from starting authentication process by the user to achieving the result of authentication by the server. The obtained results from 10 concurrent authentication indicate minimum ToPKA is 3001.5 milliseconds (ms) whereas, the maximum is equal to 7038.3 ms. This amount is between 3003 ms and 7138 ms in case of 100 parallel authentication processes. Results indicate that ToPKA varies from 3004.9 ms to 116301 ms in case of 500 concurrent requests. The time values which are between 3000 ms to 5000 ms belong to the 3-knocked dynamic port knocking. Also, values which are between 5000 ms and 8000 ms indicate that the length which is selected randomly is equal to 5. The 7-knocked dynamic port knocking takes around 7000 ms to 12000 ms in the entire authentication process. The interval time which is spent by port knocker to perform a dynamic port knocking with 5 length knock-sequence overlaps with the 7 length knock-sequence dynamic port knocking. It happens for the reason that various number of users apply for authenticating at the same time, the authentication process enters to a queue which takes more time.

Fig. 7 shows the average time of dynamic port knocking authentication per each user. The time graph raises gradually when the number of users are increased. The average time difference for individual user between 10 and 100 simultaneous authentication is less than 0.18 ms, while the difference is about 1.5 ms when the number of users increases to 500. The increase in time interval for authentication is for the reason that requests are queued by cloud gateway which are entered to the dynamic lists of buffers for each port.

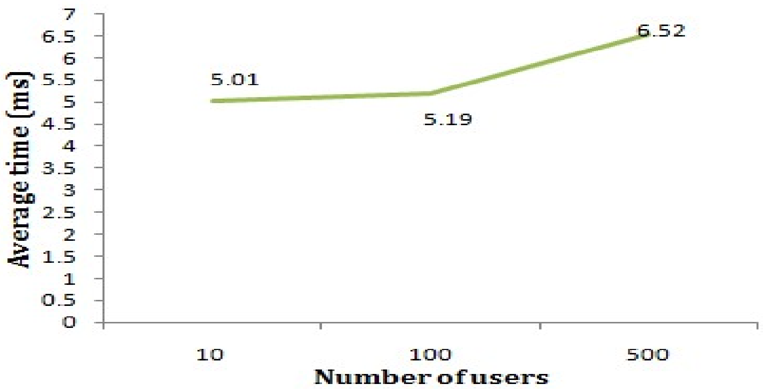

**Fig. 7:** Average Time of Dynamic Port Knocking Authentication for Each Port Knocker

To compare the ToPKA among basic and dynamic port knocking, Fig. 8 shows the time for 10,100 and 500 concurrent users. This figure reveals for each method, regardless the length of knock sequence, time of authentication is increased with the increment number of users. Moreover, it shows dynamic port knocking requires more time in contrast with 3-knocked basic port knocking, while consumes less time to serve the authentication's request in comparison with the 7-length basic port knocking. Comparing dynamic and 5-knocked basic port knocking makes clear that although, they have not considerable difference in terms of consumed time, dynamic port knocking in most of the case needs less time. Therefore, using dynamic port knocking method has priority than using basic port knocking when both time and security are important.

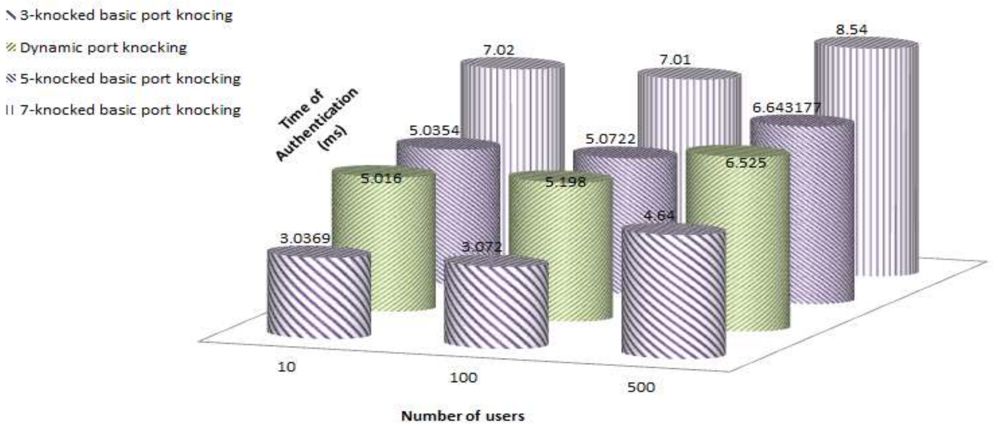

**Fig. 8:** Comparison of ToPKN of dynamic port knocking and Basic port knocking

To make an accurate decision about which of these two methods are more cost-effective, the LoB of dynamic port knocking are compared with the basic port knocking. Table 1 contains the buffering space that needs to be allocated for the dynamic port knocking when the following assumption is considered.

**Assumption 5.1:** The length of knock sequence is selected in order from the PoL. For instance, if PoL has three members *$L_1$, $L_2$ and $L_3$*, the length for first port knocker is *$L_1$*, while second and third port knockers use *$L_2$* and *$L_3$*, respectively. In the aforementioned scenario, if 10 users apply for dynamic port knocking, four of them use *$L_1$*, while each of the *$L_2$* and *$L_3$* is used by 3 port knockers.

**Table 1**. Load of buffering of the CGW in dynamic port knocking

| Number of users | Load of Buffering |
|---|---|
| 10 | 0.187 |
| 100 | 1.94531 |
| 500 | 9.7578 |

The table shows the load of buffering has dramatically upward trend when the number of users are increased.

Fig. 9 shows the allocated memory space to basic and dynamic port knocking for the increment number of users from 10 to 500. The graph provides the memory allocation for the same lengths of basic port knocking since the dynamic port knocking uses 3, 5 and 7 lengths of knock sequence. Although basic and dynamic length port knocking methods use only the cloud gateway for port knocking authentication, the amount of memory space which is allocated to buffering is different for the reason of the dynamic length which is applied in the proposed method. The occupied space for buffering extends with the number of port knocker's enlargement in both basic and dynamic port knocking. In addition, the bar charts show that the dynamic port knocking needs more memory space for buffering in comparison with the 3-knocked basic port knocking. This amount is similar for dynamic and 5-knocked port knocking while 7-knocked basic port knocking takes up more space in contrast with dynamic length port knocking especially when 500 users apply for authentication.

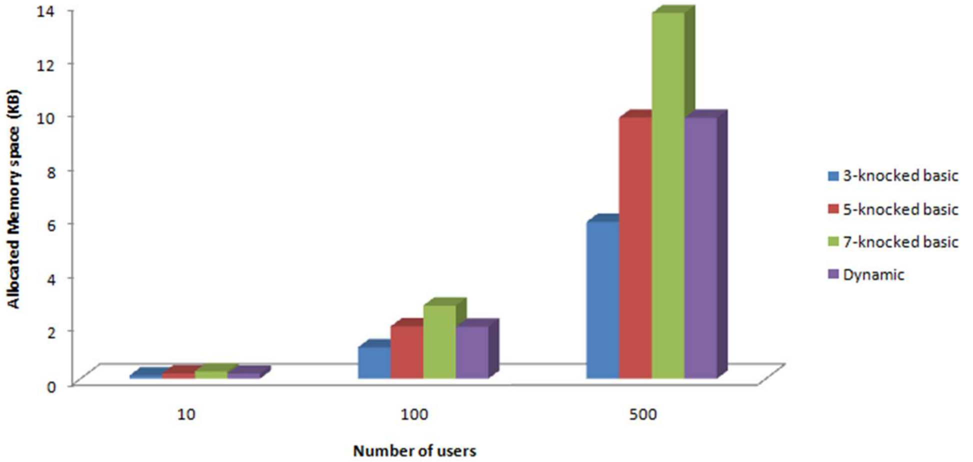

**Fig. 9:** Allocated memory space to port knocking process for basic and dynamic port knocking

The dynamic length port knocking authentication method is employed as 3, 5 and 7. Furthermore, upon the mentioned assumption, the number of users who use each of these lengths are similar. Therefore, when 100 users apply the 3-knocked basic port knocking for each of them, three buffers are occupied and the total number of buffers is 300. However, in the dynamic port knocking, 37 users work with 3 and 5-knocked sequence, and 36 users apply 7-knocked basic port knocking. Hence, 548 buffers are involved in the port knocking authentication which are created dynamically and released after finishing the process. Additionally, the same reason explains the benefit of using dynamic port knocking rather than the 7-knocked basic port knocking where 700 buffers are taken up. However, in the case of 5-knocked, basic sequence and dynamic port knocking, an exception is happened in the proposed case study that causes the memory allocation for these two methods is quite similar to each other. This similarity relates to the various lengths which are used in the dynamic port knocking. Dynamic port knocking method employs three sequential odd numbers that have two-unit differences. Hence, users who use 5-knocked basic port knocking are similar in both dynamic and basic one and the gap between numbers of buffers, which are allocated to 3-knocked basic port knocking is folded by the required buffers of people who use 7-knocked basic port knocking.

Therefore, it is concluded that the dynamic port knocking method needs more space for buffering in comparison with the basic port knocking, which use the minimum length between the PoL of dynamic port knocking. In addition, it needs less space when the basic port knocking is performed by using the longest length for all users. Moreover, the difference between occupied space of dynamic and basic port knocking, which use any of other lengths, which is existing in the PoL, depends on the lengths that dynamic port knocking is using besides the policy that indicates how many of the port knockers use each length.

## 5.0 CONCLUSION

Dynamic length port knocking is proposed as a lightweight authentication technique for MCC. This technique improves the basic port knocking by using various lengths of knock-sequence in each authentication attempt. This enhancement leads to have high security in contrast with the basic port knocking and reduces the probability of hacking near to zero especially, in case of a large number of concurrent authentication. In terms of performance, analysis of the results shows that the proposed technique improves the consumed time up to 23%. Moreover, the load of buffering in dynamic port knocking is less than the load of 7-knock basic port knocking and is roughly similar to the load of 5-knock basic port knocking. Results show the proposed technique enhances buffer management up to 28% by reducing the imposed load. However, buffering load is centralized on the cloud gateway which degrades performance in case of client and knock sequence's enlargement. Thus, combining virtualization concept of MCC can be counted as a solution in the future researches.

## REFERENCES

[1] M. Shiraz, M. Wahiduzaman, and A. Gani, “A Study on Anatomy of Smartphone” *Computer Communication & Collaboration*, vol. 1, no. 1, pp. 24-31, 2013.

[2] D. C. M. Rothman, A. Moloney, “Authentication in the Modern World 4 Best Practices for Adapting to the Shifting Paradigms in IT,” tech. rep., SafeNet, 2011.

[3] J. Bardin, J. Callas, S. Chaput, P. Fusco, F. Gilbert, C. Hoff & B. O’Higgins Security guidance for critical areas of focus in cloud computing. Cloud Security Alliance, 0-176. 2009

[4] H. Liang, D. Huang, L. Cai, X. Shen, and D. Peng, “Resource allocation for security services in mobile cloud computing,” *Computer Communications Workshops (IN-FOCOM WKSHPS)*, pp. 191-195, April 2011.

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

[6] O. Nurika, M. A. H. B. Aminz, A. S. B. A.Rahman, and M. N. B. Zakaria, “Review of various firewall deployment models,” vol. 2, pp. 825-829, 2012.

[7] B. Lakshmiraghavan, “two-factor authentication,” in *Pro ASP. NET Web API Security*, pp.319-343, Springer, 2013.

[8] D. Co_n, “Two-factor authentication,”in *Expert Oracle and Java Security*, pp. 177-208, Springer, 2011.

[9] F. Aloul, S. Zahidi, and W. El-Hajj, “Two factor authentication using mobile phones,” in *International Conference on Computer Systems and Applications*, *AICCS*, pp. 641-644, 2009.

[10] U. Ashraf, “Securing cloud applications with two-factor authentication,” Master's thesis, *Institute of Parallel and Distributed Systems University of Stuttgart*, 2013.

[11] R. Banyal, P. Jain, and V. Jain, “Multi-factor authentication framework for cloud computing,” in *Fifth International Conference on Computational Intelligence, Modelling and Simulation (CIMSim)*, pp. 105-110, Sept 2013.

[12] M. Mohammed and M. Elsadig, “A multi-layer of multi factors authentication model for online banking services,” *in International Conference on, Computing, Electrical and Electronics Engineering (ICCEEE)*, pp. 220-224, 2013.

[13] H. M. M. J. Nancie Gunson, Diarmid Marshall, “User perceptions of security and usability of single-factor and two-factor authentication in automated telephone banking,”*Computers & Security*, vol. 30, pp. 208-220, 2011.

[14] D. He, and D. Wang, D., Robust biometrics-based authentication scheme for multiserver environment. IEEE Systems Journal, Vol. 9, no. 3, 816-823. 2015.

[15] A. G. Andrew TeohBeng Jin, David Ngo Chek Ling, “Biohashing: two factor authentication featuring _ngerprint data and tokenised random number," *Pattern Recognition,*vol. 37, pp. 2245-2255, 2004.

[16] Q. Xiao, “Security issues in biometric authentication," in *Information Assurance Work-shop*, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC, pp. 8-13, June 2005.

[17] T.-H. Chen, H. lien Yeh, and W.-K.Shih, “An advanced ecc dynamic id-based remote mutual authentication scheme for cloud computing," in *Multimedia and Ubiquitous Engineering (MUE)*, 2011 5th FTRA International Conference on, pp. 155-159, 2011.

[18] J. Zhang and Q. Zhang, “Cooperative Network Coding-Aware Routing for Multi-Rate Wireless Networks," *IEEE INFOCOM* 2009 - The 28th Conference on Computer Communications, pp. 181-189, Apr. 2009.

[19] A. Gupta, C. Ferris, and A. Abdelnur, “Method and apparatus for authenticating users,”May 1 2001. US Patent 6,226,752.

[20] P. Harsh, F. Dudouet, J. S. Y. Cascella, Roberto G., and C. Morin, “Using open standards for interoperability - issues, solutions, and challenges facing cloud computing,"*CoRR*, 2012.

[21] Krzywinski, “Port Knocking: Network Authentication Across Closed Ports," *Sys Admin Magazine*, vol. 12, p. 6, 2003.

[22] T. Popeea, V. Olteanu, L. Gheorghe, and R. Rughinis, “Extension of a portknockingclient-server architecture with NTP synchronization," in *Roedunet International Conference (RoEduNet),* 2011 10th, pp. 1-5, IEEE, 2011.

[23] M. Pourvahab, R. EbrahimiAtani, and L. Boroumand, “SPKT: Secure Port Knock-Tunneling, an enhanced port security authentication mechanism," in *Computers & Informatics (ISCI)*, 2012 IEEE Symposium on, pp. 145-149, 2012.

[24] 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.

[25] D. Isabel, Port knocking: Beyond the basics. GIAC Security Essentials Certification (GSEC). SANS Institute. (2005).

[26] C. Chew Keong TAN, “Remote server management using dynamic port knocking and forwarding," Special Interest Group in *Security and Information Integrity*, 2004.

[27] 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)*, pp. 159-163, IEEE, Apr. 2011.

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

[29] H. Al-Bahadili and A. Hadi, “Network Security Using Hybrid Port Knocking," *IJCSNS*,vol. 10, no. 8, p. 8, 2010.

[30] Narayanan, A. A critique of port knocking. Newsforge, August, 2004

[31] S. Jeanquier, An Analysis of Port Knocking and Single Packet Authorization MSc Thesis. PhD thesis, 2006.

[32] L. Boroumand, M. Shiraz, A. Gani, and R. H. Khokhar, “Impact of Port Knocking Authentication on Security and Performance: A Mobile Cloud Computing Perspective, in KSII Cloud Computing Symposium, The 5th *International Conference on Internet (ICONI)*, 2013.

[33] M. Rajagopalan, M. Hiltunen, T. Jim, and R. Schlichting, “Authenticated system calls," in *Dependable Systems and Networks*, 2005. DSN 2005. Proceedings in International Conference on, pp. 358-367, 2005.

[34] S. Jeanquie, Ananalysis of port knocking and single packet authorization. PhD thesis, London, 2006.

[35] C. Hou, H. Jiang, W. Rui, and L. Liu, “Improvement of NTP synchronization accuracy for switch-oriented power monitoring networks," *Dianli Zidonghua Shebei/Electric Power Automation Equipment*, vol. 33, no. 1, pp. 148-152, 2013.

[36] P. Pawar, B. van Beijnum, K. Wac, H. Hermens, and D. Konstantas, “Towards location based QoS-aware network selection mechanism for the nomadic mobile services," 2009.

[37] H. Zhu, X. Lu, Q. Tang, X. Zhang, and C. Zhao, “A new chaos-based image encryption scheme using quadratic residue," pp. 1800-1804, 2012.

[38] W. Thomas , Cusick, D. Cunsheng, and R. Ari, eds. Chapter 2 Stream ciphers," in North-Holland Mathematical Library vol. 66, pp. 11-43, Elsevier, 2004.

[39] K. J. Kulikowski, M. G. Karpovsky, and A. Taubin, “Robust codes and robust, fault-tolerant architectures of the Advanced Encryption Standard," *Journal of Systems Architecture*, vol. 53, no. 23, pp. 139-149, 2007.

[40] R. C. W. Phan, “Impossible di_erential cryptanalysis of 7-round Advanced Encryption Standard (AES)," *Information Processing Letters*, vol. 91, no. 1, pp. 33-38, 2004.

[41] S. Sanchez, R. Criado, and C. Vega, “A generator of pseudo-random numbers sequence with a very long period," *Mathematical and Computer Modelling*, vol. 42, no. 78, pp. 809-816, 2005.

[42] R. Smits, D. Jain, S. Pidcock, I. Goldberg, & U. Hengartner. Bridge SPA: Improving Tor bridges with single packet authorization. In Proceedings of the 10th annual ACM workshop on Privacy in the electronic society (pp. 93-102). ACM, October, 2011

[43] H. Anne, L. Mark, S. Abhishek, M. L. A. S. Chris CantrellA2 - Anne Henmi, and C. Chris, eds. Chapter 6 - Deciding on a VPN," in Firewall Policies and VPN Con_gurations , pp. 267-304, Burlington: Syngress, 2006.

[44] C. Xenakis and L. Merakos, “IPsec-based end-to-end VPN deployment over UMTS," *Computer Communications,* vol. 27, no. 17, pp. 1693-1708, 2004.

[45] Z. A. Khan, N. Javaid, M. H. Arshad, A. Bibi, and B. Qasim, “Performance Evaluation of Widely Used Portknoking Algorithms," in *High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS),* pp. 903-907, 2012.

[46] S. Prowell, R. Kraus, and M. Borkin, “Seven Deadliest Network Attacks: Chapter 1 Denial of Service," pp. 1-21, Syngress, 2010.

[47] M. Doyle, Implementing a Port Knocking System in C. An Honors Thesis submitted in partial fulfillment of the requirements for Honors Studies in Physics, J. William Fulbright College of Arts and Sciences, The University of Arkansas. (2004).

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

[49] W. Du, & L. Wang, Context-aware application programming for mobile devices. In Proceedings of the 2008 C 3 S 2 E conference (pp. 215-227). ACM. 2008.

[50] M. Krzywinski, Port knocking from the inside out. SysAdmin Magazine, 12(6), 12-17. 2003.

[51] A. Manzanares, J. M_arquez, J. Estevez-Tapiador, and J. Castro, “Attacks on port knocking authentication mechanism," *Computational Science and Its Applications ICCSA*2005, pp. 1292-1300, 2005.

[52] R. Castro, J. Vega, T. Fredian, K. Purahoo, A. Pereira, and A. Portas, “Securing MD-Splus in a multi-organisation environment,” *Fusion Engineering and Design*, vol. 85 no. 34, pp. 614-617, 2010.

[53] S. Khan, M. Shiraz, L. Boroumand, A. Gani and MK. Khan “Towards Port-Knocking Authentication Methods for Mobile Cloud Computing”, Journal of Network and Computer Applications, vol. 2017, no. 97, pp.66-78, 2017