Implementation of Modified Hybrid Port Knocking (MHPK) with Strong Authentication

Priyanka Sahu
M.Tech. Scholar
CSE Dept.
CIIT INDORE

Megha Singh
Assistant Professor
CSE Dept.
CIIT INDORE

Deepak Kulhare
Associate Professor
CSE Dept.
CIIT INDORE

ABSTRACT
Port Knocking is an important concept to secure services provided by the servers. By a predefined port knocking sequence server identify whether the request is a legitimate request for a service**.** This paper presents an analyzing the network security concept of Port Knocking and assesses their suitability as firewall authentication mechanisms for opening network ports or performing certain actions on servers. This paper are developing and evaluating the performance of a new proposed modified hybrid port knocking (MHPK) technique with proposed encryption/decryption technique. Prime concerned of the proposed work is to prevent different – different types of port attack and fulfill the entire security requirement for network. Proposed technique is the combination of four concepts, these are port knocking (PK), proposed Symmetric key encryption/decryption, steganography and mutual authentication. Basically it is the enhanced modification of hybrid port knocking therefore; it is referred to as the modified hybrid port-knocking (MHPK) technique. The improvements of this development over existing port knocking mitigate some threats, but various concerns still exist. Development aimed at the enterprise quality will need to address additional needs. In conclusion, port knocking deserves future consideration and can be a valuable layer in defense-in-depth. The performance of the proposed technique is calculated by measuring some parameter like security, and average authentication time, which is showing the superiority of the proposed technique as compare to existing technique.

Keywords
Port Knocking, Authentication, Security, Steganography, cryptography, firewall.

1. INTRODUCTION
Over the years, we have seen many improvements to protocols and even the network stack itself with the aim of making them more resistant to attack. One primary concern is the notion of authorization and authentication, to which there are many solutions, a popular one being the requirement for users to enter a username and password before a service can be used. Security has also been applied at different places in the TCP/IP protocol stack including the invention of IPsec that aims to protect packets at the IP Layer, and SSL/TLS which protects packets at the Transport Layer. Confidentiality, Integrity, and Authentication have become the primary concerns for protocols carrying sensitive traffic. Without these precautions, the information sent over the net-work could be vulnerable to unauthorized disclosure, modification, or we would simply be unsure as to who actually sent that information. Port knocking is a great method to allow remote access without the security risks associated with keeping ports constantly open on the internet side of your networked host. This was intended to be used as an extra line of defense and is not an excuse for weak passwords. With port knocking, the firewalled server machine runs the knock-daemon. This daemon listens for a specific sequence of tcp or udp "knocks" and associates them with an action. The knock daemon listens at a very low level in the TCP/IP stack and does not require any open ports to operate. The program is also completely stealth from outside the system. This stealth also adds a layer of security because attackers cannot see any signs of the knock-daemon running, nor will they be likely to attempt random knocks to try to gain access. On the client side, we have a program called knock which then issues these "knocks" to the host machine in order to perform a desired action, usually to open a port for remote access. This usually restricts the port to only the IP that issues the knock which also increases security over a simple open port [1]. Port knocking is a suitable form of hardening hosts that house users who require continual access to services and data from any location and that are not running public services. Port knocking is used to keep all ports closed to public traffic while flexibly opening and closing ports to traffic from users who have authenticated them with a knock sequence. Port knocking cannot be used to protect public services - such protection cannot be effective if the knock sequence or a method to generate is made public. Port knocking can be used whenever there is a need to transfer information across closed ports. The port knock daemon can be implemented to respond in any suitable way to an authentic port knock. The knock may be used to communicate the knock information silently and/or to trigger an action. This is a form of IP over closed ports. The simplest implementation of port knocking uses a log file to interface with the firewall software. This simple approach makes port knocking highly accessible for home users who would like to harden their NIX systems. One of the strong advantages of port knocking is that the protected services do not require any modification. Cryptography has the ability to provide a number of services which aid us in protecting our information in various ways as it is sent across networks or stored on physical media. The use of encryption has allowed us to protect confidential information and prevent it being disclosed to unauthorized parties. Similarly, data integrity ensures that our information is not modified in transit, and we can trust that the information received is as intended. The slightly more contemporary field of public key cryptography has the added ability of providing non-repudiation whereby it can be proven that an individual did indeed send a particular piece of information. The evolution of proposed encryption/decryption algorithm helps in MPHK

increasing the security level of the proposed technique.

2. EXITSTING PORT KNOCKING TECHNIQUE

The HPK technique is designed to work in two different modes, interactive mode and the non-interactive mode. In any of the above modes, the HPK client does not send TCP SYN packets to initialize the service on the HPK server as in TPK techniques; instead it sends TCP packets with sophisticated payloads. The payloads send within the TCP packets represent the content of the service or task that needs to be performed on the accessed network or any of its servers. [2]

2.1 Hybrid port knocking (HPK)

The HPK technique consists of seven main steps. In what follows, is the description of the seven steps [2].

1. Traffic monitoring

PK server is installed behind the network firewall, monitoring and checking traffic arrived to firewall (gateway).

2. Traffic capturing and analyzing

The PK server captures only the traffic holding a payload (image) for further processing.

3. Image processing

In this step, the PK server extracts the payload (image) from the received packet. The payload is supposed to hide some information using Steganography that can be used to prove the knockers identity and request. If the payload, contains encrypted information, which is demand to encryption/decryption algorithm to access intended information.

4. Client authenticating

After the PK server makes sure that the payload was carrying an encrypted request, it needs to make sure that it is communicating with the correct client, so it takes a random number and encrypts it using the clients GnuPG public key and sends it as a payload to the client.

5. Server authentications

The client now receives the packet carrying the encrypted payload, extracts it and decrypts it using the servers GnuPG public key. Then the client sends the random number as a payload back to the PK server to ensure its identity.

6. Proving the identity of the client

The PK server is still in the monitoring/sniffing state and receives the reply from the client to its random number check. The server extracts the payload and checks if the received message holds the same number as the one randomly generated and sent to the client.

7. Port closing

Finally, in this step, after the task is completed, either the client informs the PK server to close the port, or the PK server decides to close the opened port.

3. PROPOSED TECHNIQUE

The main goal of proposed research is to design Port Knocking (PK) model and develop Port knocking system with high security. The proposed research will analyze all types of port attacks and will find those reasons which are the cause of big problem in the network. Proposed model will be the combination of two different techniques like port-knocking (PK), and Cryptography. Proposed concept will be highly secured and efficient so this proposed technique will known as "Modified Hybrid Port-Knocking (MHPK)" technique. Port knocking system use a cryptographically-secure challenge response authentication system that accounts for out-of order packet delivery and partially addresses the complications caused by NATs. [3]

The MHPK technique consists of nine main steps in what follows is the description of the nine steps.

1. Traffic monitoring

In this step, a PK server is installed behind the network firewall, as shown in Fig. 1, monitoring and checking traffic arrived to firewall (gateway).

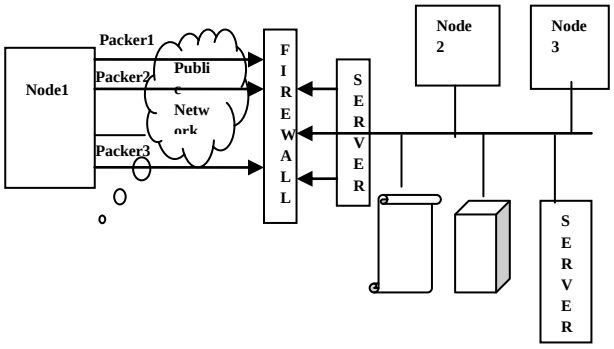


Fig.1 Traffic monitoring

2. Traffic capturing and analyzing

In this step, the PK server captures only the traffic holding a payload (image) for further processing, as shown in Fig. 3. In this figure, for example, only Traffic #3 is captured for further processing because it contains an image.

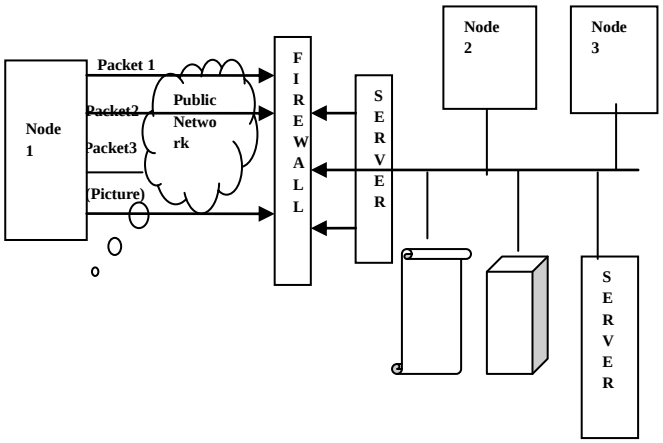


Fig.2 Traffic capturing

3. Image processes and Cryptography Functioning

In this step, the PK server extracts the payload (image) from the received packet. The payload is supposed to hide some information using Steganography that can be used to prove the knockers identity and request. If the payload, contains encrypted information, which is demand to encryption/decryption algorithm to access intended information, see figure (3).

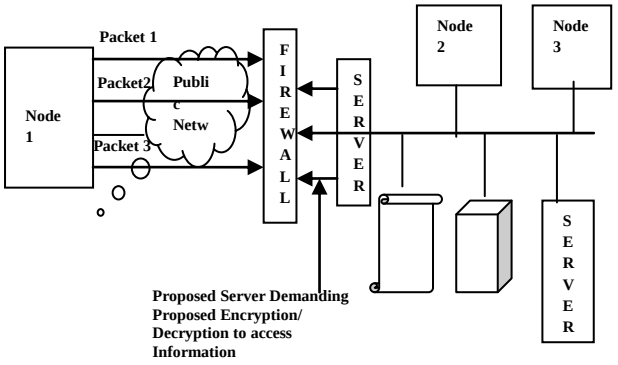


Fig.3 PK server demanding Encryption/Decryption Algorithm to Access Information

4. Client authenticating

After the PK server makes sure that the payload was carrying an encrypted request, it needs to make sure that it is communicating with the correct client, so it takes a random number and encrypts it using the clients GnuPG public key and sends it as a payload to the client.

5. Server authentications

The client now receives the packet carrying the encrypted payload, extracts it and decrypts it using the servers GnuPG public key. Then the client sends the random number as a payload back to the PK server to ensure its identity.

6. Proving the identity of the client

The PK server is still in the monitoring/sniffing state and receives the reply from the client to its random number check. The server extracts the payload and checks if the received message holds the same number as the one randomly generated and sent to the client.

7. Key Exchanging

After checking authenticity between client and server key exchange step will follow. In this step Client and Server exchanged symmetric key through GNUPG public key technique.

8. Proposed Encryption/Decryption

After key exchanging between client and server. Server called encryption/decryption process to access proper encrypted information from payload of the image. If the message is identified then the PK server executes the opening/closing of the requested port on the firewall, or executes the remote command based on the client's request. If the payload, contains intended information, which is either to demand the firewall to open/close a port for the client as shown in Fig. 4, or execute a command remotely on the appropriate server as shown in Fig. 5. Otherwise, if the result of the image processing fails to reveal valid authentication parameters, the PK server blocks the IP address of the source that sent the knocks and the payload (image). No port open/close or remote command execution is done in this step, only ensuring that the received payload holds a request which needs further authentication.

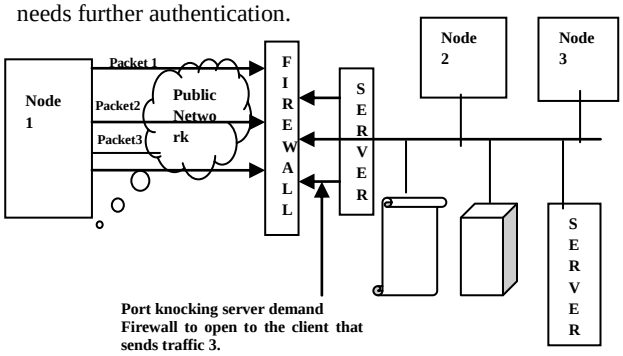


Fig.4 PK server demanding firewall to open/close port

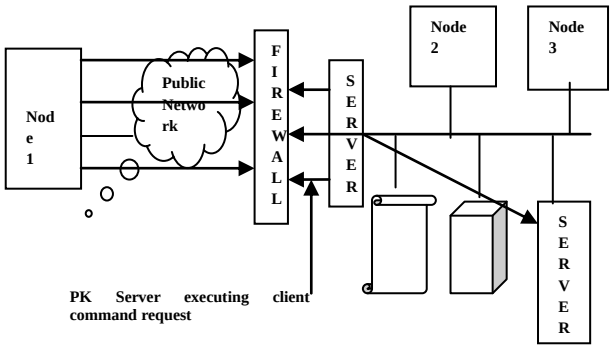


Fig.5 PK server executing clients command request

9. Port closing

Finally, in this step, after the task is completed, either the client informs the PK server to close the port, or the PK server decides to close the opened port after specified silent period on that open port as shown in fig. 6.

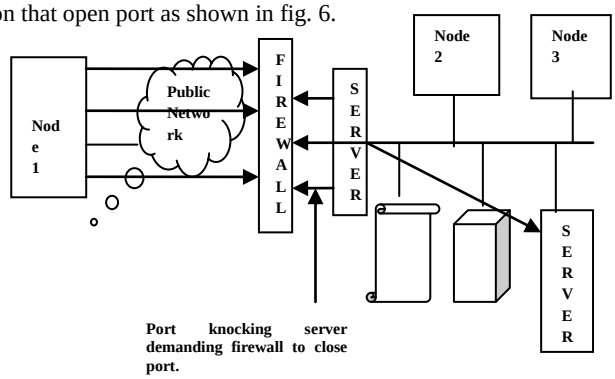


Fig.6 Port closing

In any of these two cases, the PK server demands firewall to close the open port. In this case, if the client wants to access the system again, it needs to initiate new access or authentication request, i.e., start from phase #1.

4. PROPOSED ENCRYPTION/ DECRYPTION TECHNIQUE

Proposed encryption decryption technique is based on the block cipher code (CBC) mode. Proposed encryption algorithm is using logical operations like XOR, shift (left or right circular) to mix key value and text value with each other. In this 128 bits long text can be encrypt or decrypt at a time

with 128 bits key value. Initially 160 bits are passing as a input key value, then this input key values are dividing into two part, first part of the input key is 128 bits long and this will treat as an actual key and second part of the input key is 32 bits long. Now second part again will divide into two equal parts of 16 bits each. First 16 bits will be treat as an initialization vector one (IV-1) and second part will be treat as an initialization vector two (IV-2). Figure7 and figure8 is showing architecture of the proposed encryption and decryption.

4.1 Architecture of proposed encryption

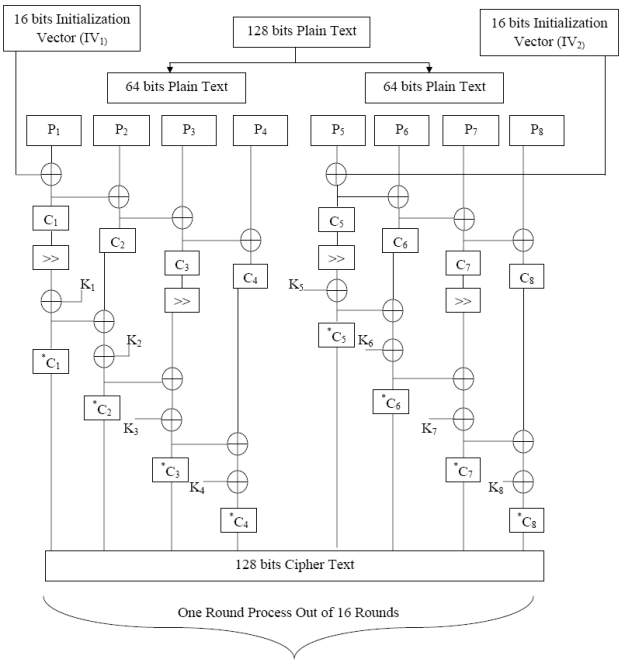


Fig. 7 Architecture of Proposed Encryption

4.1.1 Proposed Encryption Algorithm Step

i) Select plain text of 128 bits to be encrypted.
ii) Divide 128 bits plain text into two (Left, Right) blocks of 64 bits each.
iii) Now again divide these blocks (left and right) into 8 sub blocks of 16 bits each. 4 sub-blocks (P_1, P_2, P_3, P_4,) for left and 4 sub-blocks (P_5, P_6, P_7, P_8,) and for the right.
iv) Select 160 bits randomly as a key value in which first 32 bits will become initialization vectors (IV_1, IV_2) of 16 bits each. Remaining 128 bits key value will be divided into 8 sub-key values (K_1 to K_8) of 16 bits each.
v) Perform XOR operation between initialization vector (IV_1) and P1. Output of this will become C1.
vi) Perform XOR between C_1 and P_2. Output of this will become C_2.
vii) Apply 2 bits right circular shift on C_1 and perform XOR with K_1. Output of this will become $^*C_1$.
viii) Perform XOR between C_2 and $^*C_1$ and resultant will XOR with K_2. Output of this will become $^*C_2$.
ix) Perform XOR between C_2 and P_3. Output of this will become C_3.
x) Apply 2 bits right circular shift on C_3 and perform XOR with $^*C_2$. Resultant of this wills XOR with K_3. Output of this will become $^*C_3$.
xi) Perform XOR between C_3 and C_4. Output of this will become C_4.
xii) Perform XOR between C_4 and $^*C_3$. Resultant of this wills XOR with K_4. Output of this will become $^*C_4$.
xiii) Repeat process 5 to 12 for right sub block of the plain text with initialization vector (IV_2), K_5 to K_8 in place of K_1 to K_4 and P_5 to P_8 in place of P_1 to P_4.
xiv) Repeat Process 1 to 13 till 16 round.
xv) Now Combine all sub block into single cipher block.
xvi) Exit.

4.2 Architecture of proposed decryption:

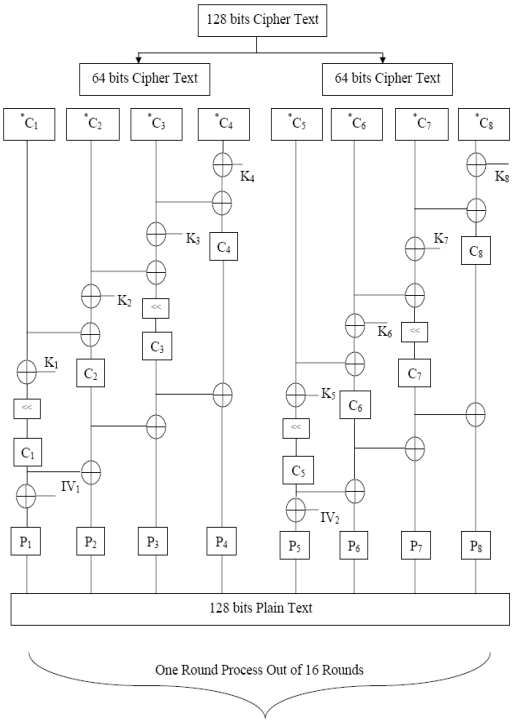


Fig. 8 Architecture of Proposed Decryption

i) Select Cipher text of 128 bits to be decrypted.
ii) Divide 128 bits Cipher text into two (Left, Right) blocks of 64 bits each.
iii) Now again divide these blocks (left and right) into 8 sub blocks of 16 bits each. 4 sub-blocks ($^*C_1$, $^*C_2$, $^*C_3$, $^*C_4$,) for left and 4 sub-blocks ($^*C_5$, $^*C_6$, $^*C_7$, $^*C_8$,) and for the right.
iv) Select 160 bits randomly as a key value in which first 32 bits will become initialization vectors (IV_1, IV_2) of 16 bits each. Remaining 128 bits key value will be divided into 8 sub-key values (K_1 to K_8) of 16 bits each.
v) Perform XOR between K_1 and $^*C_1$. And apply bits right circular shift in reverse out put this will become C_1.
vi) Perform XOR between C_1 and Initialization Vector (IV_2). Output of this will become P_1.
vii) Perform XOR between $^*C_2$ and K_2, resultant value will XOR with $^*C_1$. Output of this will become C_2.
viii) Perform XOR between C_2 and C_1. Output of this will become P_2.
ix) Perform XOR between $^*C_3$ and K_3, resultant value will XOR with $^*C_2$ and apply 2 bits right circular shift in reverse. Output of this will become C_3.
x) Perform XOR between C_3 and C_2. Output of this will become P_3.

xi) Perform XOR between $*C_4$ and K_4, resultant value will XOR with $^*C_3$. Outputs of this will C_4.
xii) Perform XOR between C_4 and C_3. Output of this will become P_4.
xiii) Repeat process 5 to 12 for right sub block of the cipher text with initialization vector (IV_2), K_5 to K_8 in place of K_1 to K_4 and ($^*C_5$, $*C_6$, $^*C_7$, $^*C_8$) in place of ($^*C_1$, $*C_2$, $^*C_3$, $^*C_4$).
xiv) Repeat Process 1 to 13 till 16 round.
xv) Exit.

There is a big difference between existing and proposed port knocking technique is the encryption/decryption algorithm used by proposed technique to provide strong authorization. Proposed modified hybrid port knocking technique is just like existing port knocking technique but number of steps has shuffled up and down and added two more steps. These two steps are key exchanging and proposed encryption/decryption which is described above. Due to this encryption/decryption step proposed port knocking technique produced more good results.
The MHPK technique is found has a built-in detection capability that can be adjusted to countermeasure after a specific number of failure attempts. For brute force attack, proposed encryption and decryption will required 2^{128} time to crack actual key which is impossible

5. COMPARISON OF BLOCK DIAGRAM OF MHPK AND HPK

Proposed encryption and decryption technique is creating differences between proposed modified port knocking (MHPK) system and other existing port knocking system.
There are various parameters to evaluate existing system and proposed system but we have select three parameter which is following

- Security
- Portability
- Symmetric Key Concept

Each parameters result is showing in the table which is following.

Security:

Existing System	Proposed System
Security	
Low	High

Table 1: Security Comparison

Portability:

Existing System	Proposed System
Portability	
No	Yes

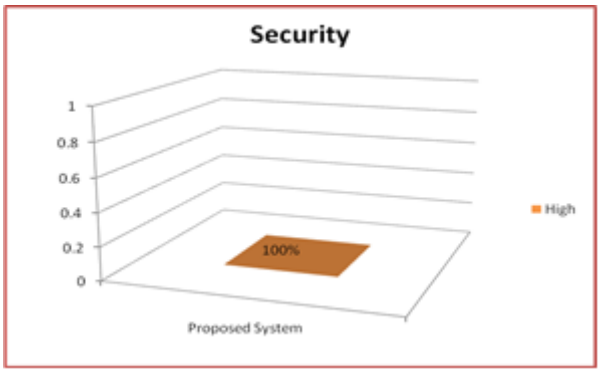
Table 2: Portability Comparison

Symmetric Key Concept:

Existing System	Proposed System
Symmetric Key	
No	Yes

Table 3: Key Comparison

From the table 1- 3 it is very clearly that proposed system is producing good results as compare existing system. Table 1 is showing the security concept between existing and proposed system. Security of the proposed system is very high as compare existing due to symmetric key cryptography concept used. Table 2 is showing the portability concept between proposed and existing system. Proposed system is portable because it is implementation language (JDK1.7). Finally table 3 is showing symmetric key cryptography concept. Existing system does not using symmetric key concept where proposed system is using this concept. In this I am using 128 bits key value which is also causes of high security.
Graph 1 is showing the security percentage of the proposed system. Which is 100% due to adding symmetric key value which 128 bits in length and this key value is not breakable?



Graph 1: Security evolution of the Proposed System

5.1 Cryptanalysis

Even if a symmetric cipher is currently unbreakable by exploiting structural weaknesses in its algorithm, it is possible to run through the entire space of keys in what is known as a brute force attack. Since longer symmetric keys require exponentially more work to brute force search, a sufficiently long symmetric key makes this line of attack impractical. With a key of length n bits, there are 2^n possible keys. This number grows very rapidly as n increases. Moore's law suggests that computing power doubles roughly every 18 to 24 months, but even this doubling effect leaves the larger symmetric key lengths currently considered acceptable well out of reach. Security level is the relative strength of an algorithm. An algorithm with a security level of x bits is stronger than one of y bits if x > y. If an algorithm has a security level of x bits, the relative effort it would take to "beat" the algorithm is of the same magnitude of breaking a secure x-bit symmetric key algorithm (without reduction or other attacks). The 128-bit security level is for sensitive information, and the 192, 256-bit level is for information of higher importance [2]. Here proposed algorithm having 128 bits key length so there are 2^{128} possible keys. The larger

number of operation (2^{128}) required to try all possible 128-bit keys is widely considered to be out of reach for conventional digital computing techniques for the future.

6. CONCLUSION

Port knocking can be used to construct authentication systems on firewalls with the goal of only allowing authorized users access to open ports. It is based on network security mechanism. Main conclusions of this proposed work are as follows.

- The MHPK technique can be easily implemented in Java/.Net Technology.
- The MHPK technique is prevent different type of attack, because it uses cryptography to authorization and integrity, and Steganography for confidential and reduce packet capturing overhead, and mutual authentication to authenticate.
- The MHPK technique is much more secure than the traditional PK and the single packet authentication techniques.
- The communication protocol used is a simple secure encryption scheme that uses GnuPG keys with Cryptography and Steganography construction.

7. REFERENCES

[1] Di Gioia P., "Behind Closed Doors: An Evaluation of Port Knocking Authentication'. Donald Bren School of Information and Computer Sciences, University of California, Irvine 2004.

[2] Dr. Hussein Al-Bahadili and Dr. Ali H. Hadi "Network Security Using Hybrid Port Knocking" IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.8, August 2010.

[3] deGraaf R, Aycock C, and Jacobson M. , "Improved Port Knocking with Strong Authentication".[Presentation]. Department of Computer Science, University of Calgary 2005.

[4] deGraaf R, Aycock C, and Jacobson M. "Improved Port Knocking with Strong Authentication". ACSAC 2005, pp. 409418, 2005.

[5] Ali Hussein, "A Hybrid Port-Knocking Technique for Host Authentication", Ph.D. Thesis, University of Banking and Financial Sciences. IJCSNS, VOL.10 No.8, August 2010.

[6] Doyle M., "Implementing a Port Knocking System in C". Department of Physics, University of Arkansas, 2004.