An Object Oriented Approach for Port Knocking

Prof. Dr. Sudan Jha
School of Computer Engineering, Campus 15
KIIT University, Patia Bhubaneswar - 751024
Email- <jhasudan@hotmail.com> <sudan.jhafcs@kiit.ac.in>

Abstract- Broadly, port knocking is a form of host-to-host communication in which information flows across closed ports. There are various variants of the port knocking method - information may be encoded into a port sequence or a packet-payload. In general, data are transmitted to closed ports and received by a monitoring daemon which intercepts the information without sending a receipt to the sender.

In one instance, port knocking refers to a method of communication between two computers (arbitrarily named here client and server) in which information is encoded, and possibly encrypted, into a sequence of port numbers. This sequence is termed the knock. Initially, the server presents no open ports to the public and is monitoring all connection attempts. The client initiates connection attempts to the server by sending SYN packets to the ports specified in the knock. This process of knocking is what gives port knocking its name. The server offers no response to the client during the knocking phase, as it "silently" processes the port sequence. When the server decodes a valid knock it triggers a server-side process.

It is sometimes desirable to allow access to open ports on a firewall only to authorized external users and present closed ports to all others. We examine ways to construct an authentication service to achieve this goal, and then examine one such method, "port knocking", and its existing implementations, in detail. We improve upon these existing implementations by presenting a novel port knocking architecture that provides strong authentication while addressing the weaknesses of existing port knocking systems.

Keywords – Network, Port, Port Knocking, Communication, Networking, Triggering Ports

I. INTRODUCTION

Thirty-five years after the birth of the Internet, it has become well established that the Internet is a hostile place. Any host connected to the Internet needs to be secure against unauthorized intrusion and other attacks. Unfortunately, the only secure system is one that is completely inaccessible, but, to be useful, many hosts need to make services accessible to other hosts. While some services need to be accessible to anyone from any location, others should only be accessed by a limited number of people, or from a limited set of locations. The research on providing a most obvious way to limit access started in late 2004, it is to require users to authenticate themselves before granting them access. Traditionally, this is left up to the services themselves: before granting users access to anything important, they must first prove their identity, using any one of a number of methods. While this is effective, it is not a perfect solution.

1.1 Associated Technology –

To simply implement port knocking on your system we need

1. A packet-filtering firewall which is capable of logging connection attempts to closed ports, allows you to monitor the log file in real time (or can send its log file to a remote syslog server), and whose rules can be dynamically modified.
2. The Perl prototype that accepts knocks and firewall log files of any format.
3. A client and a host that are needed for the communication

1.2 Associated Technology –

1.2.1 Ports

Those unaccustomed to host-based networking sometimes have trouble coming to terms with the notion of a 'port' on a computer. In the simplest of terms a port is a virtual door (represented by a 16-bit

integer) which allows the computer to keep track of which pieces of data are destined for which application or service. Networking (transport layer) protocols such as TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) both use the concept of a port when transmitting packets to and from networked hosts. The port number (when used) is included in networking packets and is interpreted not only by sending and receiving hosts, but also by intermediate routers and firewalls. A firewall can be configured to allow or deny certain packets based on their destination port. When a service is listening for requests on a port, that port is said to be open, and clients can connect to the service. If no service is listening, then the port is considered closed. A client cannot connect to a closed port.

1.22. TCP, UDP and ICMP –

TCP, UDP and ICMP are three of the most important networking protocols used regularly in modern networks. Transmission Control Protocol (TCP) is a stateful protocol that allows the two machines to create a connection between themselves and exchange information. A connection can be defined as two machines that have mutually 'agreed' to communicate. Such a connection is established by performing the 'Three-Way Handshake'. This can be compared to making a telephone call: the initiator dials the number, the receiver picks up and says "Hello?", at which point the initiator also says "Hello!", and the conversation can begin until it is ended by either end. Most applications, such as Email, Web Browsing, and File Transfers, use TCP connections to transfer information.

The opposite of TCP is the User Datagram Protocol (UDP) which is stateless and so no formal connection is established between communicating hosts. This can be compared to a postal letter: the sender writes a letter and sends it off to its destination. The letter might arrive at its destination, or it might not, either way the sender will not receive any confirmation. A host sending UDP packets to another host will receive no acknowledgement as to whether or not the packets have been received, this makes UDP a lot faster than TCP, although far less reliable. UDP is suitable for applications which require a rapid rate of transmission, and where reliability is not of up-most importance, such as Audio/Video Chat.

The Internet Control Message Protocol (ICMP) is one of the core protocols of the Internet protocol suite, and is used by networked computers to send error messages, for example when a specified port cannot be reached. In the case that a service is not available or a host cannot be reached, there are a number of 'control messages' that end hosts or intermediate routers can use in order to inform other devices of these errors. One example is the ICMP PORT UNREACHABLE (ICMP Type 3, Code 3) which informs a requesting host that the requested port cannot be reached for some reason. Applications do not tend to use the ICMP protocol directly the ,but in certain cases ICMP can be used to transmit small amounts of information within the data field of the ICMP packet.

1.3 Three way handshake –

The Three-Way Handshake is the protocol that computers use in order to establish a TCP connection with each other.

1. The initiating machine will send a 'Hello' (formally called a SYN) packet to a specified host on a specified port. For example, if you browse to http://www.foo.com, your computer will first send the server a SYN packet on port 80 (the default web server port) to the server at www.foo.com. If port 80 is not open on the web server, then your client will not receive a reply (and the connection will fail).
2. However, if port 80 is open, then the web server is listening and will reply to the client with a SYN/ACK packet, acknowledging that it received the first (SYN) packet and requesting a confirmation to complete the connection.
3. Finally, the client will send back an ACK packet, signaling that it confirms the connection.

At this point, both computers keep track that they are connected to each other. It is also important to note that some machines will only log a connection1 once the full Three-Way Handshake has been performed. The connection is ended by one side or the other, by sending a FIN packet to the other end, who then replies with a FIN&ACK packet, and finally the initiating side sends back a final ACK packet.

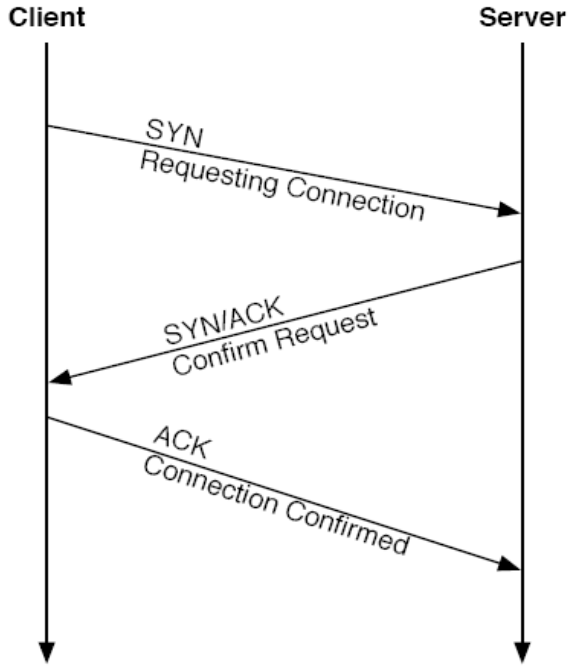

Figure 1 Three Way Handshake

II. CURRENT TECHNOLOGY AND TRENDS

2.1 Current System –

2.1.1The Vanilla Port –

The theme is to find a way to connect to the server or perform some kind of action on it. A port knocking daemon sits on the server and watches packets as they are dropped, waiting for a sequence of packets arriving at a predetermined set of ports in order. The client who wishes to connect to the server sends SYN packets (the first packet in a TCP connection) to the predetermined ports in order. The server will receive these packets and drop them silently, however, the port knocking daemon will see these incoming packets and recognize the valid 'knock' on the ports. Once the proper ports have been knocked on, the daemon can execute any predetermined action.

2.1.2 Vanilla Single Packet Authorization

Single Packet Authorization (SPA) and port knocking have the same aim but significantly different delivery mechanisms. In SPA the knock, which is called an Authorization Packet (AP), is encoded within a single packet. This can provide certain advantages not found in traditional port knocking schemes, such as eliminating the problem of out-of-order packet delivery. In traditional port knocking, if a knock sequence arrives out of order, which can easily happen as the server is not sending back any acknowledgements, then the port knocking daemon will not recognize the knock and thus access will not be allowed

Port Knocking schemes are, for the most part, network security methods for authenticating users and authorizing them to use a specific service (this may include performing an action on the protected machine). For these reasons it will be important to grasp the basics in host-based networking in order to understand how port knocking schemes function. This will serve as a very basic introduction to the aspects of networking which everyone feels most relevant in achieving this security.

Confidentiality, Integrity, and Authentication have become the primary concerns for protocols carrying sensitive traffic. Without these precautions, the information sent over the network could be vulnerable to unauthorized disclosure, modification, or we would simply be unsure as to who actually sent that information.

III. Proposed Design And Implementation

3.1 The Architecture–

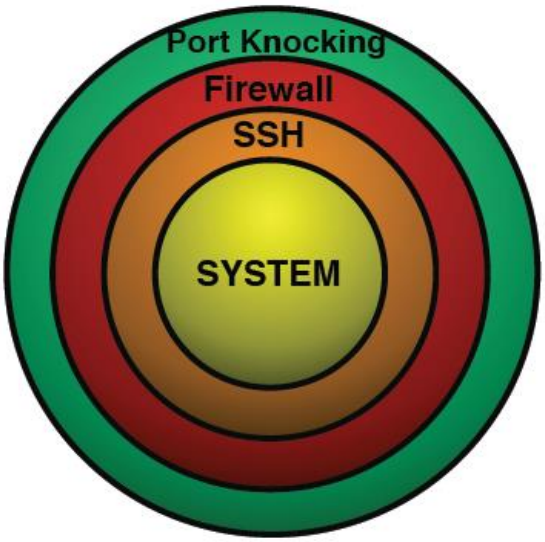

Figure 2 Port Knocking as an additional layer

Most current-day network services require some form of user authentication before a connection can be made. Services such as SSH and FTP both require a username and password when a connection is opened. Alternatively, SSH has the option of only allowing cryptographic keys to be used in the authentication process, denying the use of username and password combinations.

So in essence, SSH and FTP are layers which exist to protect the system and authenticate users before some kind of access is granted. The problem with networked services is that they are potentially vulnerable to attack, for example, by exploiting bugs in their code, or to simple dictionary attacks. For these reasons, port knocking allows the services to be hidden from the world, until a valid authenticated user attempts to connect. Figure above depicts how a system running SSH could be protected using a port knocking layer. Remember that in this configuration, the firewall is set to drop all traffic silently until a port is opened by the port knocking daemon.

3.2 Working Models –

Port knocking is a technique whereby authentication information is transmitted across closed network ports. A machine using port knocking closes all network ports to all hosts but logs incoming packets. A program watches the firewall logs for certain sequences of packets, which encode authentication information and requests to open or close ports. Based on this information, the port knocking system can choose to open network ports to the originating host.

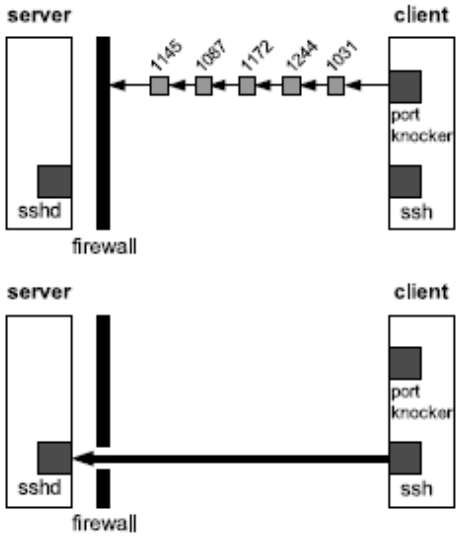

Figure 2: Port knocking example. The firewall is opened in response to a specific port sequence used for authentication.

As a simple example of port knocking, a server would close all ports and log requests to a specific port range; either TCP or UDP ports can be used. If a client transmits packets to a specific sequence of server ports (for instance, 1145, 1087, 1172, 1244, and 1031, in that order), then the server would perform some action (such as opening the SSH port to the client host). Here, the port sequence is a shared secret between the user and the server; knowledge of the secret implies that the user is authorized to access the protected service.

This particular application of port knocking is insecure, because an attacker could sniff the secret sequence from the network and replay it to get access to the protected service. However, other, more complex authentication procedures using port knocking exist such as those that transmit a cryptographic proof of knowledge of an authentication packet.

3.2.2 Implementation Model –

The most secure system is a system, which does not permit any connections from any external system. Such a system, though protected, is impractical - nobody can connect, independent of their trust status. This essentially describes a computer that is not networked. These kinds of computers are not a lot of fun. To discriminate between trusted and untrustworthy users - something that firewalls cannot natively do - an authentication method called Port Knocking is used.

In one instance, Port Knocking refers to a method of communication between two computers (arbitrary named here client and server) in which information is encoded, and possibly encrypted, into a sequence of port numbers. This sequence is termed the knock. The server initially presents no open ports to a public network and is monitoring all connection attempts. The client initiates connection attempts to the server by sending TCP SYN packets to the ports specified in the knock. This process of knocking is what gives port knocking its name. The server offers no response to the client during the knocking phase, as it "silently" processes the port sequence. When the Port Knocking daemon running on the server decodes a valid knock it triggers a server-side process and begins the session. Another knock sequence may used to trigger the closing of the port.

The following is an example of how port knocking can be used to completely secure a system against most attacks, achieving a high level of security while maintaining flexibility.

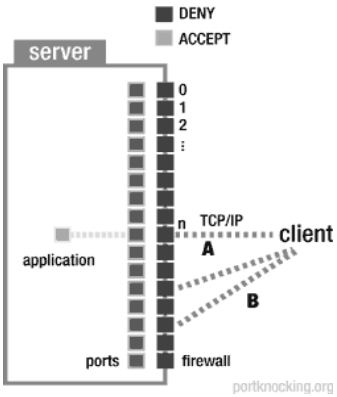

Figure 3: Step 1 (A) client cannot connect to application listening on port n; (B) client cannot establish connection to any port

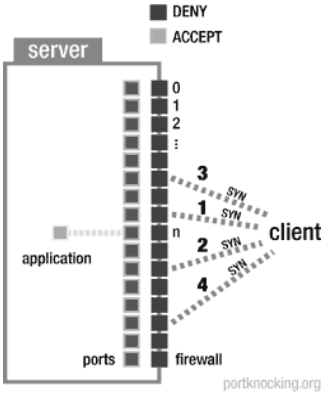

Figure 4: Step 2 | (1,2,3,4) client connects to a well-defined set of ports in a sequence that contains an encrypted message by sending SYN packets; client has a priori knowledge of the port knocking daemon and its configuration, but receives no acknowledgement during this phase because firewall rules preclude any response

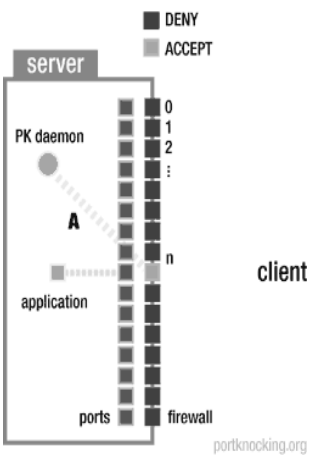

Figure 5: Step 3 | (A) server process (a port knocking daemon) intercepts connection attempts and interprets (decrypts and decodes) them as comprising an authentic "port knock"; server carries out specific task based on content of port knock, such as opening port n to client

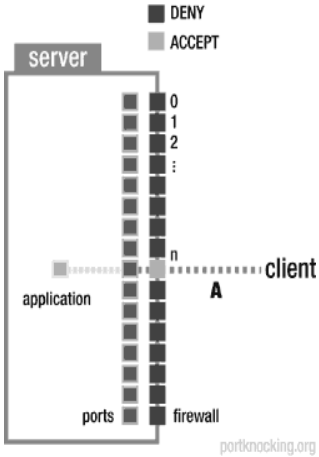

Figure 6: Step 4 | (A) client connects to port n and authenticates using application's regular mechanism

3.3 Implementation Model& Utilities –

3.2.1 Barricade –v0.0.2 –

Barricade, a simple implementation of the door knocking method aimed to open your network service or firewall only if a special icmp echo request packet is sniffed from the network interface. After the last valid packet received, barricade waits for a defined amount of time, then it closes your firewall or stops your services. There is a client included in the package called barricade_client that helps you to create special icmp packets containing the password

3.2.2 Bashportknocking -v0.0.1–

This release relies on cut, awk, iptables, bash and a significant amount of user customization required. It uses several scripts that work together to make the program run as a dameon and allow easier customization for different systems. It should be relatively easy to adapt for systems that don't use IP Tables but testing has not been done with it. Changes that should be made to future releases include moving to a config file format for all user customizations and creating options to disable or enable different features.

3.2.3 Cryptknock -v1.0.1–

Cryptknock is an encrypted port knocking tool. Unlike other port knockers which use TCP ports or other protocol information to signal the knock, an encrypted string is used as the knock. This makes it extremely difficult for an evesdropper to recover your knock (unlike other port knockers where tcpdump can be used to discover a port knock).

3.2.4 Doorman -v0.8 –

The doorman is intended to run on systems which have their firewall rules turned down tightly enough as to be effectively invisible to the outside world. The doorman adds and removes extra rules in a carefully controlled manner

IV. RESEARCH AND DEVELOPMENT

4.1 Scope –

It has become well established that the Internet is a hostile place. Any host connected to the Internet needs to be secured against unauthorized intrusion and other attacks. Unfortunately, the only secure system is one that is completely inaccessible, but, to be useful, many hosts need to make services accessible to other hosts. While some

services need to be accessible to anyone from any location, others should only be accessed by a limited number of people, or from a limited set of locations.

The port knocking most obvious way to limit access to required users and authenticate themselves before granting them access, Port Knocking schemes are, for the most part, network security methods for authenticating users and authorizing them to use a specific service (this may include performing an action on the protected machine). It provides the required security of the data transactions in the network. This internet age comprises of very dependent services , naturally almost all the financial transactions with the electronic funds transfer, e-payment require the secured mode, moreover the most important military communications, services need to more obscure, this port knocking provides the security through obscurity

4.2 Benefits –

Port knocking provides a stealthy method of authentication and information transfer to a host that has no open ports.

1. It is not possible to determine successfully whether the machine is listening for knock sequences.
2. It is unlikely that the form of connection attempts would be detected by monitoring traffic.
3. A sequence can corresponds to a request that a port be opened for a specific length of time and then closed.

4.3 Disadvantages –

1. Performance penalty: use of port knocking imposes an overhead for each connection.
2. A number of ports have to be allocated for exclusive use by port knocking.
3. In the case that no ports are initially open, if the listening daemon fails or is not able to interpret the knocks correctly, it becomes impossible to conect remotely to the host.

4.4 Convergence of Technology –

The principle of Port knocking has evolved from the concept of the Fire wall system, many protocols and cryptography.

Firewalls are an essential network component when it comes to controlling the flow of access in networked environments. In short, the goal of a firewall is to allow controlled connectivity between areas of differing trust levels, through the enforcement of a security policy and connectivity model, based on the principle of least privilege. Firewalls can be either hardware, which sits on a network between a trusted side and an untrusted side; or software, which runs on the hosts themselves. In both cases the purpose of the firewall is to provide a logical barrier to prevent unauthorized or unwanted communications between different areas of a computer network. The 'Access Control' mechanisms offered by a firewall rely on a set of administrator defined rules, which are then applied to each and every packet flowing through the firewall.

TCP, UDP and ICMP are three of the most important networking protocols used regularly in modern networks. TCP and UDP are the protocol that allows the two machines to communicate between themselves and exchange information. ICMP is one of the core protocols of the Internet protocol suite, and is used by networked computers to send error messages. 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. Confidentiality is a crucial element of network communications when private information is being stored or transmitted. The use of encryption has allowed us to protect such information and prevent it being disclosed to unauthorized parties.

V. CONCLUSION

Port knocking can be used to construct authentication systems on firewalls with the goal of only allowing authorized users access to open ports. Port knocking systems may be dismissed as simply .security through obscurity. While we disagree with this . port knocking can be done securely the fact of the matter is that such systems are in active use, and it is thus advisable to make them as strong as possible. They can be seen as complementary to existing defensive systems, as part of defense in depth, or as a means to secure weak legacy systems.

Existing port knocking systems have three main flaw: they do not always work reliably in the presence of NATs, they fail if packets are delivered out of order, and they do not associate authentication exchanges with connections

opened afterwards. The port knocking system that we have presented here improves on current systems by using a novel authentication algorithm that is unaffected by NAT and an efficient packet-reordering system that ensures that messages can be decoded on delivery.

Overall for a relatively small investment in complexity port-knocking returns a useful security dividend that doesn't require advanced knowledge of cryptography or network programming for an average user maintaining a couple of hobbyist servers. Even for a host with more secure requirements, such as an IDS system which by nature probably doesn't need to be broadcasting public services, port-knocking is an added layer of security which makes an attackers task more difficult. It offers a unique advantage in that if vulnerability does occur in a service being protecting with port-knocking, an administrator probably has some grace time to patch it, because the service is not widely available to anyone using a portscanner. Port knocking is an interesting concept and as implementations mature further should result in wider use and broader application within computer security.

VI. FUTURE SCOPE

Port Knocking in Malware (Backdoors) –

The concealment aspect of port knocking is a feature which may be of interest to malware writers, especially when a newly-opened port may be indicative that a host has been compromised and is running a 'listener' to allow the attacker to connect to the machine. Similarly, an open port may allow other attackers to discover the already-compromised machine and claim it as their own. Worms spread automatically, sometimes by scanning the local network (or the Internet) and connecting to vulnerable services running on discovered machines in order to exploit them (exactly one threat that port knocking aims to protect against), and it is in the worm owner's interest to maintain control over his network of compromised computers.

By implementing a form of port knocking into their worms, the authors can maintain control over machines that become compromised. This is already believed to be in use in trojans and worms, although no actual malware has actually been found to contain a port knocking implementation yet.

There do exist two implementations of port knocking that are particularly well suited for use in malware. Both of these implementations allow for the server to listen for a valid knock before opening a backdoor server for an attacker to connect. SAdoor and its predecessor both have this functionality with, and are particularly well suited for use as backdoors as opposed to normal port knocking mechanisms.

Port Knocking in Enterprise Environments –

Port knocking and SPA remain relatively novel ideas which have not been fully considered by the security community, and most implementations currently available are proof-of-concepts, with very few exceptions. Due to this, users in enterprise environments may be reluctant to adopt these mechanisms to help protect some of their vital servers for fear of unwanted side-effects or excessive server loads. Port knocking and SPA have probably never been tested to actively protect a system with more than 10 users, and if they have, the results of such trials would be highly interesting. However, limiting the use of such authentication mechanisms to the purpose of administrating servers, for example to allow admins to open port 22, or to allow them to run commands directly on the server without having to connect to them, is entirely within the scope of the current implementations.

REFERENCES

1) Jeanquier S (2006) An Analysis of Port Knocking and Single Packet Authorization. M.Sc. Thesis (Royal Holloway, University of London).
2) Krzywinski M (2005) Port Knocking From the Inside Out. *hakin9* **5**.
3) Rash M (2005) Combining Port Knocking and Passive OS Fingerprinting with fwknop
4) Krzywinski M (2003) Port Knocking: Network Authentication Across Closed Ports [txt]. *SysAdmin Magazine*
5) Graham-Cumming J (2004) Practical Port Knocking. *Dr. Dobb's Journal* **366**
6) Doyle M Implementing a Port Knocking System in C, *Department of Physics, University of Arkansas*
7) Maddock B (2004) Port Knocking: An overview of Concepts, Issues and Implementations. *SANS Institute*
8) M. Krzywinski, .portknocking.org,. URL: http://www.portknocking.org, accessed Nov.
9) http://new.linuxjournal.com/articles/web/2003-06/6811/6811l1.htm