ISSN (ONLINE): 2395-695X
ISSN (PRINT): 2395-695X
Available online at www.ijarbest.com
International Journal of Advanced Research in Biology, Engineering, Science and Technology
(IJARBEST) Vol. 2, Issue 1, January 2016

Port Knocking

K. Ketzial Jebaseeli[1], C.Subha[2], M. Benaiah Deva Kumar[3]
Assistant Professor, Department of Information Technology, Sri Krishna Arts and Science College, Coimbatore, India [1]
Assistant Professor, Department of Information Technology, Sri Krishna Arts and Science College, Coimbatore, India [2]
Scholar, Department of Information Technology, Sri Krishna Arts and Science College, Coimbatore, India [3]

Abstract-* “Port Knocking” aims to develop an application which could provide a high level secured communication between client and server. This paper would provide an insight into some of the challenges involved with designing and building client server architecture for communication.Normally client server communication systems provide authentication mechanisms which could be easily gets bypassed by well-known intruders. It creates intensive security vulnerability in the client server communication.So we need a facility to access the server with high confidentiality and integrity. The concept of port knocking is fairly simplistic, it does not allow access to network ports from external IP addresses without the client sending a specific sequence of port knocks. Once the correct port knock has been executed, a customizable script can be run that for example, creates a firewall rule, activates a service, or changes a run level.The project is based on the client server model and monitors network ports using Transmission Control Protocol (TCP) for primary communications. The server is used to generate a portknock file. A portknock is a sequence of ports like 8120,8100,12090,14532,2093.

***Index Terms-* Brute Force Attacks, IP Spoofing, Port Scanning, Single Sign ON.**

I. INTRODUCTION

In computer networking, port knocking is a method of externally open ports on a firewall by generating a connection attempt on a set of pre specified closed ports. Once a correct sequence of connection attempts is received, the firewall rules are dynamically modified to allow the host which sent the connection attempts to connect over specific port(s). A variant called single packet authorization exists, where only a single “knock” is needed, consisting of an encrypted packet. The primary purpose of port knocking is to prevent an attacker from scanning a system for potentially exploitable services by doing a port scan, because unless the attacker sends the correct knock sequence, the protected ports will appear closed [1].

II. EXISTING SYSTEM

The existing system normally uses the Single Sign ON authentication method to provide a secure client server communication. Unfortunately it is easy for a hacker to bypass this authentication mechanism and can steal confidential data. It is also noticeable that password breaking is effortless for experienced hackers. Some of the modern systems provide second level authentication like Access Control List. Even though these mechanisms provide a good security level, hackers used to bypass these mechanisms by IP spoofing, port scanning, brute force attacks etc.

III. DRAWBACKS

- Low level security
- Bypass password authentication is easy
- Attacks based on port scanning, brute force happens
- Server hacks based on IP spoofing

IV. PROPOSED SYSTEM

With increasing attacks from hostile networks to all externally exposed ports a need has been identified to hide services from these high threat networks. A method known as port knocking has been developed to allow ports to only be exposed after connects are made in a specific order. The act of opening and closing ports as part of a combination of is termed as port knocking. Providing the correct sequence of portknocks can allow firewall rules to be dynamically reconfigured or any scripted action to occur. The primary threats that are reduced by using port knocking technology are automated vulnerability scanning, system profiling, and brute force attacks.The concept of port knocking is fairly simplistic, it does not allow access to network ports from external IP addresses without the client sending a specific sequence of port knocks. Once the correct port knock has been executed, a customizable script can be run that for example, creates a firewall rule, activates a service, or changes a run level. Our implementation of the port knocker concept is based on the client server model and monitors network ports using Transmission Control Protocol (TCP) for primary communications.The following overall decisions were made for this project during the design phase.

- TCP would be used to reduce the risk of spoofing attempts.
- Decoy ports will be opened to prevent port scanning and brute force port knocks.
- The number of ports contained in a port knock would be user configurable.
- Since all network connection requests are visible to network observation, no successful port knock would be reused.
- The server would have a centralized port that the clients would use to retrieve the current portknock.
- The server would shutdown for sixty seconds after an incorrect port knock was attempted.
- Java would be our primary development language.
- No efforts would be made to securely transfer the initial secret, the file containing the generated port knocks.
- A successful knock would be defined as the client opening and closing ports in a defined order over a specified period of time.
- A successful knock will execute a script on the server and leave the script function and scope to the administrator.
- Logging subsystem will record the connection times, IP addresses, and port knocks on the server component

ISSN (ONLINE) : 2395-695X
ISSN (PRINT) : 2395-695X
Available online at www.ijarbest.com
International Journal of Advanced Research in Biology, Engineering, Science and Technology (IJARBEST) Vol. 2, Issue 1, January 2016

and will be used detection of malicious events and troubleshooting.

- Logging on the client will record port knock attempts and will be used for troubleshooting client issues.

V. ADVANTAGES

- High level security.
- Authentication bypassing using brute force is impossible.
- Port scanning will not work effectively to hack the machine.
- Server hacks based on IP spoofing will never happen.
- Problem definition.
- The existing system normally uses the single sign on authentication method to provide a secure client server communication. Unfortunately it is easy for a hacker to bypass this authentication mechanism and can steal confidential data. It is also noticeable that password breaking is effortless for experienced hackers. Some of the modern systems provide second level authentication like access control list. Even though these mechanisms provide a good security level, hackers used to bypass these mechanisms by IP spoofing, port scanning, brute force attacks etc.
- With increasing attacks from hostile networks to all externally exposed ports a need has been identified to hide services from these high threat networks. A method known as port knocking has been developed to allow ports to only be exposed after connects are made in a specific order. The act of opening and closing ports as part of a combination of is termed as port knocking. Providing the correct sequence of portknocks can allow firewall rules to be dynamically reconfigured or any scripted action to occur. The primary threats that are reduced by using port knocking technology are automated vulnerability scanning, system profiling, and brute force attacks.
- The concept of port knocking is fairly simplistic, it does not allow access to network ports from external IP addresses without the client sending a specific sequence of port knocks. Once the correct port knock has been executed, a customizable script can be run that for example, creates a firewall rule, activates a service, or changes a run level. Our implementation of the port knocker concept is based on the client server model and monitors network ports using transmission control protocol (TCP) for primary communications.

VI. MODULE DESCRIPTION

As long as ports remain open, network applications are susceptible to attack, hence ports are initially closed. A handful of ports are configured to deny all traffic – no ICMP error packets are sent back to the connecting client. Users make connection attempts to sequence of closed ports. All failed connection attempts are logged by the server-side packet filtering firewall and detected by a daemon that monitors the firewall log file. When a properly formatted knock sequence is received, firewall rules are manipulated based on the information content of the sequence.

VII. MODULES

This project concentrated in the following list of modules:

- Server/Client Setup.
- Port Knock.
- Configuration.
- Server/Client Startup.

A. SERVER/CLIENT SETUP

In this module, the server will first get setup before the client. The server will generate the keys and writes into a file. This file will be then shared with clients.

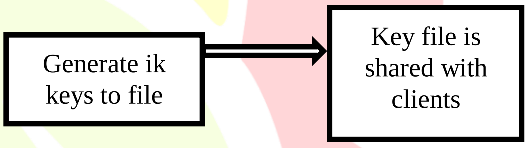

Fig. 1 Server side setup

B. PORT KNOCK

In this module, the client will try to connect to the server with the predefined port hits. Each client will be having unique knock sequence. These knock sequences will be stored in the configuration file. The server and client will share this configuration file at the time of startup.

C. CONFIGURATION

In this module the server will generate the sequence keys for the knock.

The sequence can be like this:

8120,8100,12090,14532,2093

The sever will then checks with the configuration file, in which all the settings are made. In this module the client will make a secret handshake with the server daemon. The client needs a secret knock sequence number to deal with the server. Here, client will send the server with a request for the secret handshake. When the server receives & verifies this handshake, it will grand with the sequence number.

D. SERVER/CLIENT STARTUP

In this module, the sever daemon will start running. The server will monitor the knock happening on each port. A daemon will be running in thin module. It will read the configuration file, detect the knock sequence, and then if it is an

ISSN (ONLINE) : 2395-695X
ISSN (PRINT) : 2395-695X
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology, Engineering, Science and Technology (IJARBEST) Vol. 2, Issue 1, January 2016

authorized one, it will then allow the client to deal with the server resources.

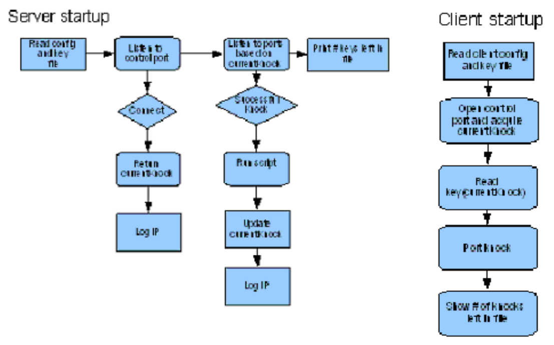

Fig. 2 Server/Client startup

- These modules itself describes the different functionalities of the system. The initial design for our portknocker project was based on the following diagram that we developed in our early planning sessions.

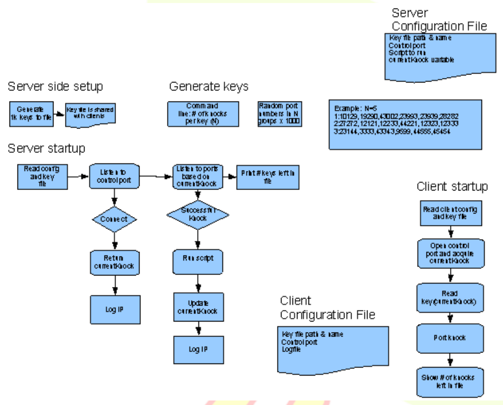

Fig. 3 Server/Client startup

VIII. IMPLEMENTATION DETAILS

Once the portknock file has been generated and securely transferred to the clients, our portknocker application is ready to be used. Although all applications should have security reviews and assessments, our application has been designed with security as forethought.

By using a complete TCP session as a requirement for the portknock we are reducing the threat of an attacker spoofing a portknock. While it would still be possible if the OS uses predictable sequence numbers, the likelihood is reduced because of the TCP session handshake and two-way communication.

By using decoy ports we are creating a large keyspace for our portknock secret. Using the default six ports for each portknock out of one hundred total decoy ports it creates a keyspace of over 1.19 billion (1.19 * 10 to the power of 9) combinations. Using seven ports, the keyspace moves the total combinations to over 1.60*10 to the power of 10.

Under normal operations the client is requesting the current knock number from the server, which does not allow any other clients to connect while the first client applies the portknock. Since all clients are synchronized on the current knock there should be no failures and all other activity is assumed malicious. Our response to this is to have the server lockout for sixty seconds on all failed attempts.

The combination of a large keyspace and server lockout would severely inhibit an attack on our port knocker application. An attacker would be only able to attempt a portknock fourteen hundred times a day in a keyspace of over a billion possible keys.

Our client server communication uses TCP ports to communicate a secret – a series of port connects and disconnects. The client and server communication and program flow is as follows.

The server is used to generate a portknock file. A portknock is a sequence of ports like 8120,8100,12090,14532,2093. By default, the generated portknock file contains 1000 portknocks with six ports per knock.

- The portknock file is securely communicated to the client.
- The client loads and reads its configuration file.
- From the configuration file the client learns of the centralized port and the location of the portknock file.
- The client opens a connection to the centralized port and retrieves the current knock file number.
- The server opens the ports identified in its copy of the portknock file and additional decoy ports.
- The client retrieves the portknock that corresponds to the current knock file number.
- The client opens and closes the ports from the portknock in the specified order.
- Once the client finishes with the correct ports the server executes the predefined script.

IX. CONCLUSION

Transactions performed by the system are carefully studied in order to attain the aim with accurate results. Each task is divided in to modules. Hence modifications and enhancement can be easily made without affecting any other part of the program. All modules are tested separately and put together to the form the main system. Thus the system has fulfilled all the objectives identified. The system had been developed in an attractive dialogs fashion so user minimum knowledge about computers can also operate the system easily. It has provision for future changes. Succinctly, the new system is more reliable, accurate, secure and effective. The system is flexible for maintenance. It generates reports very quickly and also it reduces the chance of occurring errors. Port Knocker is used for the client server communication. It provides an interface to manage the server in an effective manner. It provides high level security. Port knocking can be further enhanced with a wide variety of concept like generating the knock file automatically, loading the knock file to a centralized remote server with enough security measures etc.

REFERENCES

[1]https://en.m.wikipedia.org/wiki/Port_knocking
[2]www.duderman.com
[3]www.portknocking.org
[4]www.cipherdyne.org
[5]www.debian-administration.org