A

Project Report

on

Network Admission Control using Single Packet Authorization

Developed at

Space Applications Centre
Indian Space Research Organisation
Satellite Road, Ahmedabad-380015

Developed by

Shyam Makwana – Department of IT, DD University

Guided By

Internal Guide:
Prof. (Dr.) Vipul K. Dabhi
Department of Information Technology
Faculty of Technology
DD University

External Guide:
Sri. Rohit Tyagi
SCI/ENGR-SD
SAC, ISRO
Ahmedabad, Gujarat

Department of Information Technology
Faculty of Technology, Dharmsinh Desai University
College Road, Nadiad-387001

April - 2022

COMPANY CERTIFICATE

भारत सरकार
अंतरिक्ष विभाग
अंतरिक्ष उपयोग केन्द्र
आंबावाडी विस्तार डाक घर,
अहमदाबाद-380 015. (भारत)
दूरभाष : +91-79-26913050, 26913060
वेबसाईट : www.sac.isro.gov.in/www.sac.gov.in

Government of India
Department of Space
SPACE APPLICATIONS CENTRE
Ambawadi Vistar P.O.
Ahmedabad - 380 015. (INDIA)
Telephone : +91-79-26913050, 26913060
website : www.sac.isro.gov.in/www.sac.gov.in

Scientific Research and Training Division (SRTD)
Research, Outreach and Training Coordination Group (RTCG)
Management and Information Systems Area (MISA)

CERTIFICATE

This is to certify that **Mr. Shyam B. Makwana**, B.Tech., Department of Information Technology, Dharmsinh Desai University, Nadiad, has satisfactorily completed his project **"NETWORK ADMISSION CONTROL USING SINGLE PACKET AUTHORIZATION"** under the guidance of **Sri. Rohit Tyagi,** SCI/ENGR-SD, MISA/CSIG/ITND, Space Applications Centre, Ahmedabad from 06-12-2021 to 26-03-2022. The research work was carried out under **Training and Research in Earth Eco-System (TREES)** Program of Space Applications Centre, Ahmedabad.

डॉ सर्वेश्वर व्यास / Dr. S P Vyas
प्रधान, वैज्ञानिक अनुसन्धान एवं प्रशिक्षण प्रभाग
Head, Scientific Research and Training Division
एसआरटीडी-आरटीसीजी-मीसा/ SRTD-RTCG-MISA
अंतरिक्ष उपयोग केंद्र (इसरो)
Space Applications Centre(ISRO)
अंतरिक्ष विभाग / Department of Space
भारत सरकार / Government of India
अहमदाबाद / Ahmedabad - 380015

CANDIDATE'S DECLARATION

I declare that the final semester report entitled "Network Admission Control using Single Packet Authorization" is my own work conducted under the supervision of the external guide Sri. Rohit Tyagi from Space Applications Centre, ISRO.

I further declare that to the best of our knowledge, the report for B.Tech. Final semester does not contain part of the work which has been submitted for the award of B.Tech. Degree either in this or any other university without proper citation.

Shyam Makwana
Branch: Information Technology Student ID: 18ITUOS079

DHARMSINH DESAI UNIVERSITY

NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled "Network Admission Control using Single Packet Authorization" is a bonafide report of the work carried out by Mr. Shyam Makwana, Student ID No: 18ITUOS079 of Department of Information Technology, semester VIII, under the guidance and supervision for the award of the degree of Bachelor of Technology at Dharmsinh Desai University, Nadiad. (Gujarat). He was involved in Project training during the academic year 2021-2022.

Prof. (Dr.) V. K. Dabhi,
(Project Guide)
Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date: 9/4/2022

Prof. (Dr.) V. K. Dabhi,
Head, Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date: 9/4/2022

ACKNOWLEDGEMENT

I would like to express my most profound appreciation to all those who have helped me directly or indirectly during this training period. I am more thankful to the department and institute that have given us the opportunity to work in the industry.

I am thankful to Dr. S. P. Vyas, Head & SCI/ENG-SG, SRTD/RTCG/MISA, and Sri. H. J. Kotecha, Head & SCI/ENG-SG, MISA/CSIG/ITND, Space Applications Centre, ISRO for giving me the opportunity to work at their esteemed organization and encouraging me to push my boundaries during this training period. I am indeed thankful to Sri. Rohit Tyagi, who has been a great mentor during this time and helped me a lot during the project work.

My sincere thanks to the internal guide and Head of the Information Technology Department, Prof. (Dr.) Vipul K. Dabhi, who gave me the opportunity to gain practical knowledge in the industry. I am grateful to them for their guidance, encouragement, and insightful support during the training period.

Last but not least, I would like to thank the employees of Space Applications Centre, all the faculty members of our IT department, friends, and family for encouraging us throughout the project.

With Sincere Regards,
Shyam Makwana

TABLE OF CONTENTS

ABSTRACT

Traditional network security approaches, like LAN, VPN, and Firewall, have demonstrably failed to adequately protect organizations today. The reason for the failure is that TCP/IP—which was initially designed to operate in an environment where the user community knew and trusted each other—is based on implicit trust, with a "connect first, authenticate second" approach. In today's hyperconnected and highly adversarial threat landscape, this approach puts organizations at risk and has enabled far too many data breaches.

The terms "authentication" and "authorization" in this project are commonly construed to mean the same thing. However, authentication refers to the verification that communication from one party to another came from the first party, whereas authorization essentially refers to the process of verifying whether one party is allowed to communicate with a second party at all.

"Single Packet Authorization" is an authentication protocol through which a single packet is sent; based on that Packet, an authentication process is carried out. A key point to note is that nothing is listening on the Service itself, so you have no open ports. For the SPA service to operate, there is not anything explicitly listening. When the Client sends a SPA packet, the Packet will be rejected, but a second service (Packet Sniffer) identifies the SPA packet and then authenticates it. If the SPA packet is successfully authenticated, the SPA Server will open a port in the Server's Firewall so that the Client can establish a secure and encrypted connection with the intended Service.

COMPANY PROFILE

Space Applications Centre, ISRO

Space Applications Centre (SAC) is one of the major centres of the Indian Space Research Organisation (ISRO). SAC focuses on designing spaceborne instruments for ISRO missions and the development and operationalization of applications of space technology for societal benefits. The applications cover communication, broadcasting, navigation, disaster monitoring, meteorology, oceanography, environment monitoring, and natural resources survey.

It plays a crucial role in realizing the vision and mission of ISRO. Located in Ahmedabad, SAC is spread across two campuses having multi-disciplinary activities. The centre's core competence lies in the development of spaceborne and airborne instruments/payloads and their applications for national development and societal benefits. These applications are in diverse areas and primarily meet the country's communication, navigation, and remote sensing needs. Besides these, the center also contributes significantly to scientific and planetary missions of ISRO like Chandrayan-1, Mars Orbiter Mission, etc.

Company's website for more info: https://www.sac.gov.in/Vyom/overview.jsp

LIST OF FIGURES

ABBREVIATIONS

- ZTNA – Zero Trust Network Access
- ZTS – Zero Trust Security
- SDP – Software Defined Perimeter
- NAC – Network Admission Control
- FWKNOP – Firewall Knock Operator
- SPA – Single Packet Authorization
- TCP – Transmission Control Protocol
- UDP – User Datagram Protocol
- IP – Internet Protocol
- LAN – Local Area Network
- VPN – Virtual Private Network
- AES – Advanced Encryption Standard
- HMAC – Hash Message Authentication Code
- RSA – Rivest–Shamir–Adleman
- NMAP – Network Mapper

1. INTRODUCTION

1.1 INTRODUCTION TO RESEARCH PROBLEM

Single Packet Authorization (SPA) uses cryptographic techniques to make internet-facing servers invisible to unauthorized users. SPA (and port knocking) employs a default-drop stance that provides Service only to those IP addresses that can prove their identity via a passive mechanism. It hides all the ports from the external world to reduce attacks until a cryptographically protected packet is received and authorized by the SPA Server. Only devices that have been seeded with the cryptographic secret will generate a valid SPA packet and send it to SPA Server for authentication, which Packet Sniffer captures before the Firewall drops it and subsequently be able to establish a network connection with the desired port. This is how it reduces the attack surface and becomes invisible to attackers as there are no open ports that attackers can scan.

1.2 MOTIVATION

In today's hybrid, distributed IT estates, it doesn't matter where resources reside because, from a network perspective, they all look the same protocols associated with ports. The problem is that unless you're physically plugging into a LAN, you're accessing resources over a network, using a range of ports (22,80,443….) and protocols (Tcp, Udp, Icmp…). Those ports are exposed doors into your resources listening for inbound connections and susceptible to exploitation.

The problem with this approach is that once an attacker gains access to the network, they have free rein over everything inside. It is hard to obtain access from outside the LAN, but everyone inside the network is trusted by default. To overcome this issue, SPA uses Zero Trust Security Architecture to block all the ports in the Firewall by default, so no user can see which ports are running, and uses a Single Encrypted Authorization Packet for accessing specific services running in the LAN.

1.3 OBJECTIVE

The primary objective is to access services running in the network and securely communicate authentication and authorization information across closed firewall ports, usually with the goal of opening specific ports to allow temporary access. Keeping most or all ports closed on a server hosting remotely-accessible services makes it possible to make that host invisible to the outside, thus protecting each listening service. Even with the dynamic IP Address, it is easier for an administrator to maintain the Server as the IP Addresses are added automatically based on the SPA Packet.

1.4 SCOPE

The scope of this research work enables the SDP Gateway of any organization to distinguish authorized and unauthorized connection attempts while only needing to evaluate a single network packet. This means that SPA makes SDP systems much more efficient and less susceptible to brute force attacks while protecting against a wide range of network-based attacks (DDoS, MITM, CSRF, XSS, and SQLi).

2. BACKGROUND THEORY

The knowledge required prior to the project includes understanding the Zero trust Network Access, Software Defined Perimeter, Local area network, Firewall, Packet sniffer, and Cryptography. It can be easily obtained through blogs and websites available online. Moreover, the study of tools and technology is also necessary.

2.1 UNDERSTANDING OF DOMAIN

2.1.1 Zero Trust Network Access

The Zero-Trust Network Access is an advanced security model that operates on the fundamental principle: **trust no one – verify everything**. In other words, no user or device trying to gain access to a network, regardless of their location, will ever trust until they are entirely verified based on the established identity and access control policies.

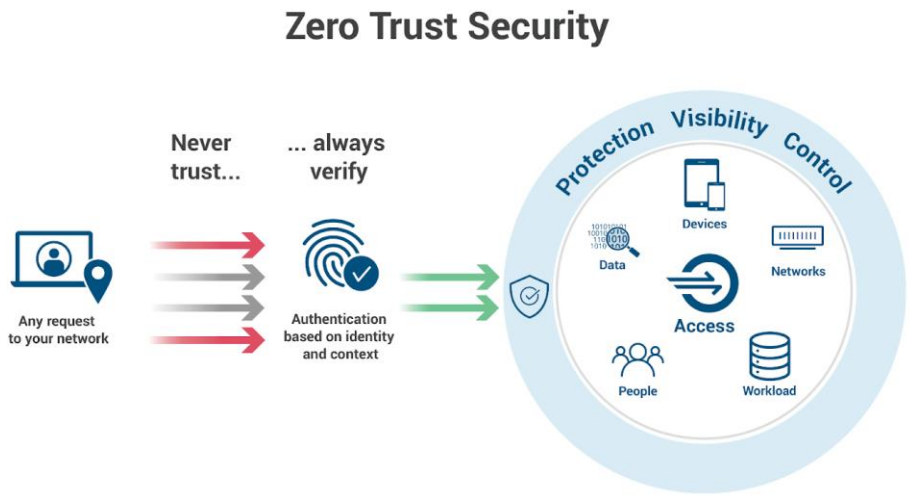


Figure 2.1.1.1 Architecture of Zero Trust Security

ZTNA ensures that every person and managed device attempting to access resources on a zero-trust network undergoes a tight identity verification and authentication process, whether they are inside or outside the network perimeter. Once ZTNA has established access and validated the user, the system grants the user access to the Application over a secure, encrypted channel. This adds an extra layer of security to corporate apps and services by hiding IP addresses that would otherwise be exposed to the public.

2.1.2 Software Defined Perimeter

A software-defined perimeter (SDP) is a way to hide Internet-connected infrastructure (servers, routers, etc.) so that external parties and attackers cannot see it, whether it is hosted on-premise or in the cloud. The goal of the SDP approach is to base the network perimeter on software instead of hardware. A company that uses an SDP essentially drapes a cloak of invisibility over its servers and other infrastructure so that no one can see it from the outside or inside the network; however, authorized users can still access the infrastructure.

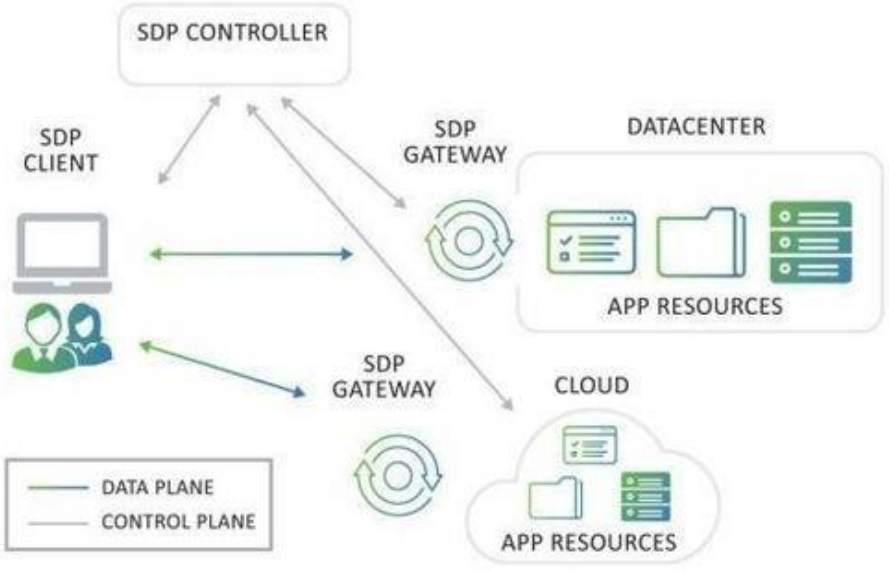


Figure 2.1.2.1 Architecture of Software Defined Perimeter

A software-defined perimeter forms a virtual boundary around company assets at the network layer, not the application layer. This separates it from other access-based controls that restrict user privileges but allow broad network access. Another key difference is that an SDP authenticates devices as well as user identity. The Cloud Security Alliance first developed the SDP concept.

2.1.3 Local Area Network

A local area network (LAN) is a collection of devices connected in one physical location, such as a building, office, organization, or home. A LAN can be small or large, ranging from a home network with one user to an enterprise network with thousands of users and devices in an office or school. A LAN comprises cables, access points, switches, routers, and other components that enable devices to connect to internal servers, web servers, and other LANs via wide area networks.

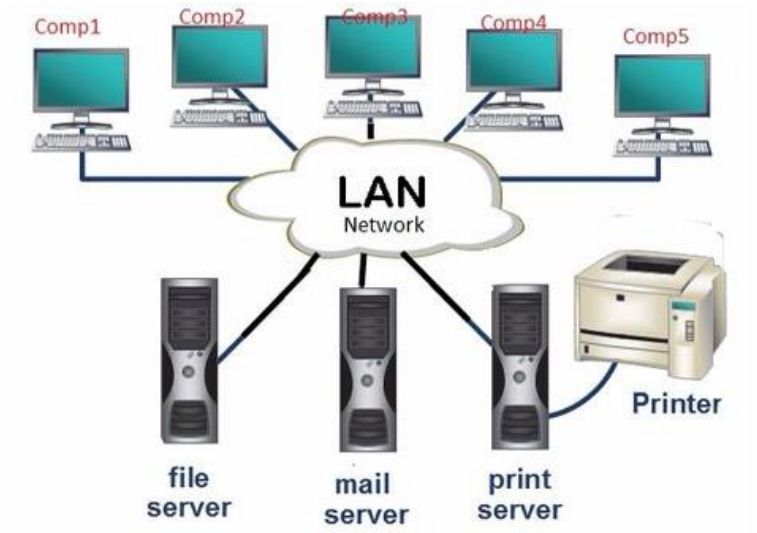


Figure 2.1.3.1 Architecture of Local Area Network

2.1.4 Firewall

A firewall can be defined as a particular type of network security device or a software program that monitors and filters incoming and outgoing network traffic based on a defined set of security rules. It acts as a barrier between internal private networks and external sources (such as the public Internet).

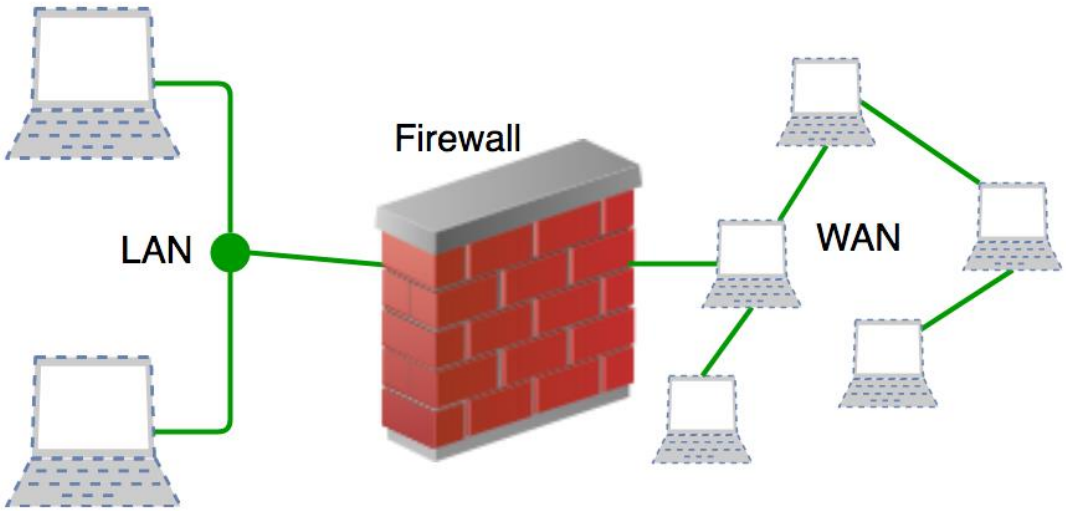


Figure 2.1.4.1 Firewall

A firewall's primary purpose is to allow non-threatening traffic and prevent malicious or unwanted data traffic to protect the computer from viruses and attacks. A firewall is a cybersecurity tool that filters network traffic and helps users block malicious software from accessing the Internet on infected computers.

	Source IP	Dest. IP	Source Port	Dest. Port	Action
1	192.168.21.0	--	--	--	deny
2	--	--	--	23	deny
3	--	192.168.21.3	--	--	deny
4	--	192.168.21.0	--	>1023	Allow

Sample Packet Filter Firewall Rule

Figure 2.1.4.2 Filtering table of Firewall

2.1.5 Packet Sniffer

Packet sniffers are applications or utilities that read data packets traversing the network within the Transmission Control Protocol/Internet Protocol (TCP/IP) layer. When in the hands of network administrators, these tools "sniff" internet traffic in real-time, monitoring the data, which can then be interpreted to evaluate and diagnose performance problems within servers, networks, hubs, and applications, while hackers may use similar tools for nefarious purposes. In this project, Packet Sniffer is built in C++ with the PcapPlusPlus library (which is built on packet processing engines such as libpcap, WinPcap, NPcap, DPDK, and PF_RING).

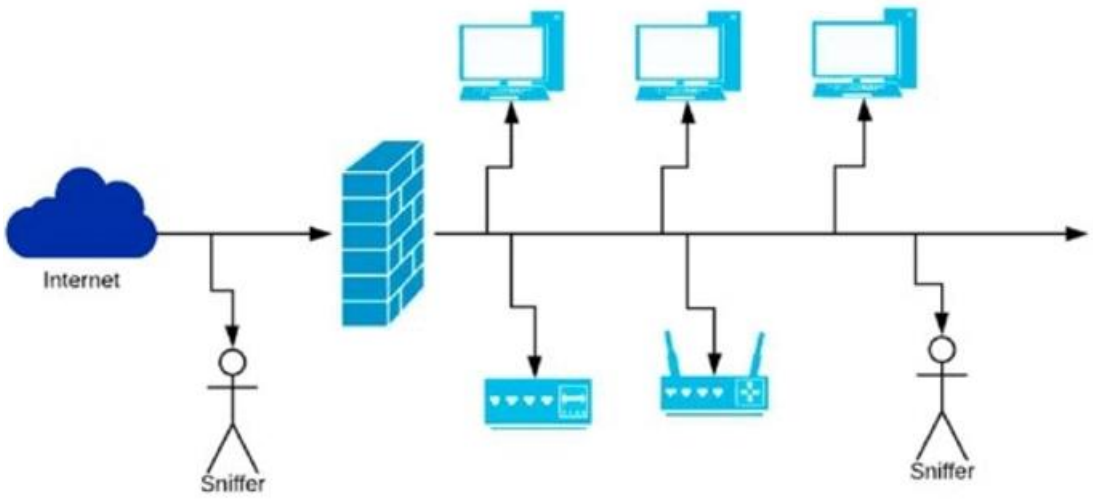


Figure 2.1.5.1 Sniffing the Packet from the network

2.1.6 Cryptography

Cryptography is the study of secure communications techniques that allow only the sender and intended recipient of a message to view its contents. The term is derived from the Greek word kryptos, which means hidden, and it is closely associated with encryption, which is the act of scrambling ordinary text into what's known as ciphertext and then back again into plain text upon arrival.

Encryption is one of the most common ways to protect sensitive data. Encryption works by taking plain text and converting it into ciphertext, consisting of seemingly random characters. Only those who have the secret key can decrypt it. AES uses symmetric key encryption, which uses only one secret key to cipher and decipher the information.

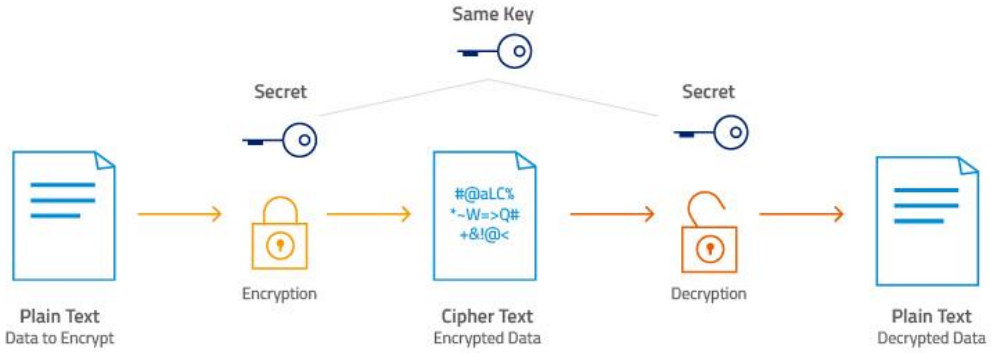


Figure 2.1.6.1 Flow of AES Encryption

HMAC (Hash-based Message Authentication Code) provides the Client and Server with a shared private key known only to them. The Client makes a unique hash (HMAC) for every request. When the Client requests the Server, it hashes the requested data with a private key and sends it as a part of the request. The message and key are hashed in separate steps, making it secure. When the Server receives the request, it makes its HMAC. Both the HMACS are compared, and if both are equal, the Client is considered legitimate.

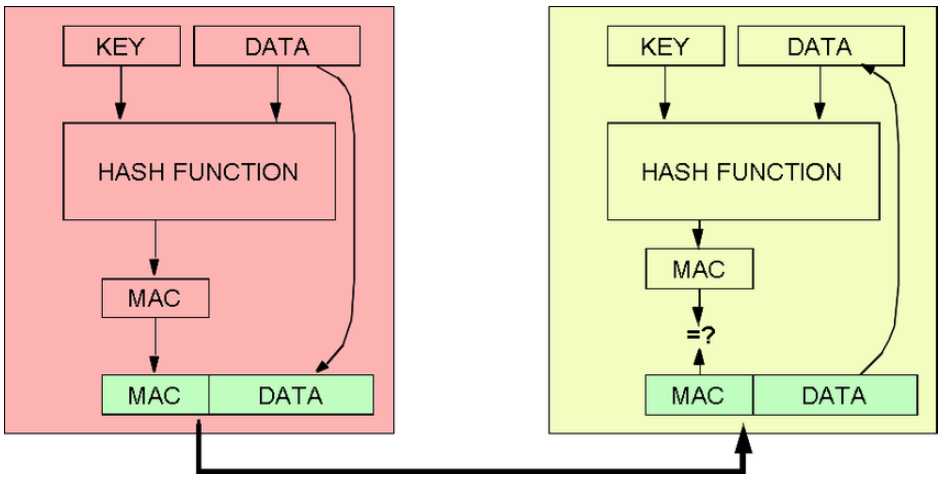


Figure 2.1.6.2 Flow of HMAC Authentication

When transmitting electronic data, the most common use of cryptography is to encrypt and decrypt email and other plain text messages. The simplest method uses the symmetric or "secret key" system. Here, data is encrypted using a secret key, and then both the encoded message and secret key are sent to the recipient for decryption. The problem? If the message is intercepted, a third party has everything they need to decrypt and read the message. To address this issue, cryptologists devised the asymmetric or "public key" system. In this case, every user has two keys: one public and one private. Senders request the public key of their intended recipient, encrypt the message and send it along. When the message arrives, only the recipient's private key will decode it — meaning theft is of no use without the corresponding private key.

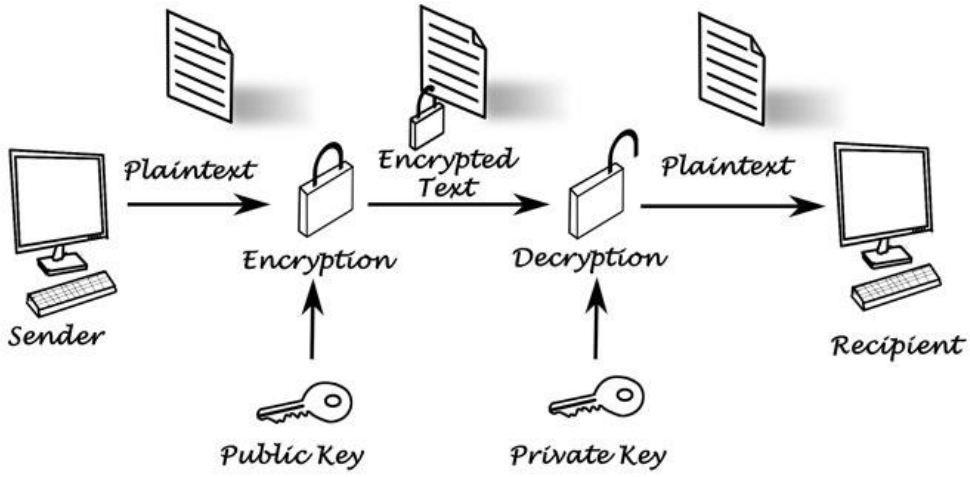


Figure 2.1.6.3 Flow of RSA Cryptography

2.2 STUDY OF THE CURRENT SYSTEM

2.2.1 Local Area Network

The function of Local Area Networks is to link computers together and provide shared access to printers, files, and other services. Local area network architecture is categorized as either peer-to-peer or client-server. On a client-server local area network, multiple client-devices are connected to a central server, and all the clients in the network can access all the services running in the network because all the ports are open and quickly scan which ports are running using Nmap. Applications running on the Local Area Network server provide services such as database access, document sharing, email, and printing.

2.2.2 Static Ip In Firewall

A static IP address is simply an address that doesn't change. Once your device is assigned a static IP address, that number typically stays the same until the device is decommissioned or your network architecture changes. Static IP addresses generally are used by servers or other essential equipment. Once static IP is added to the Firewall, it has to be changed manually by an administrator, and the Administrator has to allow all the IP addresses of all the devices in the network to make the task easy for him. So, they add a single group of IPs in which all the devices' IPs are stored. Thus, all the clients in the network have access to all the services running in the network.

2.2.3 Port Knocking

Port knocking is a first-generation technology that uses the port fields within TCP and UDP packet headers to communicate information. Normally, these protocols encapsulate application layer data, but port knocking encodes information in sequences of packets to various ports by using the port numbers themselves as fields to transmit data. These packets are typically either monitored out of a firewall log or via a packet capture mechanism, such as libpcap. The Client is responsible for generating the port sequences, and the Server is responsible for passively collecting the sequences and reconfiguring the packet filter to allow connections to protected services upon receipt of a valid sequence. For example, the Server could require the Client to send TCP SYN packets to the following ports in order: 7000->8000->9000

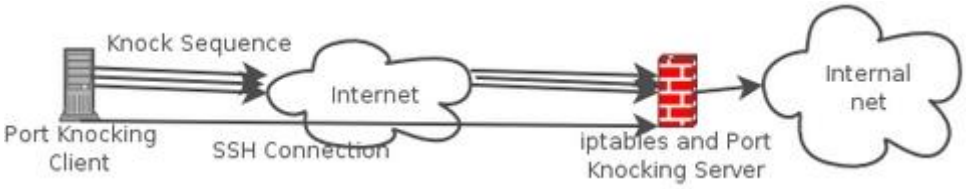


Figure 2.2.3.1 Architecture of Port Knocking for Linux

2.3 WEAKNESS OF THE CURRENT SYSTEM

2.3.1 Local Area Network

LAN relies on open ports to establish connections. The problem is that cybercriminals often target open ports and use them to access networks. Once authenticated users are inside the network, they have unrestricted access, which exposes the network to threats. This design flaw leaves the company's data, applications, and intellectual property vulnerable to attacks. LAN does not require identification for both users and devices trying to access a network. And with users always having poor password practices and not to mention the millions of stolen user credentials available for sale on the dark web, hackers can capture and bypass two-factor authentication codes on your online accounts. LAN is an expensive and time-consuming procedure requiring much effort from the security team and users. In addition, LAN is not a secure network security solution because of the typical technological vulnerabilities that increase the attack surface.

2.3.2 Static Ip In Firewall

Firewall rules are added with static IP, so it is hard to change if the IP Address changes of any Client. Even the Administrator has to add the whole LAN IP Address group in the Firewall to access all the devices in the network. If the Administrator wants to give a few clients different services than another, he has to create a new LAN for that particular group, and that combination of groups increases with varying possibilities of connections. So, the Administrator has to do all this configuration manually frequently. With a static IP address, hackers know exactly where your Server is on the Internet. That makes it easier for them to attack it. Anyone with the right network tools can find where you and your computers are located.

2.3.3 Port Knocking

The limitation of port knocking is that each Packet within a port knock sequence can send only two bytes of information due to the 16-bit-wide port fields in the TCP and UDP headers. Hence, an encrypted sequence must contain at least B/(2*8) packets for a block cipher, where B is the block size in bits. This would not be so bad when considering today's networks' general speed and reliability, but the real issue is out-of-order delivery. Packets may take different routing paths, some of which may be slow. It is difficult to guard against a replay attack effectively. Anyone who can monitor a knock sequence as it is sent from the Client to the Server is free to replay the sequence against the Server in an effort to gain the same access.

2.4 STUDY OF RELATED TOOLS AND TECHNOLOGIES

2.4.1 C#

C# (pronounced "See Sharp") is a modern, object-oriented, and type-safe programming language launched by Microsoft in 2001. C# enables developers to build many types of secure and robust applications that run in .NET. C# has its roots in the C family of languages and will be immediately familiar to C, C++, Java, and JavaScript programmers. C# provides language constructs to directly support these concepts, making C# a natural language in which to create and use software components. Since its origin, C# has added features to support new workloads and emerging software design practices. At its core, C# is an object-oriented language, and you define types and their behavior.

The purpose of C# was to develop a programming language that is not only easy to learn but also supports modern-day functionality for all kinds of software development. C# is open source under the .NET Foundation, governed and run independently of Microsoft. C# language specifications, compilers, and related tools are open source projects on Github. While Microsoft leads c # language feature design, the open-source community is active in language development and improvements. C# is fast compared to several other high-level programming languages.

2.4.2 Visual Studio

Visual Studio is an Integrated Development Environment(IDE) developed by Microsoft to develop GUI(Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silverlight, Windows API, etc. It is not a language-specific IDE as you can use this to write code in C#, C++, VB(Visual Basic), Python, JavaScript, and many more languages. It provides support for 36 different programming languages. It is available for Windows as well as for macOS.

3. REVIEW OF LITERATURE

3.1 SUMMARY OF STUDIED RESEARCH PAPERS

The overall summary from the above three research papers published by Michael Rash, the inventor of FWKNOP, the Linux version of SPA, is that Single Packet Authorization provides similar security benefits to port knocking in terms of protecting services with a packet filter configured in a default-drop stance. Anyone scanning for a protected target service in this way will be unable to detect such a service is listening, which makes even the exploitation of zero-day vulnerabilities much more difficult. Single Packet Authorization mandates a similar architecture to port knocking. Both have Client and server components, the Server maintains control of a default-drop packet filter, and the Server monitors packets passively.

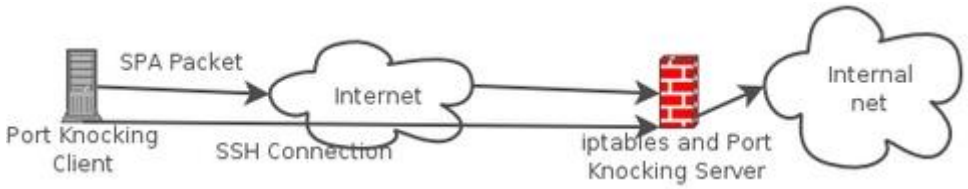


Figure 3.1.1 Architecture of SPA for Linux

SPA offers elegant solutions to many limitations in port knocking implementations. These allow SPA to solve the replay problem, achieve a data transmission rate that makes the use of asymmetric encryption possible, thwart simple spoofing attacks and remain under the radar of intrusion detection systems that are monitoring networks for port scans. In this paper[2], Michael Rash developed this system for Linux, Mac OS, FreeBSD, and OpenBSD. He designed for a single client, i.e., only a single client can send a SPA packet to SPA Server for authentication and access the Service in the default drop firewall policy without sharing Encryption keys between SPA Client and the SPA Server. Even he did not implement for accessing specific ports, i.e., any valid clients can access all the ports in the network.

3.2 ANALYSIS AND FINDINGS

1. He developed SPA Server for most operating systems except Windows because the Firewall is entirely different and more complex than other operating systems. So, in this research project, I developed an authentication protocol for the Windows system.

2. He developed SPA Server for most operating systems except Windows because the Firewall is entirely different and more complex than other operating systems. So, in this research project, I developed an authentication protocol for the Windows system.
3. He has not implemented any mechanism for sharing AES and HMAC keys for authentication and securely transmitting the SPA Packet to SPA Server. So, I implemented a Key Distribution Center (KDC) for sharing AES and HMAC keys using RSA public-private key mechanism.
4. He implemented for a single client, i.e., a single client can send a packet at a time to SPA Server for getting access to a service. However, I used multiple client mechanisms using the asynchronous method to access various services at a time.
5. He implemented that any authorized client could access all the ports running in the network. As a result, any specific port that the Administrator does not permit can also access that port.

4. PROPOSED WORK

4.1 DESIGN OF PROPOSED ARCHITECTURE

4.5.1 Flowchart

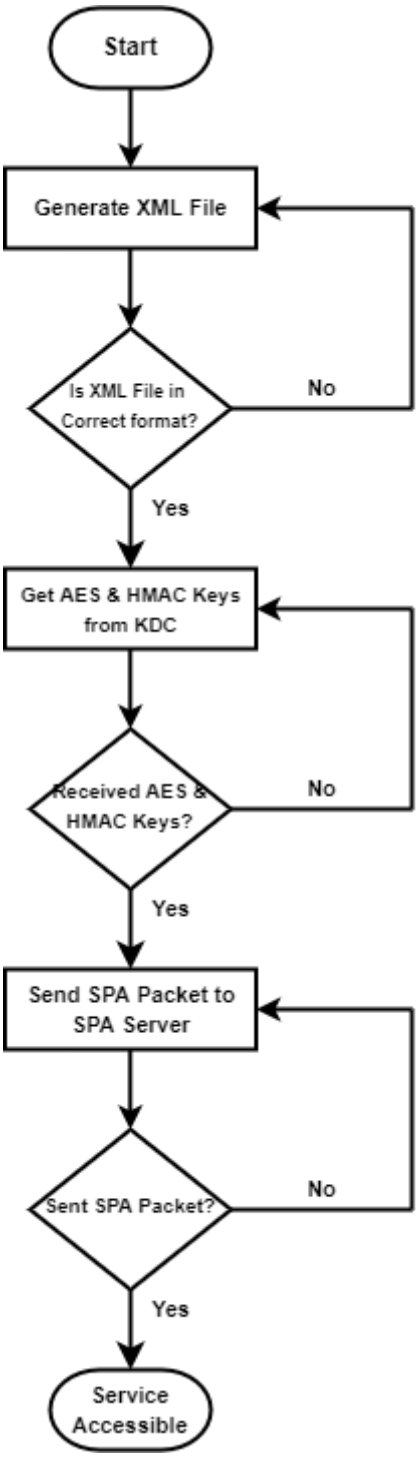


Figure 4.5.1.1 Client's Flowchart

4.5.2 Use Case

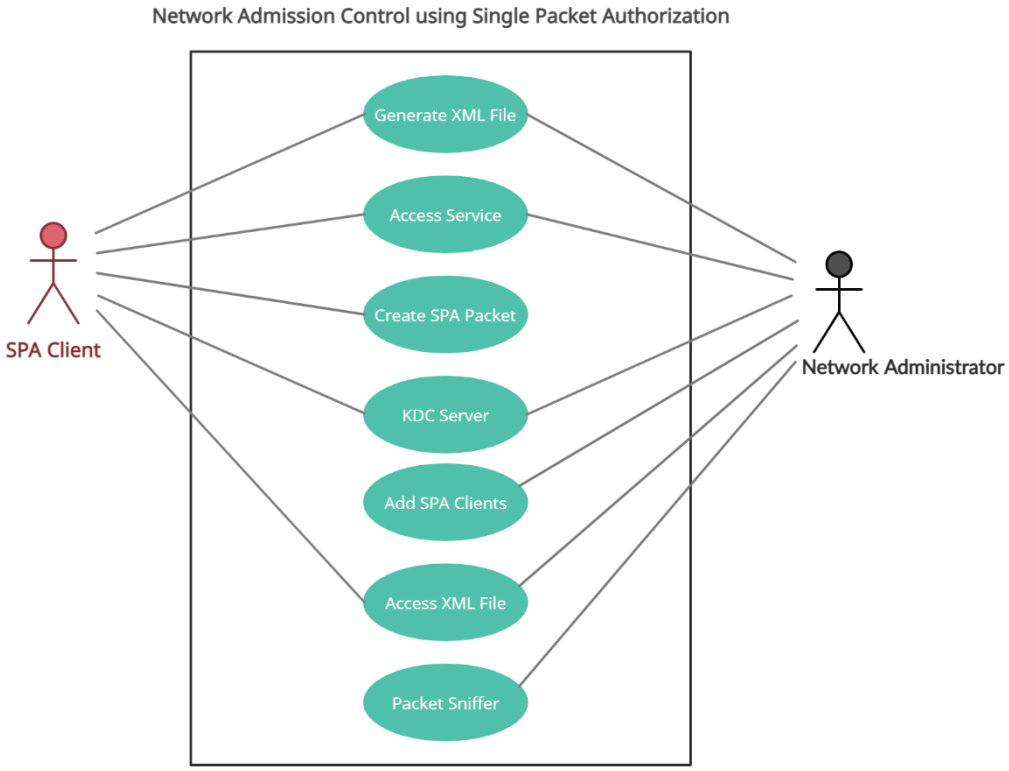


Figure 4.5.2.1 SPA's Use-case Diagram

4.5.3 Deployment Diagram

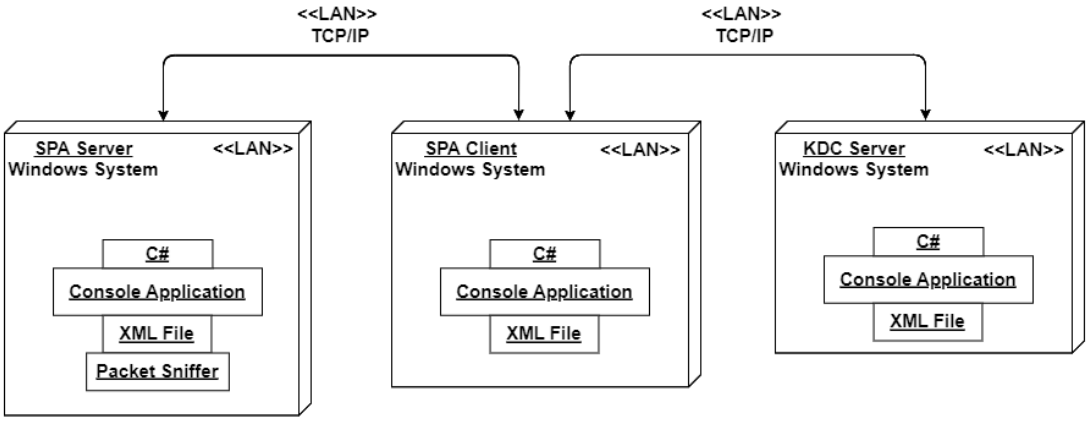


Figure 4.5.3.1 SPA's Deployment Diagram

4.5.4 Flow of the Architecture

ZTNA is an architecture that enables secure access to applications and services based on access control regulations. To implement this model, SDP is a computer security approach that conceals Internet-connected infrastructure such as servers, routers, and other company assets from being seen by outside parties and attackers. The first component needed to access is a resource (entity) or an application. The entity has no DNS entry, which is a significant security feature. So, it remains hidden and has no direct access. Instead, the Client is given an SDP-enabled means of communication to access that Service. This is an SDP Server (Packet Sniffer) that umpires connectivity between Client and Server. The Server contains a limited list of authorized users.

The Client first generates his configuration file in which all the necessary fields for generating the Packet are available. Then, it sends his RSA Public key and UUID to KDC, and it is stored by the Administrator while developing a list of authorized clients. Afterward, the Client requests AES and HMAC keys, and KDC sends them after verifying the Client's identity from the configuration file of the Server, which is shared by both KDC and the Server.

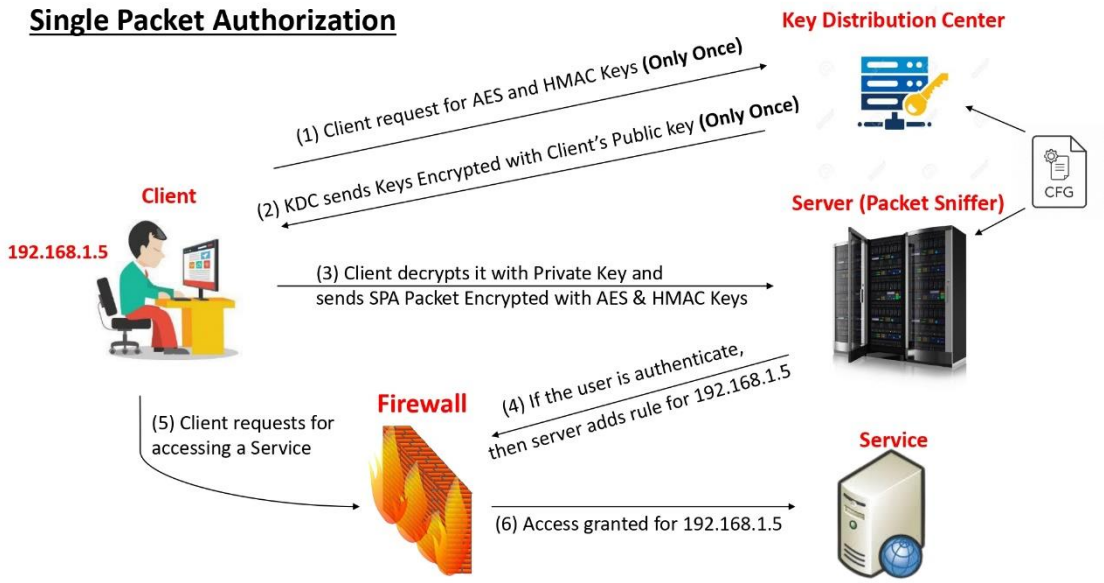


Figure 4.5.4.1 Architecture of SPA for Windows

After this, the Client sends a valid SPA Packet to the Server for accessing that specific Service, but because of the default drop policy, that Packet is discarded, and the Packet Sniffer captures it before the Firewall rejects it. SPA Server then verifies this Packet. If the Client is authenticated and has accessed that particular port, the Server then lets the gateway know that a client will send an authenticated communication and adds the inbound firewall rule for that specific Client with IP Address, Protocol, and Port number. Finally, the client access that Service and is allowed to access that Service only; the rest of the services in the network are blocked for him.

4.2 DESCRIPTION OF COMPONENTS OF THE SYSTEM

There are seven components in my research project.

1. **SPA Client:** The Client is any user who is permitted to access any service in an organization, school, and college. The Client can generate a valid packet to send it to SPA Server for authentication purposes and even access KDC to get AES and HMAC keys for secure communication and authentication purposes.

2. **KDC:** Key Distribution Center is the critical component of SPA; without that user will not be able to generate a SPA Packet. KDC shares AES and HMAC keys to a user to access a network service. KDC uses RSA Public-Private key cryptography to share secret keys with the Client.

3. **SPA Server:** The primary role of the SPA Server is to authenticate and authorize any user who has sent a SPA Packet. It checks which ports the user has access to, from checking into a configuration file, even checking the UUID of the user for two-factor authentication, and adding the inbound firewall rule for that particular Client.

4. **Packet Sniffer**: As SDP employs a Zero Trust architecture by default, even SPA Server would be running, but the Server's port number is also blocked in the Firewall. So, the role of Packet Sniffer is to capture the SPA Packet before Firewall blocks it and gets the information of all the layers of a packet, and sends it to the SPA Server for authentication. Therefore, my projects completely follow Zero Trust architecture.

5. **Firewall:** As mentioned above, there will be one inbound rule in the Windows Firewall that blocks all the Ports, Ips, and Protocols. So, after SPA Server authenticates any client, the Server will add one inbound rule for that particular Client.

6. **Service:** Service may be any port number running in the LAN that the Client wants access to.

7. **Configuration File (XML File):** This file is similar to Database. As my project is a console application, fewer clients will have access to the Service running in the LAN. So, using a Database is not a proper approach as retrieving a Client from the Database is a time-consuming process, and even it follows Default Drop Policy. Thus, this Configuration File stores all the Client's details, which both SPA Server and KDC share.

4.3 IMPLEMENTATION DETAILS

The whole project consists of SPA Client, SPA Server, and KDC, written in C# language. On the other hand, Packet Sniffer is in C++, which uses the PcapPlusPlus library.

1. **SPA Client:** Client can do three operations: create a configuration file, get AES and HMAC keys, and send the SPA Packet to the SPA Server. The Client creates a SPA Packet which AES Key encrypts for secure communication, and then the Hash Code is generated via HMAC key for authentication purposes.

2. **KDC:** KDC generates several clients with its details for authentication and authorization purposes and is stored in an XML file. Even it sends AES and HMAC keys to the Client when any client requests them.

3. **SPA Server:** The primary role of the SPA Server is to add an inbound firewall rule for a particular Client who requests to access the Service in the LAN. Even the Server has access to XML File which KDC generated.

4. **Packet Sniffer:** Packet Sniffer code generates a DLL file that SPA Server uses via the DllImport function in the C# code. As SDP employs Default Drop Policy. So, the Packet Sniffer sniffs the SPA Packet before the Firewall drops it and sends all the necessary details requested by SPA Server for authentication purposes because Packet Sniffer operates before the Network Interface Card.

4.4 FEATURES OF THE NEW SYSTEM

This authentication protocol gives new features compared to LAN, Static Ip, and Port Knocking. Following are the features of the new system:

1. **Device Verification:** Device identity is typically verified via a unique ID, i.e., UUID (Universally Unique Identifiers). UUID is automatically integrated into the SPA Packet, so the Client cannot change UUID in the Application Layer. Thus, SPA Server checks UUID and verifies the device's identity.

2. **User Identity Verification:** User identity is typically verified via a unique Client ID and HMAC key. The Client generates a Hash Code from Encrypted Payload and sends it to Server, and the Server generates the Hash Code on that Encrypted Payload and compares the Hash Code sent by the Client and generated Hash Code. Therefore, the User Identity is verified via unique ID and HMAC.

3. **SDP Controller (Server) Approval**: The SDP "controller" is the logical component of the SDP that is responsible for determining which devices and servers should be allowed to connect. Once the user and device are authenticated, the controller passes along the approval of the user and device to the SDP Gateway (Firewall). The SDP Gateway is where access is actually allowed or denied.

4. **Client Access:** The Client is able to access previously hidden network resources and can continue using their device like normal. The user operates within an encrypted network to which only they and the services they access belong. Moreover, access would only be granted for the IP address of the Client, which is encrypted within the SPA message and hence not available to the attacker.
5. **Least Privilege Access:** This principle means any users will grant the minimum level of access or permissions to carry out their job successfully and not more than that. So the user will not have access to the rest of the resources, and it will prevent the exposure of sensitive data and assets to everyone in the network. Not only does this principle apply to humans, but it also extends to non-human tools such as applications and connected devices that need access permissions to perform specific tasks.

6. **Micro-segmentation:** Microsegmentation is another critical principle imperative to the ZTNA model. A security approach divides the network into multiple zones, defining fine-grained and flexible security policies for distinct network segments. As a result, it mitigates the attackers' lateral movements within the network even after gaining access inside the network perimeter because of the different security policies in place in various segments; the infiltration into every component is difficult.

4.5 HARDWARE AND SOFTWARE REQUIREMENTS

4.5.1 Hardware Requirements

- Windows Device for generating SPA Packet by a Client
- Windows Device for KDC Server
- Windows Device for SPA Server
- Windows Firewall
- Device for running Application or Service

4.5.2 Software Requirements

- Visual Studio versions 2017, 2019, and 2022 with .NET Framework
- C# Console Application or Command Prompt
- C#, C++ language installed on the device
- Windows Firewall

4.6 ASSUMPTIONS AND DEPENDENCIES

4.6.1 Assumptions

- The Client should generate a valid SPA Packet and send it to the proper destination, i.e., the IP and Port number of KDC and SPA Server.
- The Client should be in the Local Area Network, and his details should be in the Server's XML File.

4.6.2 Dependencies

- XML File of Server should be shared between KDC and SPA Server.
- The Server and KDC should be in the organization's network.

5. EXPERIMENTS AND RESULTS

5.1 DATASET

My research project is an authentication protocol to access a desired service in the LAN. So, there are no Database requirements and should not be dependent on the Database as it is more time-consuming to fetch the Client's details. Furthermore, even as the SDP model employs ZTNA, so, to get details from Databases like MySQL and MongoDB is impossible. So, using a Database is not a proper approach. Configuration File (XML File) works similar to Database. As my project is a console application, fewer clients will have access to the Service running in the LAN. Thus, this Configuration File stores all the Client's details, which both SPA Server and KDC share. Moreover, the Client has its XML File for storing details of his own, KDC, and SPA Server.

Figure 5.1.1 XML File of a Client

Nine fields are used in the Client's XML File to generate a SPA Packet. Client ID is used for uniquely identifying the Client in the Database, UUID for two-factor authentication, KDC Ip And Port number, SPA Server Ip and Port number, AES key for secure transmitting a Packet, HMAC for generating a Hash Code for authentication, and RSA Private-Public key for acquiring AES and HMAC keys from KDC.

Figure 5.1.2 XML File of a Server

Similarly, several numbers of the Client's details will be stored in Server's XML file, i.e., six fields are used to authenticate a SPA Packet. Client ID is used for uniquely identifying the Client in the Database, UUID for two-factor authentication, list of port numbers to which Client has access, AES key for secure receiving a Packet, HMAC for generating a Hash Code for authentication, and RSA Private-Public key for acquiring AES and HMAC keys from KDC.

5.2 RESULTS

The results of my research project are that after eliminating all the limitations and weaknesses of the current system, SPA serves as a complete replacement for that architecture. Due to the access of all the ports in the Local Area Network, Software Defined Perimeter model acts as a new modern security approach to overcome all the drawbacks of the current system. The below picture shows the existing architecture.

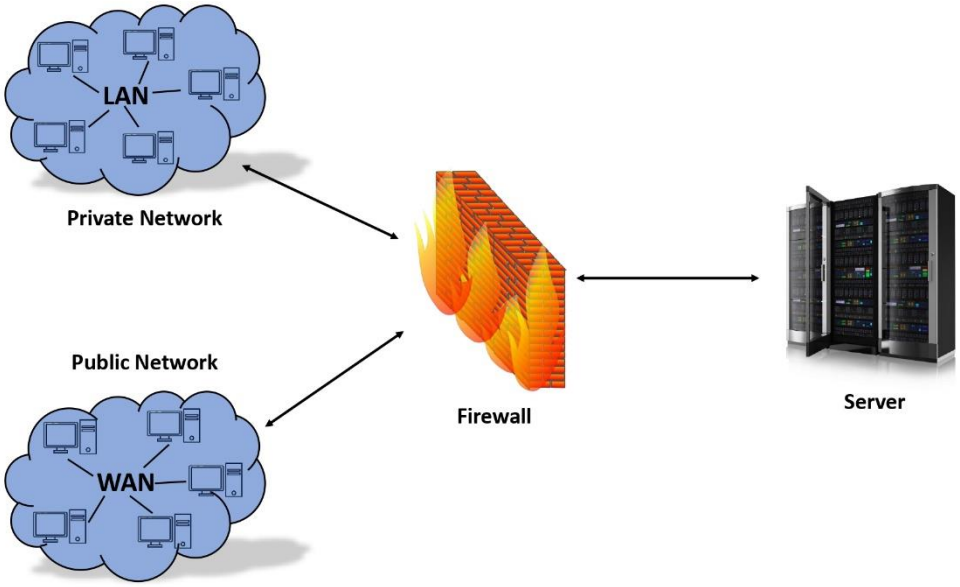


Figure 5.2.1 Current Architecture of an organization

To solve all these issues, we have to add one controller between any Server and Firewall, freeing the system from attackers. The Client in the LAN will not be able to access all the services running because Firewall blocks all the ports. So, the controller verifies the Packet and adds the Inbound Firewall policy for that particular Client at runtime. Thus, we do not have to add the Firewall rule manually even if IP Address changes with time, and we can make several combinations of the Client and his desired Service to access.

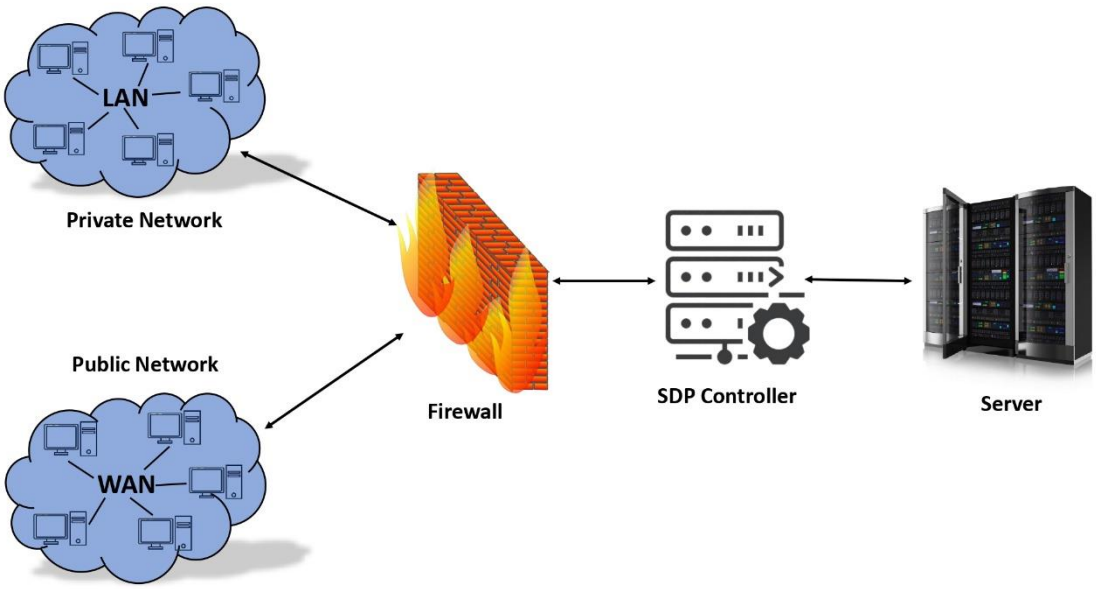


Figure 5.2.2 Modified Architecture of an organization

6. USER MANUAL

1. Client creates a Configuration (XML) File

```
D:\SPA_Client>Client.exe
(1) Send SPA Packet
(2) Generate Conf File
(3) Get Aes and Hmac Keys
Enter your choice: 2
Enter following Info to make your Conf File:
Enter your Id: sac_4311
Enter IP Add of KDC: 192.168.244.40
Enter Port No of KDC: 8000
Enter IP Add of SPA: 192.168.244.40
Enter Port No of SPA: 9000
```

Figure 6.1.1 Creating a Config file

2. Configuration file of Client without AES and HMAC Keys

```
▼<Client>
   <Client_Id>sac_4311</Client_Id>
   <UUID>41B02D90-82AE-E911-8102-F8B46A70C1DB</UUID>
   <Kdc_Ip>192.168.244.40</Kdc_Ip>
   <Kdc_Port>8000</Kdc_Port>
   <Spa_Ip>192.168.244.40</Spa_Ip>
   <Spa_Port>9000</Spa_Port>
   <Aes_Key> </Aes_Key>
   <Hmac_Key> </Hmac_Key>
   <Rsa_Key><RSAKeyValue>
   <Modulus>vEGlOWu2DW5UrZjccfIen6fdY+j6MdnXggqT70j55Qscguc0W6mN
   </Modulus> <Exponent>AQAB</Exponent>
   <P>4h5bXpyzaCpQgbznOKRQgWrEsD19VTW4SYnpCFibGK1SMhO0xM6+N4clTq
   <Q>1SJnN9+UHej0C5PlSvWeyuBNgFMNFmYfU4XyG9Q0GC4e2vGPlPVAXTbftE
   <DP>i5foGmmtDFKLLvz61eoC1VKfqJndZRxr7+JW3dMttLdTdhzaTWHA5rDlJ
   <DQ>kK8HFMXg3aELMmTWrdkpT52O3wMCCmeO8nsmTL/0EUoVxhlfmAYZCG481
   <InverseQ>K7p3DQGFwE7Ld+KsLkrbOta1y6yeukhVlLq2mzGTwAU77SPdKw4
   <D>qODevkcQ1rWeAuj5AMLpk+G7qg/queOo/0BL+0Fk44J3Pjr4bi4JIRl+BJ
   </D> </RSAKeyValue></Rsa_Key>
 </Client>
```

Figure 6.2.1 Config file of Client

3. Administrator adds Client's details in his Configuration File

```
D:\KDC>Kdc.exe
Do you want to run KDC Server or to add Clients in Configuration file?
(1) Run KDC Server
(2) Add Clients in Configuration file
Enter your choice: 2
Enter Number of Clients: 1
Client - 1:
Enter Client Id: sac_4311
Enter UUID: 41B02D90-82AE-E911-8102-F8B46A70C1DB
Enter the Number of Ports (Seperate by ','): 22,21,643,80,8500
Enter Modulus and Exponent of RSA key in XML Format: <Modulus>vEGlOWu2DW5UrZjccfIen6fdY+j6MdnXggqT70j55Q
scguc0W6mNpAmgy6jZRODNCXLMemDxlbWPDBDUgbL/gZKmgS+h8SZ9UOyDEbxLOe6AkGnNXtkp2RT+VS3QYEjwBl7ZUZImaWCPcNjHPU
F+HamT0kOgp5xSgP45+MsFv9yIubUhUiDVFkw+bPq5SrkmnL5XvYoW21GQg05llAupMRN6IPd9s+RfUkktBPLOnZeGLe96hbFsJ7kZtb
Lq4nD3PpxfwRPxUDxpwPweSvx4dV50JrvODAgP2yS52OHyOPfj27FcGaUe5v5hpJvrKMRzX7ImTZR7+DyXSeP0XH8NmQ==</Modulus>
<Exponent>AQAB</Exponent>
```

Figure 6.3.1 Adding Client's detail in a config file

4. Configuration file of Administrator

Figure 6.4.1 Configuration file of Administrator

5. Client request for AES and HMAC Keys

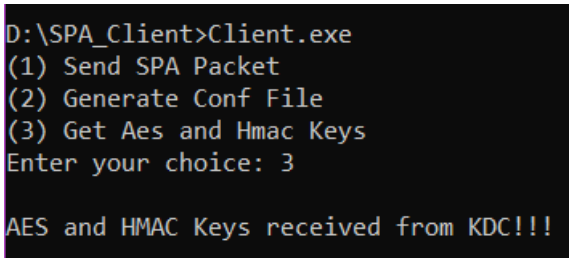


Figure 6.5.1 Client request for the Keys

6. KDC sends AES and HMAC Keys to the Client

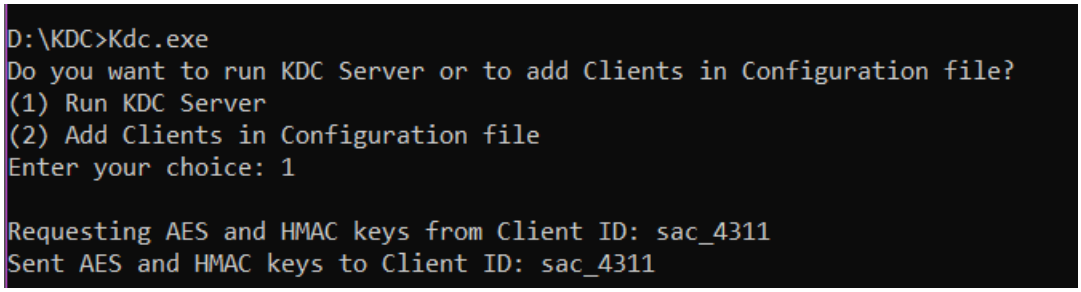
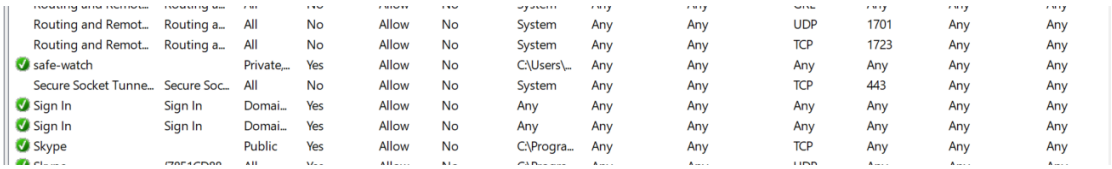


Figure 6.6.1 KDC sends both the Keys

7. Configuration file of Client after receiving both Keys

Figure 6.7.1 Config file of Client after receiving the Keys

8. Windows Firewall without the rule of SPA Client



Routing and Remot...	Routing a...	All	No	Allow	No	System	Any	Any	UDP	1701	Any	Any
Routing and Remot...	Routing a...	All	No	Allow	No	System	Any	Any	TCP	1723	Any	Any
safe-watch		Private,...	Yes	Allow	No	C:\Users\...	Any	Any	Any	Any	Any	Any
Secure Socket Tunne...	Secure Soc...	All	No	Allow	No	System	Any	Any	TCP	443	Any	Any
Sign In	Sign In	Domai...	Yes	Allow	No	Any	Any	Any	Any	Any	Any	Any
Sign In	Sign In	Domai...	Yes	Allow	No	Any	Any	Any	Any	Any	Any	Any
Skype		Public	Yes	Allow	No	C:\Progra...	Any	Any	TCP	Any	Any	Any

Figure 6.8.1 Windows Firewall without the rule of SPA Client

9. Client sends SPA Packet to SPA Server (with an error such as Client ID is not present in Config file, UUID of Client is different)

```
D:\SPA_Client>Client.exe
(1) Send SPA Packet
(2) Generate Conf File
(3) Get Aes and Hmac Keys
Enter your choice: 1
Enter following Info to build a SPA Packet:
Enter Allow Ip Address: 192.168.1.25
Enter Access Protocol: tcp
Enter Access Port: 443

---Sending the SPA Packet of 229 Bytes to 127.0.0.1 on 9000

D:\SPA_Client>Client.exe
(1) Send SPA Packet
(2) Generate Conf File
(3) Get Aes and Hmac Keys
Enter your choice: 1
Enter following Info to build a SPA Packet:
Enter Allow Ip Address: 192.168.1.25
Enter Access Protocol: tcp
Enter Access Port: 22

---Sending the SPA Packet of 229 Bytes to 127.0.0.1 on 9000

D:\SPA_Client>Client.exe
(1) Send SPA Packet
(2) Generate Conf File
(3) Get Aes and Hmac Keys
Enter your choice: 1
Enter following Info to build a SPA Packet:
Enter Allow Ip Address: 192.168.1.25
Enter Access Protocol: tcp
Enter Access Port: 22

---Sending the SPA Packet of 229 Bytes to 127.0.0.1 on 9000
```

Figure 6.9.1 Client sends SPA Packet

10. SPA Server receives SPA Packet (with an error such as Client ID is not present in Config file, UUID of Client is different)

```
D:\SPA_Server>Server.exe
---Packet Sniffer is waiting to receive SPA Packet from Client...

---Received SPA Packet from sac_4311

Error from sac_4311: sac_4311 does not have permission to access the Port Number = 443

---Received SPA Packet from sac_4311

Error from sac_4311: UUID does not match!!! Another Client is sending the Packet of sac_4311

---Received SPA Packet from sac_4311
RandomValue : 0555157280
UserName : theSHYAM
UUID : 41B02D90-82AE-E911-8102-F8B46A70C1DB
Allow Ip : 192.168.1.25
Access Protocol : tcp
Access Port : 22

---FireWall Rules
03-04-22 02:48:12 PM spaserver: SPA Packet from IP: 192.168.1.25 received with access source match
03-04-22 02:48:12 PM spaserver: Added Rule for sac_4311 to INBOUND RULE with 192.168.1.25, tcp/22
```

Figure 6.10.1 SPA Server receives SPA Packet

11. Windows Firewall with the rule of SPA Client

Routing and Remot...	Routing a...	All	No	Allow	No	System	Any	Any	GRE	Any	Any	Any
Routing and Remot...	Routing a...	All	No	Allow	No	System	Any	Any	UDP	1701	Any	Any
Routing and Remot...	Routing a...	All	No	Allow	No	System	Any	Any	TCP	1723	Any	Any
sac_4311		All	Yes	Allow	No	Any	Any	192.168.1.25	TCP	Any	22	Any
safe-watch		Private,...	Yes	Allow	No	C:\Users\...	Any	Any	Any	Any	Any	Any
Secure Socket Tunne...	Secure Soc...	All	No	Allow	No	System	Any	Any	TCP	443	Any	Any
Sign In	Sign In	Domai...	Yes	Allow	No	Any	Any	Any	Any	Any	Any	Any

Figure 6.11.1 Windows Firewall with the rule of SPA Client

12. Client can get detailed information of SPA Packet using --verbose argument

```
D:\SPA_Client>Client.exe --verbose
(1) Send SPA Packet
(2) Generate Conf File
(3) Get Aes and Hmac Keys
Enter your choice: 1
Enter following Info to build a SPA Packet:
Enter Allow Ip Address: 192.168.1.25
Enter Access Protocol: tcp
Enter Access Port: 22

---Field Values
RandomValue: 0380810875
Username: theSHYAM
UUID: 41B02D90-82AE-E911-8102-F8B46A70C1DB
Allow Ip: 192.168.1.25
Access Protocol: tcp
Access Port: 22
Encryption Type: AES
AES Type: AES256
Authentication Type: HMAC
HMAC Type: SHA256
Encryption Mode: CBC

---Plain Text
PlainText: 0380810875:theSHYAM:41B02D90-82AE-E911-8102-F8B46A70C1DB:192.168.1.25:tcp:22

---Generating SPA Packet
Protocol: UDP
Source Port: <OS assigned>
Server IP: 192.168.244.40
Server Port: 9000

---Authentication Fields
HMAC Key = jeQt1o/YSzf5OSkRbE8werXn+2vZIX0vyrwqVy00jwxTtCHVDz2t6rGTqM9y0VA76ditjuZZbyzC9Cs25xGJYQ==
Hash Code = iIvku4J6dQQ+nnVs3+oCHeVST55x102wkGy6+P18pPA=
AES Key = 3QrmVGRnwAjgEp3ODf5JqAkemjboitHpy7v58LYrl1c=
IV = h9oY/DY9UMEqoaniEtr6JA==
ENC + IV = UkFudmNMMi9jY2xBUUozcGRYRlVQQURiYUErWFhISEw3ZUNPZGF2Z0hBL3hzZnBQaEdUaDBVbkRKT0hvRHltRktVe
ENC + IV + HashCode = UkFudmNMMi9jY2xBUUozcGRYRlVQQURiYUErWFhISEw3ZUNPZGF2Z0hBL3hzZnBQaEdUaDBVbkRKT6
nVs3+oCHeVST55x102wkGy6+P18pPA=

---Final Packed/Encrypted/Encoded Data:
UkFudmNMMi9jY2xBUUozcGRYRlVQQURiYUErWFhISEw3ZUNPZGF2Z0hBL3hzZnBQaEdUaDBVbkRKT0hvRHltRktVeW9wZzVpMzVp
6+P18pPA=


---Sending the SPA Packet of 229 Bytes to 192.168.244.40 on 9000
```

Figure 6.12.1 Detailed information using --verbose command

7. LIMITATIONS AND FUTURE ENHANCEMENT

7.1 LIMITATIONS

- This project is limited to console applications only. So, the user does not have any graphical user interface to experience it.
- Authentication is only based on user and device identity.
- It only runs on Visual Studio Version 2017, 2019, and 2022. Thus, the Administrator cannot use an older version of Visual Studio to alternate changes.
- Client has to create a SPA Packet every time it needs to access a request.
- The network administrator has to delete all the rules manually in the Firewall after the Client's Packet is expired.

7.2 FUTURE ENHANCEMENT

- This project can be converted into a Web Application or Android Application for a better user experience.
- More Authentication methods, such as identity-based on location and token, can be integrated for better security.
- We can create this protocol for all the older Visual Studio Version.
- SPA Packet can be saved in the Application, so the Client does not have to send a Packet every time to access the same Service.
- Deletion of Inbound Firewall rules can be made automatic after the rule is expired. Therefore, the network administrator must not delete all the rules manually in Firewall.

8. CONCLUSION AND DISCUSSION

8.1 CONCLUSION

Single Packet Authorization provides similar security benefits to port knocking in terms of protecting services with a packet filter that is configured in a default-drop stance. Anyone scanning for a target service that is protected in this way will be unable to detect such a service is listening, which makes even the exploitation of zero-day vulnerabilities much more difficult. SPA offers elegant solutions to many limitations in port knocking implementations. These allow SPA to solve the replay problem, achieve a data transmission rate that makes the use of asymmetric encryption possible, thwart simple spoofing attacks and remain under the radar of intrusion detection systems that are monitoring networks for port scans. Its data transmission capabilities, coupled with its clean strategy for preventing replay attacks, make it an ideal candidate for expanding the configuration of packet filters to drop all connections to some critical services by default. This makes it much more difficult because an arbitrary IP address cannot enumerate or interact with these services until a valid SPA message is generated.

8.2 DISCUSSION

8.2.1 Problems Encountered and Possible Solutions

- Overall, before developing the project, while looking at the functionality – the project seems to be very easy to implement. I thought that I'll need to write the code now. But as I progressed in the project, I faced many complex difficulties.
- It was not easy to understand the flow of SPA protocol because there are fewer resources available for SPA protocol, and projects related to it are fewer than other field projects.
- Working with .NET Framework was the first time for me, and it was quite tricky, but with the help of my mentor and other senior colleagues of the company, I made it through and completed my work.
- Different project components like SPA Server, KDC, and Packet Sniffer were challenging to build. However, Packet Sniffer was the most challenging part of developing because setting the environment of Packet Sniffer using the PcapPlusPlus library was quite different from others, and I tried three-four libraries for it.

8.2.2 Summary

This project aimed to show how SDP can enhance an organization's network security, such as ISRO and IPR. It was an experimental research project that looked at how we can add one more layer to current network architecture and modify it, so it can provide better security to the network administrator and overcome all the difficulties they have faced like all the clients in the LAN can access all the services, static IP in the Firewall. Thus, it was a pretty different experience for me to develop this project.

9. REFERENCES

[1] Michael Rash: Single Packet Authorization, April 1, 2007
https://www.linuxjournal.com/article/9565
[2] Michael Rash: Single Packet Authorization with Fwknop, December 2005
https://www.cipherdyne.org/fwknop/docs/SPA.html
[3] Krzywinski, M. 2003. Port Knocking: Network Authentication Across Closed Ports. SysAdmin Magazine 12: 12-17.
http://mkweb.bcgsc.ca/portknocking/view/
[4] Michael Rash's Website for all the concepts related to SPA:
https://www.cipherdyne.org/
[5] Michael Rash Fwknop SPA's Linux Version:
https://www.cipherdyne.org/fwknop/docs/fwknop-tutorial.html
[6] All concepts related to C# Language for developing this project:
https://docs.microsoft.com/en-gb/
[7] Packet Sniffer Library:
https://pcapplusplus.github.io/