**ASSIGNMENT - CS 3052**

**DESIGN AND IMPLEMENT A FIREWALL**

**T.K.P.PERERA**

**170446L**

**Code**

*Github Link: https://github.com/kavindaperera/computer-security-firewall.git*

import ipaddress

import json

#Considering the two networks 192.168.1.0 and 10.10.10.0

#interface\_1 => 192.168.1.0

#interface\_2 => 10.10.10.0

# creating a dictionary from ip datagram

def analyse\_datagram(datagram\_header):

    protocol\_dict = {1: 'ICMP', 2: 'IGMP', 6: 'TCP', 17: 'UDP'}

    version\_bin = datagram\_header[0:4]

    ihl\_bin = datagram\_header[4:8]

    total\_length\_bin = datagram\_header[16:32]

    #protocol (TCP/UDP)

    protocol\_bin = datagram\_header[72:80]

    #source address

    saddress\_bin = datagram\_header[96:128]

    #destination address

    daddress\_bin = datagram\_header[128:160]

    #Internet Header Length

    ihl = int(ihl\_bin, 2) \* 32

    protocol = int(protocol\_bin, 2)

    protocol = protocol\_dict.get(protocol)

    payload = datagram\_header[ihl:]

    #getting the dot notations for addresses

    saddress = ipaddress.ip\_address(

        int('.'.join(str(int(x, 2)) for x in saddress\_bin.split())))

    daddress = ipaddress.ip\_address(

        int('.'.join(str(int(x, 2)) for x in daddress\_bin.split())))

    sport = int(payload[:16], 2)

    dport = int(payload[16:32], 2)

    headers = {'saddress': saddress, 'daddress': daddress,

               'sport': sport, 'dport': dport, 'protocol': protocol}

    return (headers)

# ip firewall filter

def filter(headers, interface):

    saddress = str(headers.get("saddress")).split(".")

    daddress = str(headers.get("daddress")).split(".")

    sport = str(headers.get("sport"))

    dport = str(headers.get("dport"))

    protocol = headers.get("protocol")

    print(headers)

    with open('config.json') as f:

        rules = json.load(f)

    for key in rules:

        if (rules[key][0].get("interface") == interface):

            if (rules[key][0].get("saddress") != 'any'):

                r\_saddress = rules[key][0].get("saddress").split('.')

                if (r\_saddress[0] == saddress[0] and r\_saddress[1] == saddress[1] and r\_saddress[2] == saddress[2]):

                    if (rules[key][0].get("daddress") != 'any'):

                        r\_daddress = rules[key][0].get("daddress").split('.')

                        if (r\_daddress[0] == daddress[0] and r\_daddress[1] == daddress[1] and r\_daddress[2] == daddress[2]):

                            if (rules[key][0].get("dport")!= 'any'):

                                r\_dport = rules[key][0].get("dport")

                                if (int(dport)==int(r\_dport)):

                                    print(interface)

                                    print(rules[key][0].get("action"))

                                    break

                                else:

                                    continue

                            else:

                                print(interface)

                                print(rules[key][0].get("action"))

                                break

                        else:

                            continue

                    else:

                        print(interface)

                        print(rules[key][0].get("action"))

                        break

                else:

                    continue

            else:

                print(interface)

                print(rules[key][0].get("action"))

                break

        else:

            continue

def firewall(interface):

    with open(interface+'.json') as f:

        tests = json.load(f)

    for key in tests:

        headers = analyse\_datagram(tests[key])

        filter(headers, interface)

        print('=============================================================================')

firewall('interface\_1')

firewall('interface\_2')

**Firewall filtering Rules**

{

    "1": [

        {

            "saddress": "192.168.1.0",

            "daddress": "10.10.10.0",

            "sport": "any",

            "dport": "43",

            "interface": "interface\_1",

            "action": "ACCEPT"

        }

    ],

    "2": [

        {

            "saddress": "192.168.1.0",

            "daddress": "10.10.10.0",

            "sport": "any",

            "dport": "80",

            "interface": "interface\_1",

            "action": "ACCEPT"

        }

    ],

    "3": [

        {

            "saddress": "192.168.1.0",

            "daddress": "10.10.10.0",

            "sport": "any",

            "dport": "any",

            "interface": "interface\_1",

            "action": "DENY"

        }

    ],

    "4": [

        {

            "saddress": "10.10.10.0",

            "daddress": "192.168.1.0",

            "sport": "any",

            "dport": "43",

            "interface": "interface\_2",

            "action": "ACCEPT"

        }

    ],

    "5": [

        {

            "saddress": "10.10.10.0",

            "daddress": "192.168.1.0",

            "sport": "any",

            "dport": "80",

            "interface": "interface\_2",

            "action": "ACCEPT"

        }

    ],

    "6": [

        {

            "saddress": "10.10.10.0",

            "daddress": "192.168.1.0",

            "sport": "any",

            "dport": "any",

            "interface": "interface\_2",

            "action": "DENY"

        }

    ],

    "7": [

        {

            "saddress": "any",

            "daddress": "any",

            "sport": "any",

            "dport": "any",

            "interface": "interface\_1",

            "action": "DROP"

        }

    ],

    "8": [

        {

            "saddress": "any",

            "daddress": "any",

            "sport": "any",

            "dport": "any",

            "interface": "interface\_2",

            "action": "DROP"

        }

    ]

}

**Interface 1 – Test Datagrams**

{

    "datagram\_1": "00100101000000000000000000000000000000000000000000000000000000000000000000000110000000000000000011000000101010000000000100000001000010100000101000001010000000010000000001010000000000000010101100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

    "datagram\_2": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000010101100000100000000000100000001000010100000101000001010000000010000000110010011000000000101000000000000000000000000000000000000",

    "datagram\_3": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000010101100000100000000000100000001000010100000101000001010000000010000000110010011000000000010101100000000000000000000000000000000",

    "datagram\_4": "00100101000000000000000000000000000000000000000000000000000000000000000000000110000000000000000011000000101010000000000100000010000010100000101000001010000000100000000001010000000000000010101100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

    "datagram\_5": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000011000000101010000000000100000011000010100000101000001010000010100000000000101011000000000101000000000000000000000000000000000000",

    "datagram\_6": "00100101000000000000000000000000000000000000000000000000000000000000000000000110000000000000000011000000101010000000000100000011000010100000101000001010000000010000000001010000000000011001001100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000"

}

**Interface 2 – Test Datagrams**

{

    "datagram\_1": "00100101000000000000000000000000000000000000000000000000000000000000000000000110000000000000000000001010000010100000101000000010110000001010100000000001000000010000000110010011000000000101000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

    "datagram\_2": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000010101100000100000000000100000001110000001010100000000001000000010000000001010000000000000010101100000000000000000000000000000000",

    "datagram\_3": "00100101000000000000000000000000000000000000000000000000000000000000000000000110000000000000000000001010000010100000101000001010110000001010100000000001000000100000000000100000000000011001001100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

    "datagram\_4": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000000001010000010100000101000000010110000001010100000000001000000100000000110010011000000000010101100000000000000000000000000000000",

    "datagram\_5": "00100101000000000000000000000000000000000000000000000000000000000000000000010001000000000000000010101100000100000000000100000010110000001010100000000001000000010000000001010000000000000010101100000000000000000000000000000000"

}