This file contains information about how the firewall works and behaves.

The firewall comprises of :-

* Database
* Program

On the terminal, open prolog.

Load the code file -

?- [code].

This loads the database as well as the code file on prolog.

The database is where rules are stored which define how the firewall responds to different incoming packet in the following ways :-

* It can **allow** the packet to pass through. This happens when an **allow** rule is present in the database to which each and every clause of the incoming packet corresponds and print the message “Packet accepted!”.
* It can **reject** the packet when the clauses of the incoming packet correspond to a mentioned **reject** rule in the database, printing the message “Packet rejected!”
* It can **drop** the packet silently when the clauses of the packet match with a **drop** rule in the database without printing anything.
* It is assumed that our firewall is not meant to consider packets with clauses other than those mentioned in any of the rules, and such packets are **rejected** by default.

The following are the clauses being handled by our program :-

1. Adapter clause – The assumed supported adapter clauses are denoted by **capital** characters - ‘**A**’ or ‘**C**’ for example. The keyword ‘**any**’ should be passed as the adapter clause if the packet is meant to be supplied without an adapter clause.
2. Ethernet clause - Specifies either a **network** **protocol** **type** or **virtual** **LAN** (**VLAN**) **identifier** to match the 802.1 frame. The clause is supposed to be passed as a **list with 2 elements** as decimal numbers, with the **first** one specifying the **protocol** type and **second** one for **VLAN** identifer. Pass the keyword ‘**null**’ if any or both the clauses are missing in the packet.
3. IP clause – Specifies IPv4 addresses of the **source** and the **destination** as the **first** and **second** items of a list, and **IP** **protocol** **number** as the **third** argument. Pass the keyword ‘**null**’ in place of any of the missing sub-clauses of the IP clause.
4. ICMP conditions – Specify ICMP conditions in decimal in the form of a 2 element list, first one for the type and second one for the code. Pass the keyword ‘null’ iin place of any of the missing elements.
5. TCP/UDP conditions – Specify TCP and UDP port numbers in decimal. The correct format is again as a list of 3 elements, first one for source port, second for the destination port and third one for specifying the type of condition - ‘tcp’ for TCP and ‘udp’ for UDP conditions. The firewall is designed to operate even if the source or destination is missing in the packet if ‘null’ is used in its place.

There is a way specific of writing rules in the database so that it responds correctly to the incoming packet. The correct way to specify rules in the database Prolog program is -

<type of response> **(<**adapter clause>, <ethernet clause>, <IP clause>, <ICMP condition>, <TCP/UDP condition>**).**

* Types of responses –
  + allow
  + reject
  + drop

Expressions describe the values mentioned in the rules for clauses to match with. The firewall supports writing expressions in the following forms -

* <value>
  + Eg. ‘A’ for adapter clause or ‘192.168.1.1’ for IP source address
* <value> hyphen <value>, which means that all the values in between (extremes included) are covered in the expressions
  + Eg. ‘2-5’ for ethernet protocol type or ‘192.168.1.1-10’ for IP destination address.
* <value> comma <value>., which means that both these values are covered in the expressions.
  + Eg. ‘L, M’ for adapter clause and ‘5, 7’ for ethernal VLAN identifier.

The firewall is also designed to handle the input of the clauses to be correct, as per the [Internet Assigned Numbers Authority](https://www.iana.org/). Some examples are -

* The IPv4 address as n.n.n.n must have n in the range of 0-255, incorporated to be checked in the code.
* The TCP/UDP port values to be in the range 0-65535.
* The ethernet protocol ID to be in the range 0-255.

Here are some examples of complete rules -

allow('any', ['6-8', '40'], ['172.12.3.4', '182.168.3.0-182.168.3.255', '30'], ['2-3', '30'], [‘22-25’, ‘4’, ‘tcp’]).

reject('L', ['3-10', '56'], ['172.43.45.9,172.43.45.10', '182.168.3.0-182.168.3.255', '30'], ['2-3', '30'], [2, 25, 16, ‘tcp’]).

drop('Q-S', ['1-5', '48'], ['null', '182.168.3.0-182.168.3.255', '30'], ['2-3', '30'], [15, ‘null’, ‘udp’]).

Here are some examples of how to pass a packet -

packet('any', [6, 40], ['172.12.3.4', '182.168.3.1', '30'], [3, 30], [25, 4, ‘tcp’]).

packet('L', [7, 'null'], ['172.43.45.9', '182.168.3.0', '30'], [2, 65], [25, 16, ‘udp’]).