***README***

***Rule Syntax-***

1. For adapter:

accept clause|deny clause (where clause is of the form adapter <adapter-id>)

1. For ether

Accept clause |deny clause(where clause is of the form ether vid <vlan-number> proto <protocol-id>)

1. For IPV4

accept clause|deny clause (where clause is of the form ipv4 src addr <ipv4-addr> dst addr <ipv4-addr> proto <protocol-type> )

1. For IPV6

accept clause|deny clause (where clause is of the form ipv6 src addr <ipv6-addr> dst addr <ipv6-addr> proto <protocol-type>)

1. For TCP

accept clause|deny clause (where clause is of the form tcp dst port <tcp-udp-port> src port <tcp-udp-port>)

1. For UDP

accept clause|deny clause (where clause is of the form udp dst port <tcp-udp-port> src port <tcp-udp-port>)

1. For ICMP

accept clause|deny clause (where clause is of the form icmp type <protocol-type> code <message-code>)

1. For ICMPV6

accept clause|deny clause (where clause is of the form icmpv6 type <protocol-type> code <message-code>)

ENTER

Remark- In the packet passed, if the user does not provide a specific constraint then any should be passes.

**Assumption-In rule constraints of the form A-D OR 3-999 etc. , upper bound is not inclusive**.

***Packet Syntax-***

Adapter\_id

ether\_vid;ether\_proto

ipv4\_src\_addrs;ipv4\_dest\_addrs;ipv4\_proto

ipv6\_src\_addrs;ipv6\_dest\_addrs;ipv6\_proto

tcp\_dest\_port;tcp\_dest\_port

udp\_dest\_port;udp\_dest\_port

icmp\_code;icmp\_type

icmpv6\_code;icmpv6\_type

*(each of the above given rules are provided in a new line in the order specified)*

***Assumption- The user passes all the above mentioned inputs in the packet.***

***Returns***- **True + “Accepted”** if accepted, **True + “Denied”** if denied **True**  nothing if the packet is dropped. True indicates that the program is running .

***Input*** : Input for rules and packet taken from respective files where their complete address needs to be mentioned in the code file.

***How to run program:***  After changing the file path in the code run **firewall\_test** on the command prompt.

***Some points to be considered :***

1. Use fresh text files each time with no extra keystrokes . (due to use of \n as delimiters etc.)
2. Ignore warnings on command prompt indicating singleton variables.
3. Deny call in a rule denies the package even if one of the parameters is matched (in case of multiple parameters).

***Recursion :***

Firewall\_test calls the recursive function firewall which splits the given rule set into a list of rules through recursion and the final result is printed considering the conjunction of result of all the given rules . Even if one of the rules is not satisfied by the package the package is denied.

***Code Structure-***

* List of rules and packet read in two different files.
* Firewall called recursively on list of rules
* Each rule of the form accept clause|deny clause
* Fun called from Firewall (takes a rule and the packet as arguments)
* Fun matches the rule with adapter , ether ,tcp etc. and accordingly calls the matching predicate(passing rule and packet constraint as arguments)
* In the called predicate , terms ‘accept’ and ‘deny’ checked.
* The constraints of the rule and packet element passed to comp predicate which applies the constraints on the given packet’s specification .
* Conditions matched+’accept’ – this rule matches –go to next rule

Conditions matched+’deny’ –this rule denied –packet cannot be accepted—denied—program terminated.

* End of Recursion.