Joel van Egmond
10102094
CPSC 526 T03

1) Packet Filter

Assumptions:

- assumes that rule file is well-formed and of form:
  [allow/deny] [udp/tcp] [source ip]:[source port] -> [dest ip]:[dest port]
- assumes that the packet file is well-formed, and contains IP header, TCP/UDP header, and payload (no link layer header)

Test Cases:

Test cases follow the format of an input to **pgen.py** to create the test packet file, and then a call to **pfilter.py** to test it. Packets are tested against **testRules<n>.txt** rule files, which are included in the submission, but their specific contents are listed before each test.

Calls to **pgen.py** are of the form:
python pgen.py [tcp/udp/other] [source ip] [source port] [dest ip] [dest port] [out file name]
Note: pgen.py requires scapy

Calls to **pfilter.py** are of the form:
python pfilter.py [rules] [packet]
Note: pfilter.py does **not** require scapy

If you don't want to generate packets using pgen.py, all packets are found in the **/testPkts/** folder, and all rules are found in the **/testRules/** folder.

Case 1:
Rules:
allow tcp 1.*:* -> 1.*:*
deny udp 1.*:* -> 1.*:*

Purpose: Tests if packet with a protocol other than TCP/UDP works.

pgen input:
python pgen.py other 1.0.0.0 1 1.0.0.1 1 test1.dat
pfilter input:
python pfilter.py testRules1.txt test1.dat
Tests against:
no matching rule
Expected output:
unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 2:

Rules:
allow tcp 2.*:* -> 2.*:*
deny udp 2.*:* -> 2.*:*

Case 2a:

Purpose: Tests if a tcp packet can match a matching wildcard rule (tests matching protocol)

pgen input:
    python pgen.py tcp 2.0.0.0 1 2.0.0.1 1 test2a.dat
pfilter input:
    python pfilter.py testRules2.txt test2a.dat
Tests against:
    allow tcp 2.*:* -> 2.*:*
Expected output:
    allow

Case 2b:

Purpose: Tests if a udp packet can match a matching wildcard rule (tests matching protocol)

pgen input:
    python pgen.py udp 2.0.0.0 2 2.0.0.1 2 test2b.dat
pfilter input:
    python pfilter.py testRules2.txt test2b.dat
Tests against:
    deny udp 2.*:* -> 2.*:*
Expected output:
    deny

Joel van Egmond
10102094
CPSC 526 T03

Case 3:

Rules:
allow tcp 3.*:* -> 3.*:*
deny udp 3.*:* -> 3.*:*

Case 3a:

Purpose: Tests if a tcp packet can match a non-matching rule (tests matching protocol)

pgen input:
    python pgen.py tcp 3.0.0.0 1 3.0.0.1 1 test3a.dat
pfilter input:
    python pfilter.py testRules3.txt test3a.dat
Tests against:
    no matching rules
    (should fail to match: allow tcp 3.*:* -> 3.*:*)
Expected output:
    unspecified

Case 3b:

Purpose: Tests if a udp packet can match a non-matching rule (tests matching protocol)

pgen input:
    python pgen.py udp 3.0.0.0 2 3.0.0.1 2 test3b.dat
pfilter input:
    python pfilter.py testRules3.txt test3b.dat
Tests against:
    no matching rules
    (should fail to match: deny udp 3.*:* -> 3.*:*)
Expected output:
    unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 4:

Rules:
allow tcp 4.0.0.0:* -> 4.*:*

Case 4a:

Purpose: Tests if a packet can match a matching specific source ip rule (tests matching source ip)

pgen input:
    python pgen.py tcp 4.0.0.0 1 4.0.0.1 1 test4a.dat
pfilter input:
    python pfilter.py testRules4.txt test4a.dat
Tests against:
    allow tcp 4.0.0.0:* -> 4.*:*
Expected output:
    allow

Case 4b:

Purpose: Tests if a packet can match a non-matching specific source ip rule (tests matching source ip)

pgen input:
    python pgen.py tcp 4.1.0.0 2 4.0.0.1 2 test4b.dat
pfilter input:
    python pfilter.py testRules4.txt test4b.dat
Tests against:
    no matching rule
    (should fail to match: allow tcp 4.0.0.0:* -> 4.*:*)
Expected output:
    unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 5:

Rules:
deny tcp 5.*:1 -> 5.*:*

Case 5a:

Purpose: Tests if a packet can match a matching specific source port rule (tests matching source port)

pgen input:
    python pgen.py tcp 5.0.0.0 1 5.0.0.1 1 test5a.dat
pfilter input:
    python pfilter.py testRules5.txt test5a.dat
Tests against:
    deny tcp 5.*:1 -> 5.*:*
Expected output:
    deny

Case 5b:

Purpose: Tests if a packet can match a non-matching specific source port rule (tests matching source port)

pgen input:
    python pgen.py tcp 5.0.0.0 2 5.0.0.1 2 test5b.dat
pfilter input:
    python pfilter.py testRules5.txt test5b.dat
Tests against:
    no matching rule
    (should fail to match: deny tcp 5.*:1 -> 5.*:*)
Expected output:
    unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 6:

Rules:
allow tcp 6.*:* -> 6.0.0.1:*

Case 6a:

Purpose: Tests if a packet can match a matching specific destination ip rule (tests matching destination ip)

pgen input:
    python pgen.py tcp 6.0.0.0 1 6.0.0.1 1 test6a.dat
pfilter input:
    python pfilter.py testRules6.txt test6a.dat
Tests against:
    allow tcp 6.*:* -> 6.0.0.1:*
Expected output:
    allow

Case 6b:

Purpose: Tests if a packet can match a non-matching specific destination ip rule (tests matching destination ip)

pgen input:
    python pgen.py tcp 6.0.0.0 2 6.1.0.1 2 test6b.dat
pfilter input:
    python pfilter.py testRules6.txt test6b.dat
Tests against:
    no matching rule
    (should fail to match: allow tcp 6.*:* -> 6.0.0.1:*)
Expected output:
    unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 7:

Rules:
deny tcp 7.*:* -> 7.*:1

Case 7a:

Purpose: Tests if a packet can match a matching specific destination port rule (tests matching destination port)

pgen input:
    python pgen.py tcp 7.0.0.0 1 7.0.0.1 1 test7a.dat
pfilter input:
    python pfilter.py testRules7.txt test7a.dat
Tests against:
    deny tcp 7.*:* -> 7.*:1
Expected output:
    deny

Case 7b:

Purpose: Tests if a packet can match a non-matching specific destination port rule (tests matching destination port)

pgen input:
    python pgen.py tcp 7.0.0.0 2 7.0.0.1 2 test7b.dat
pfilter input:
    python pfilter.py testRules7.txt test7b.dat
Tests against:
    no matching rule
    (should fail to match: deny tcp 7.*:* -> 7.*:1)
Expected output:
    unspecified

Joel van Egmond
10102094
CPSC 526 T03

Case 8:

Rules:
allow tcp 8.*:* -> 8.*:*
deny udp 8.*:* -> 8.*:*

Case 8a:
Purpose: Tests if the program can differentiate between tcp and udp protocols

pgen input:
    python pgen.py tcp 8.0.0.0 1 8.0.0.1 1 test8a.dat
pfilter input:
    python pfilter.py testRules8.txt test8a.dat
Tests against:
    allow tcp 8.*:* -> 8.*:*
Expected output:
    allow

Case 8b:
Purpose: Tests if the program can differentiate between tcp and udp protocols

pgen input:
    python pgen.py udp 8.0.0.0 1 8.0.0.1 1 test8b.dat
pfilter input:
    python pfilter.py testRules8.txt test8b.dat
Tests against:
    deny udp 8.*:* -> 8.*:*
Expected output:
    deny

Joel van Egmond
10102094
CPSC 526 T03

Case 9:

Rules:
allow tcp 9.0.0.0:* -> 9.*:*
deny tcp 9.*:* -> 9.0.0.0:*

Case 9a:

Purpose: Tests if the program can differentiate between source ip addresses and destination ip addresses

pgen input:
    python pgen.py tcp 9.0.0.0 1 9.0.0.1 1 test9a.dat
pfilter input:
    python pfilter.py testRules9.txt test9a.dat
Tests against:
    allow tcp 9.0.0.0:* -> 9.*:*
Expected output:
    allow

Case 9b:

Purpose: Tests if the program can differentiate between source ip addresses and destination ip addresses

pgen input:
    python pgen.py tcp 9.0.0.1 1 9.0.0.0 1 test9b.dat
pfilter input:
    python pfilter.py testRules9.txt test9b.dat
Tests against:
    deny tcp 9.*:* -> 9.0.0.0:*
Expected output:
    deny

Joel van Egmond
10102094
CPSC 526 T03

Case 10:

Rules:
allow tcp 10.*:1 -> 10.*:*
deny tcp 10.*:* -> 10.*:1

Case 10a:

Purpose: Tests if the program can differentiate between source ports and destination ports

pgen input:
    python pgen.py tcp 10.0.0.0 1 10.0.0.1 0 test10a.dat
pfilter input:
    python pfilter.py testRules10.txt test10a.dat
Tests against:
    allow tcp 10.*:1 -> 10.*:*
Expected output:
    allow

Case 10b:

Purpose: Tests if the program can differentiate between source ports and destination ports

pgen input:
    python pgen.py tcp 10.0.0.0 0 10.0.0.1 1 test10b.dat
pfilter input:
    python pfilter.py testRules10.txt test10b.dat
Tests against:
    deny tcp 10.*:* -> 10.0.0.1:1
Expected output:
    deny

Joel van Egmond
10102094
CPSC 526 T03

Case 11:

Rules:
allow tcp 11.*:1 -> 11.*:*
deny tcp 11.0.*:2 -> 11.*:*
allow tcp 11.0.0.*:3 -> 11.*:*

Case 11a:

Purpose: Tests if the program can match on all levels of wildcards in the ip address

pgen input:
    python pgen.py tcp 11.0.0.0 1 11.0.0.1 0 test11a.dat
pfilter input:
    python pfilter.py testRules11.txt test11a.dat
Tests against:
    allow tcp 11.*:1 -> 11.*:*
Expected output:
    allow

Case 11b:

pgen input:
    python pgen.py tcp 11.0.0.0 2 11.0.0.1 0 test11b.dat
pfilter input:
    python pfilter.py testRules11.txt test11b.dat
Tests against:
    deny tcp 11.0.*:2 -> 11.*:*
Expected output:
    deny

Case 11c:

pgen input:
    python pgen.py tcp 11.0.0.0 3 11.0.0.1 0 test11c.dat
pfilter input:
    python pfilter.py testRules11.txt test11c.dat
Tests against:
    allow tcp 11.0.0.*:3 -> 11.*:*
Expected output:
    allow

Joel van Egmond
10102094
CPSC 526 T03

Case 12:

Rules:
allow tcp 12.0.0.0:* -> 12.*:*
deny tcp 12.*:* -> 12.*:*

Case 12a:

Purpose: Tests if the program will match rules in the correct order (first rule to match is response)

pgen input:
    python pgen.py tcp 12.0.0.0 1 12.0.0.1 0 test12a.dat
pfilter input:
    python pfilter.py testRules12.txt test12a.dat
Tests against:
    allow tcp 12.0.0.0:* -> 12.*:* (should match)
    deny tcp 12.*:* -> 12.*:* (would also match, but shouldn't because the other rule is first)
Expected output:
    allow

Case 12b:

Purpose: Tests if the program will match rules in the correct order (first rule to match is response)

pgen input:
    python pgen.py tcp 12.1.1.1 1 12.0.0.1 0 test12b.dat
pfilter input:
    python pfilter.py testRules12.txt test12b.dat
Tests against:
    deny tcp 12.*:* -> 12.*:*
Expected output:
    deny