CSE508: Network Security, Spring 2016

Homework 1: Passive Network Monitoring

-------------------------------------------------------------------------------

Submission deadline: 2/26/2016 11:59pm EDT

Submission through https://blackboard.stonybrook.edu

In this assignment you will get familiar with passive network traffic analysis.

Tcpdump is probably the most popular and widely used passive network monitoring

tool. Tcpdump is built on top of the Libpcap packet capture library and can

capture and display packet headers and payloads either from a network interface

or a network trace file. Your task is to analyze a network trace file and

answer various questions.

0. Preparation

Download the pcap trace file from:

https://piazza.com/class\_profile/get\_resource/ijor5zasnfz4iv/ik7sunqx4c35cg

Use the following command to read the trace file:

tcpdump -n -r hw1.pcap | head

-n skips the DNS resolution of IP addresses (faster)

The output is redirected to 'head' to avoid flooding the console - this is a

good strategy while assembling your commands. Alternatively, you can use

more(1) to scroll through the output, or you can redirect all output to a text

file for further analysis.

1. Network traffic analysis using tcpdump

You can find the answers to all of the following questions using only tcpdump's

man page and a few shell commands. You may find the following utilities useful:

wc, grep, sed, awk, cut, head, tail, more, sort, uniq. You are free to use any

other tools you might want (although the above are more than enough).

1.1 How many packets does the trace contain?

32664

tcpdump -n -r hw1.pcap | wc –l

1.2 How many ARP packets and how many UDP packets?

11304

tcpdump -n -r hw1.pcap arp | wc –l

18382

tcpdump -n -r hw1.pcap udp | wc -l

1.3 How many packets were exchanged between 192.168.0.200 and 91.189.92.190?

543

tcpdump -n -r hw1.pcap host 192.168.0.200 and 91.189.92.190| wc

1.4 Print the unique source IP addresses found in the trace.

33

tcpdump -n -r hw1.pcap ip | cut -f 3 -d ' ' | cut -f 1,2,3,4 -d '.' |sort -n | uniq -c | sort -r -n| wc -l

11890 192.168.0.1

3424 192.168.0.11

2684 192.168.0.3

838 192.168.0.200

697 91.189.91.14

387 192.168.0.7

369 91.189.92.190

263 192.168.0.12

219 0.0.0.0

164 192.168.0.10

104 192.168.0.5

61 91.189.91.15

38 91.189.88.33

38 194.168.4.100

36 87.98.246.8

32 1.234.31.20

16 92.240.68.152

12 91.189.90.40

12 62.252.170.91

11 159.148.96.184

10 192.168.0.6

7 46.51.197.89

7 216.137.63.137

6 91.189.90.41

6 216.137.63.121

5 192.168.0.2

5 122.154.101.54

4 91.189.89.88

4 87.230.23.162

4 46.51.197.88

3 169.254.56.55

2 62.252.170.81

2 130.75.16.5

1.5 Print the unique destination IP addresses found in the trace.

31

tcpdump -n -r hw1.pcap ip | cut -f 5 -d ' ' | uniq -u | wc –l

13275 239.255.255.250

2323 192.168.0.255

1516 224.0.0.252

1397 192.168.0.200

914 224.0.0.22

841 224.0.0.251

295 91.189.91.14

259 255.255.255.255

174 91.189.92.190

48 224.0.0.253

46 91.189.88.33

42 91.189.91.15

38 194.168.4.100

28 192.168.0.12

24 87.98.246.8

20 1.234.31.20

16 92.240.68.152

16 91.189.90.40

15 159.148.96.184

14 62.252.170.91

9 46.51.197.89

9 216.137.63.137

8 91.189.90.41

8 46.51.197.88

6 91.189.89.88

6 216.137.63.121

4 62.252.170.81

3 169.254.255.255

3 122.154.101.54

2 87.230.23.162

1 130.75.16.5

1.6 What are the top-5 TCP and top-5 UDP destination ports?

TCP ports:

727 80:

697 54634:

346 49836:

61 47110:

36 40341:

UDP ports:

13178 1900:

1925 137:

1516 5355:

840 5353:

399 138:

tcpdump -n -r hw1.pcap udp | cut -f 5 -d ' ' | cut -f 5 -d '.' | sort -n | uniq -c | sort -rn | head -5

1.7 How many TCP packets have the SYN flag set?

75

tcpdump -n -r hw1.pcap 'tcp[13]&2 == 2' | wc -l

1.8 How many TCP connection attempts were made?

37

tcpdump -n -r hw1.pcap 'tcp[13] == 2' | wc -l

1.9 Towards which ports were TCP connection attempts made?

33 80:

2 9100:

1 465:

1 443:

tcpdump -n -r hw1.pcap 'tcp[13] == 2' | cut -f 5 -d ' ' | cut -f 5 -d '.' | sort -rn | uniq -c | sort -rn

1.10 How many HTTP GET requests were made? Print the URLs of all HTTP requests for JPG files.

94

tcpdump -n -r hw1.pcap 'tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x47455420'| wc –l

tcpdump -n -A -r hw1.pcap 'tcp dst port 80' | grep 'HTTP: GET' | wc -l

5

http://pic.leech.it/i/f166c/479246b0asttas.jpg

/i/f166c/479246b0asttas.jpg

http://ecx.images-amazon.com/images/I/41oZ1XsiOAL.\_SL500\_AA300\_.jpg

http://www.nature.com/news/2009/090527/images/459492a-i1.0.jpg

/news/2009/090527/images/459492a-i1.0.jpg

tcpdump -n -r hw1.pcap 'tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x47455420' | grep '.jpg'| awk ‘{print $24}’

1.11 When (date and time) was the first and last packet of the trace sent?

Last: 2013-01-14 13:27:03.691498

First: 2013-01-12 11:37:42.871346

tcpdump -n -tttt -r hw1.pcap | cut -f 1,2 -d ' ' | sort -n| head

tcpdump -n -tttt -r hw1.pcap | cut -f 1,2 -d ' ' | sort –n -r| head

1.12 What is the brand of the device that sent most of the packets? What is its IP address?

19743 c4:3d:c7:17:6f:9b Netgear IP : 192.168.0.1

4982 00:16:44:b5:86:2e

4587 44:6d:57:f6:7e:00 Liteon Technology Corporation

940 3c:d0:f8:4e:4b:a1

tcpdump -n –e –i “eth0”-r hw1.pcap | cut -f 2 -d ' ' | sort | uniq -c| sort -n -r

1.13 Report the distribution of Ethernet packet sizes (how many packets of size X exist in the trace, for all values of X in the trace).

tcpdump -n -e -r hw1.pcap | awk '{print $9}'| sort -n | uniq -c | sort -nr

12190 60

2773 331

2759 395

2758 405

2749 340

1740 92

1046 66

1034 1514

887 64

341 91

326 342

279 243

232 63

180 110

165 484

164 475

162 539

161 555

158 527

155 541

146 175

121 120

88 386

88 338

87 74

87 392

87 390

86 394

86 362

86 318

85 398

85 326

62 167

60 165

56 249

54 81

52 82

43 84

40 1033

39 219

36 231

33 460

32 602

28 119

24 42

24 315

22 698

22 246

17 551

17 202

16 245

15 302

15 127

14 87

13 90

13 62

13 489

12 666

12 307

12 168

11 142

10 556

10 542

10 540

10 528

10 485

10 476

10 332

10 330

10 154

10 107

9 284

9 248

8 816

8 75

8 68

7 328

7 149

7 125

6 76

6 69

6 610

6 350

6 333

6 282

6 264

6 1034

5 88

5 817

5 628

5 613

5 329

4 89

4 85

4 79

4 65

4 346

4 345

4 322

4 267

4 254

4 244

4 177

4 170

4 124

4 106

4 1035

3 70

3 497

3 344

3 298

3 269

3 247

3 233

2 95

2 94

2 93

2 77

2 704

2 636

2 630

2 624

2 621

2 615

2 614

2 612

2 611

2 607

2 598

2 593

2 590

2 550

2 546

2 502

2 478

2 418

2 383

2 321

2 310

2 309

2 305

2 294

2 288

2 268

2 266

2 265

2 261

2 257

2 255

2 252

2 236

2 229

2 220

2 208

2 184

2 129

2 1212

2 111

2 105

1 979

1 952

1 926

1 760

1 752

1 746

1 730

1 72

1 694

1 690

1 679

1 678

1 640

1 608

1 601

1 596

1 592

1 588

1 568

1 552

1 548

1 544

1 54

1 535

1 518

1 506

1 482

1 479

1 446

1 433

1 432

1 429

1 428

1 417

1 412

1 396

1 389

1 374

1 372

1 347

1 335

1 320

1 317

1 312

1 308

1 306

1 223

1 207

1 195

1 178

1 171

1 156

1 1469

1 144

1 1218

1 1179

1 1170

1 1162

1 1102

2. What to submit

An ASCII text file with the answers to the above questions, along with the

command/script/approach used to find them.