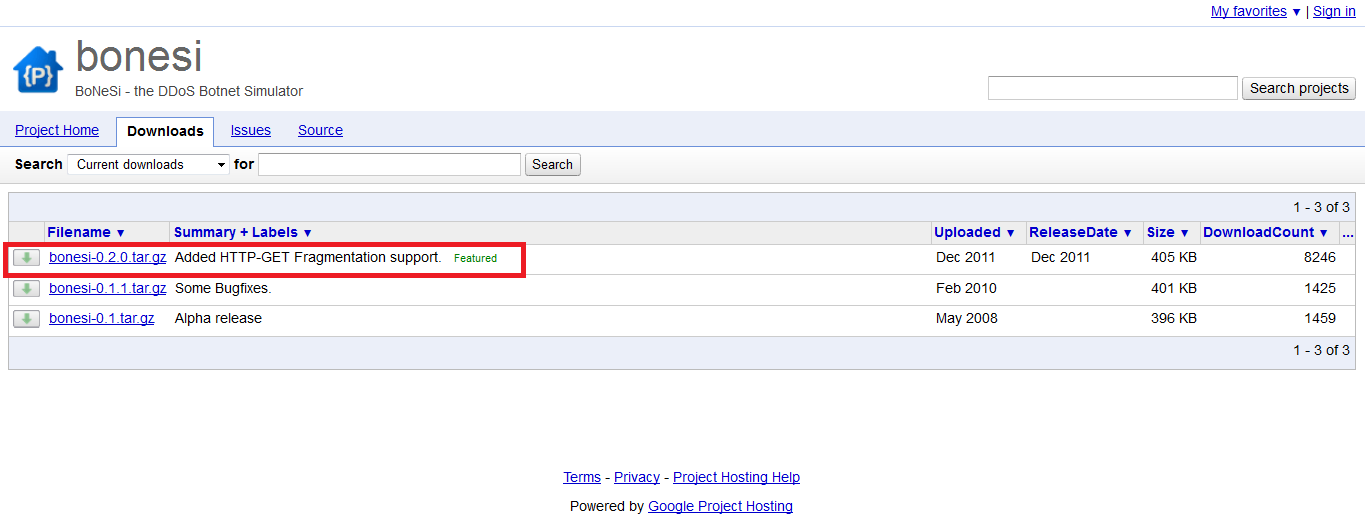
# Appendix A – BoNeSi Guide

The guide below walkthrough the process of downloading and installing the DDoS simulation tool BoNeSi in the EstiNet SDN environment [21] [37].

**1. Visit BoNeSi website**

Visit the following link: <https://code.google.com/p/bonesi/>

**2. Download "**[bonesi-0.2.0.tar.gz](https://code.google.com/p/bonesi/downloads/detail?name=bonesi-0.2.0.tar.gz&can=2&q=)**"**



**3. Unzip BoNeSi**

tar -jxvf bonesi-0.2.0.tar.gz

**4. Install prerequisite**

yum -y install libpcap-devel

yum -y install libnet-devel

**5. Move to bonesi-0.2.0 folder**

cd ../ bonesi-0.2.0

**6. Compile BoNeSi**

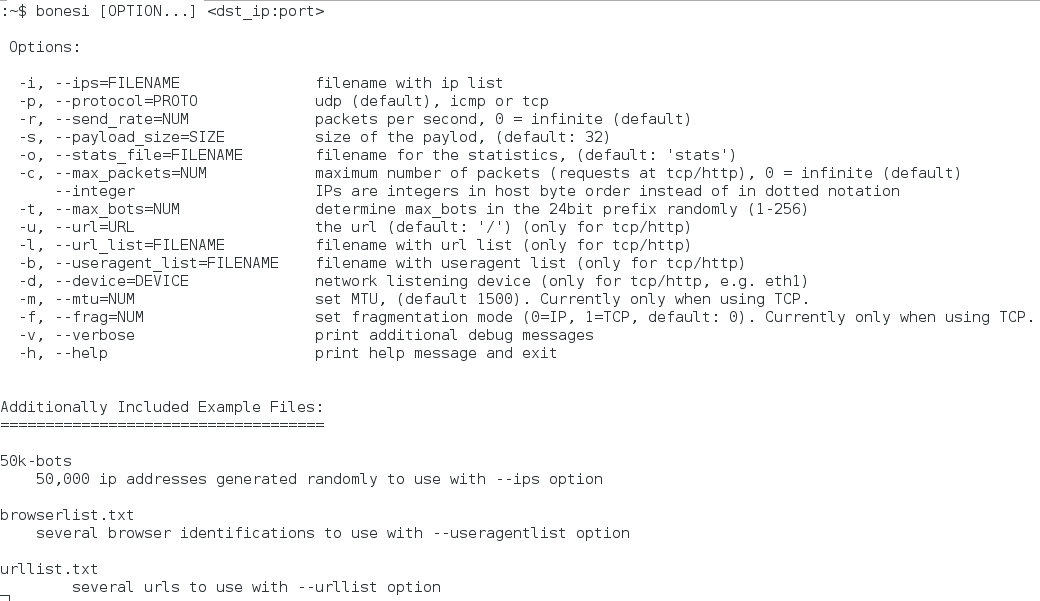
./configure

make

make install

**7. See the usage of BoNeSi**

vim README



**8. Find the binary of BoNeSi**

whereis bonesi



**9. Add BoNeSi binary to EstiNet**

cp /usr/local/bin/bonesi /usr/local/estinet/tools

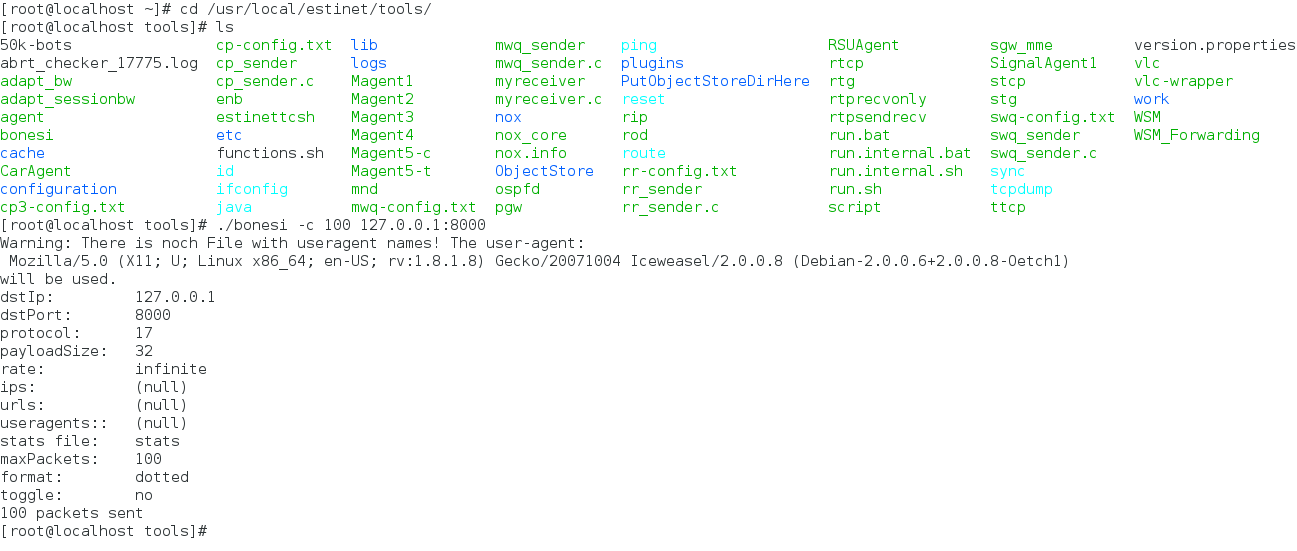
**10. Move to /usr/local/estinet/tools**

cd /usr/local/estinet/tools

**11. Try to run BoNeSi on terminal**

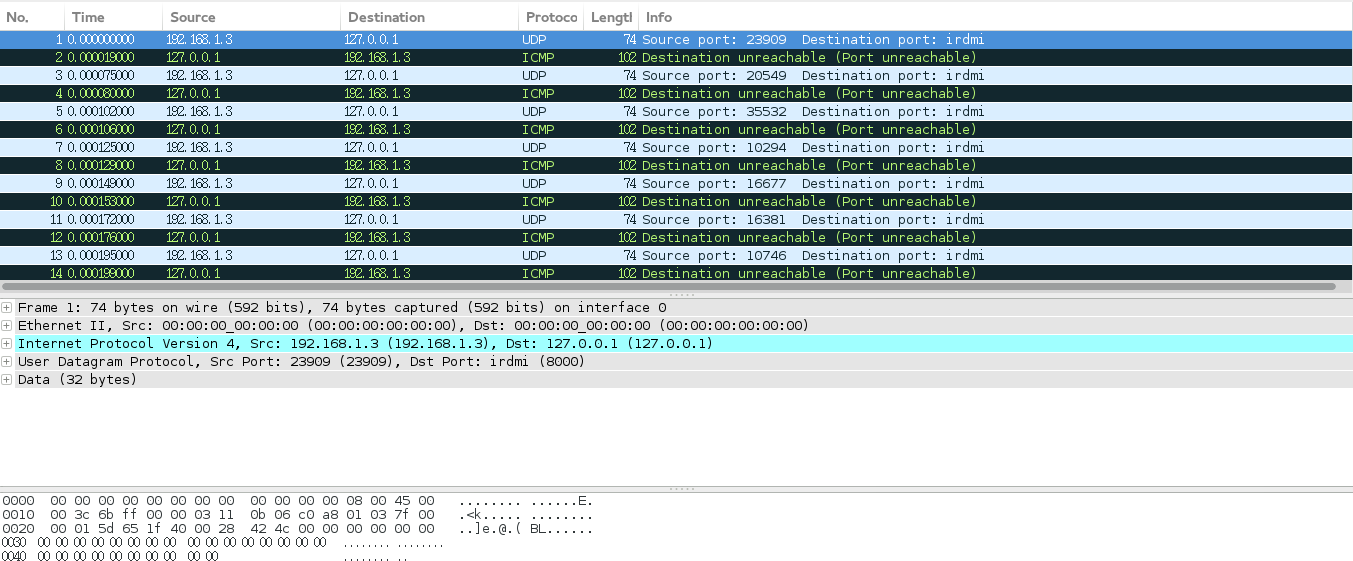
ls

./bonesi -c 100 127.0.0.1:8000

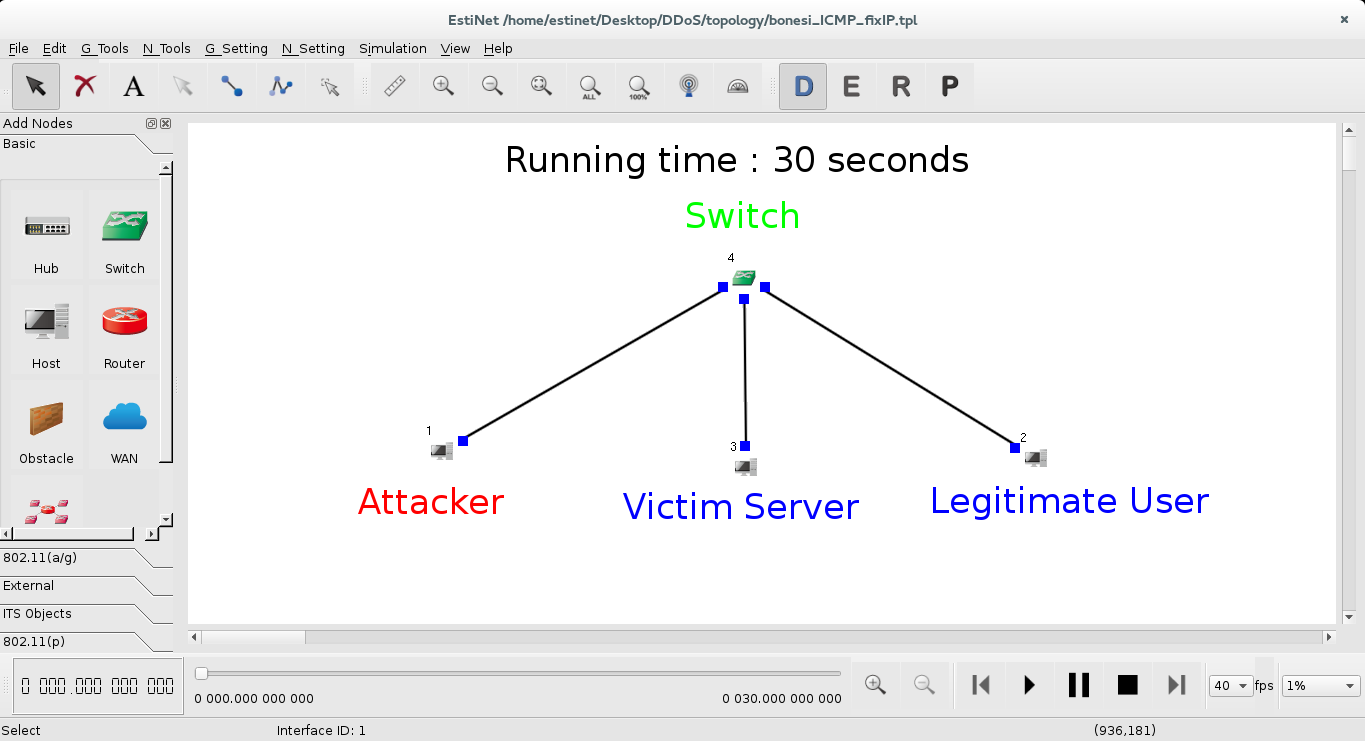


Send 100 UDP packets to lo device with port 8000

**11. Use Wireshark to see lo device**



**12. Draw the topology in EstiNet**



**13. Run BoNeSi in "Attacker" (host 1)**

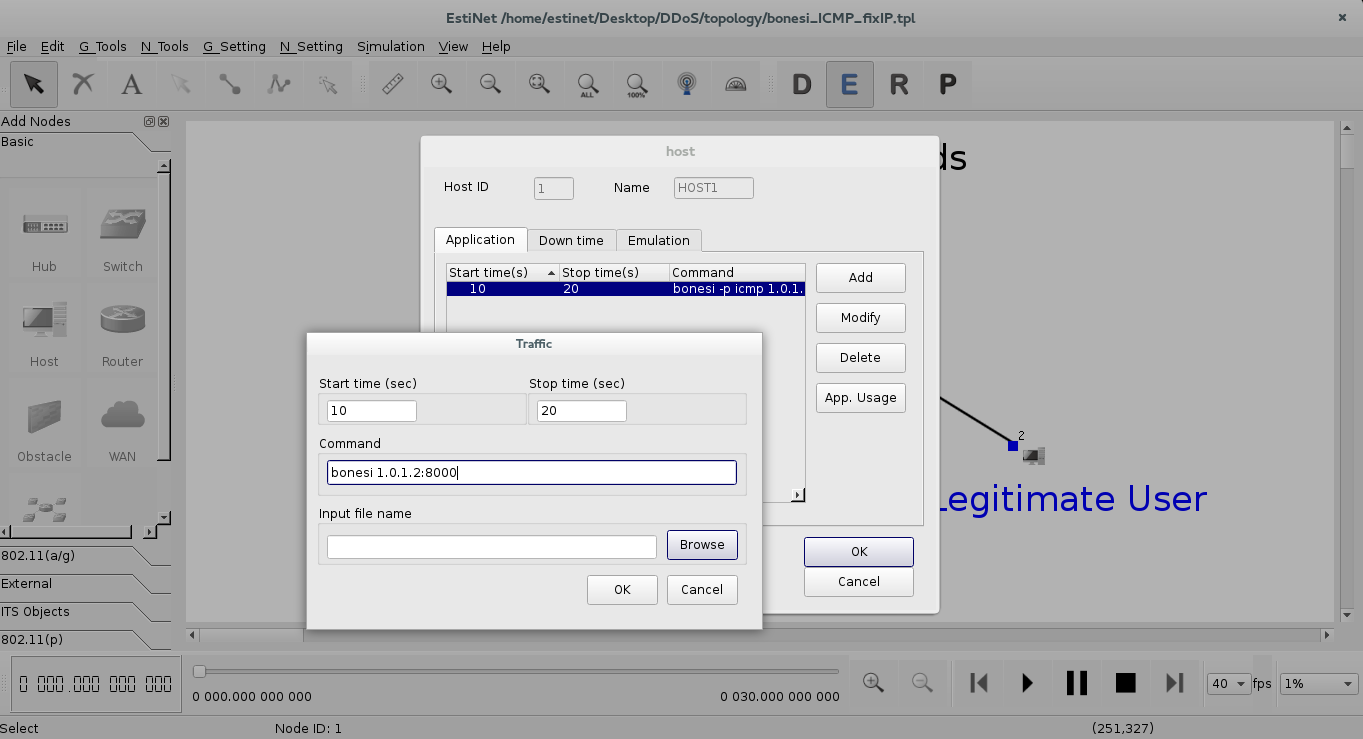
Double-click host1

Left-click "Add" button

Enter bonesi 1.0.1.2:8000 (where 1.0.1.2 is the victim server's IP in the simulator)

set Bonesi's running time at 10~20 seconds

Left-click "OK" button to save and leave



**13. Run stcp in "Legitimate User" (host 2) to simulate a legal TCP sender.**

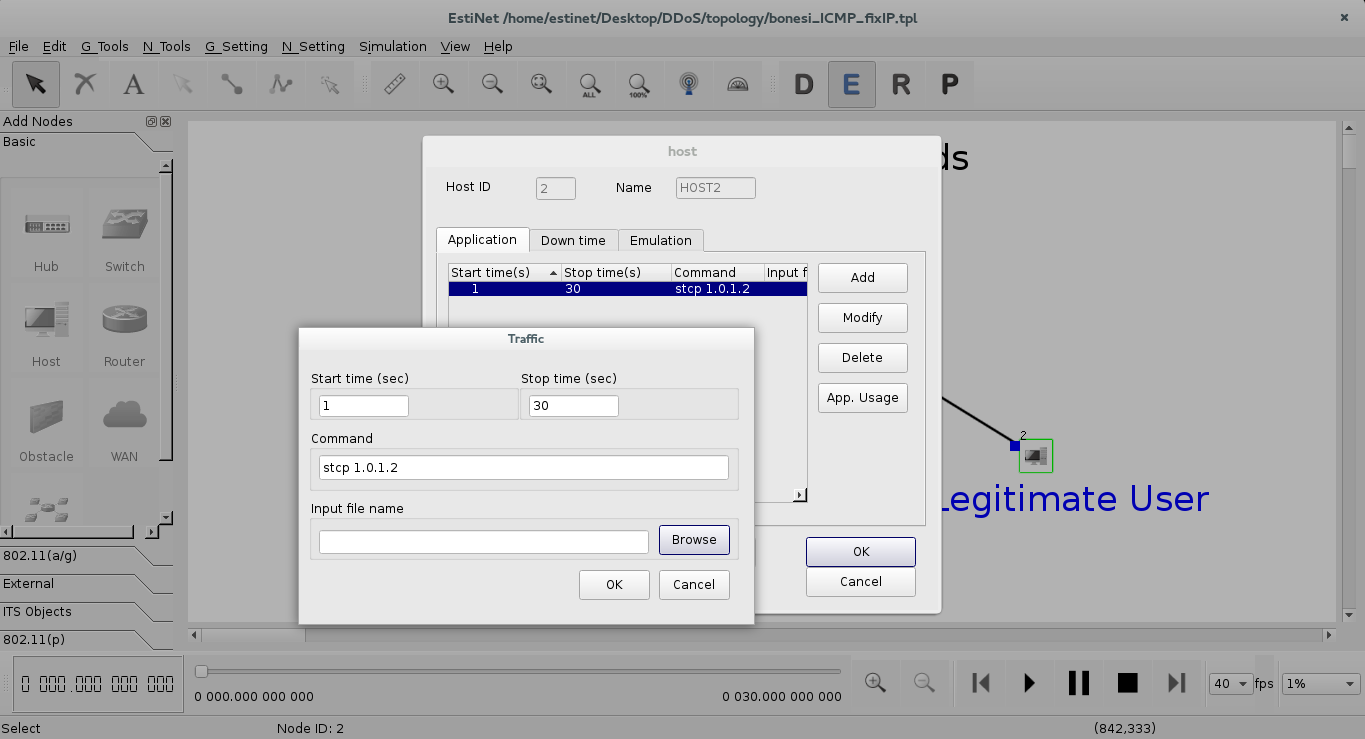
Double-click host2

Left-click "Add" button

Enter stcp 1.0.1.2 (where 1.0.1.2 is the victim server's IP in the simulator)

set stcp's running time at 1~30 seconds

Left-click "OK" button to save and leave



**14. Run rtcp in "Victim Server" (host 3) to simulate a legal TCP receiver.**

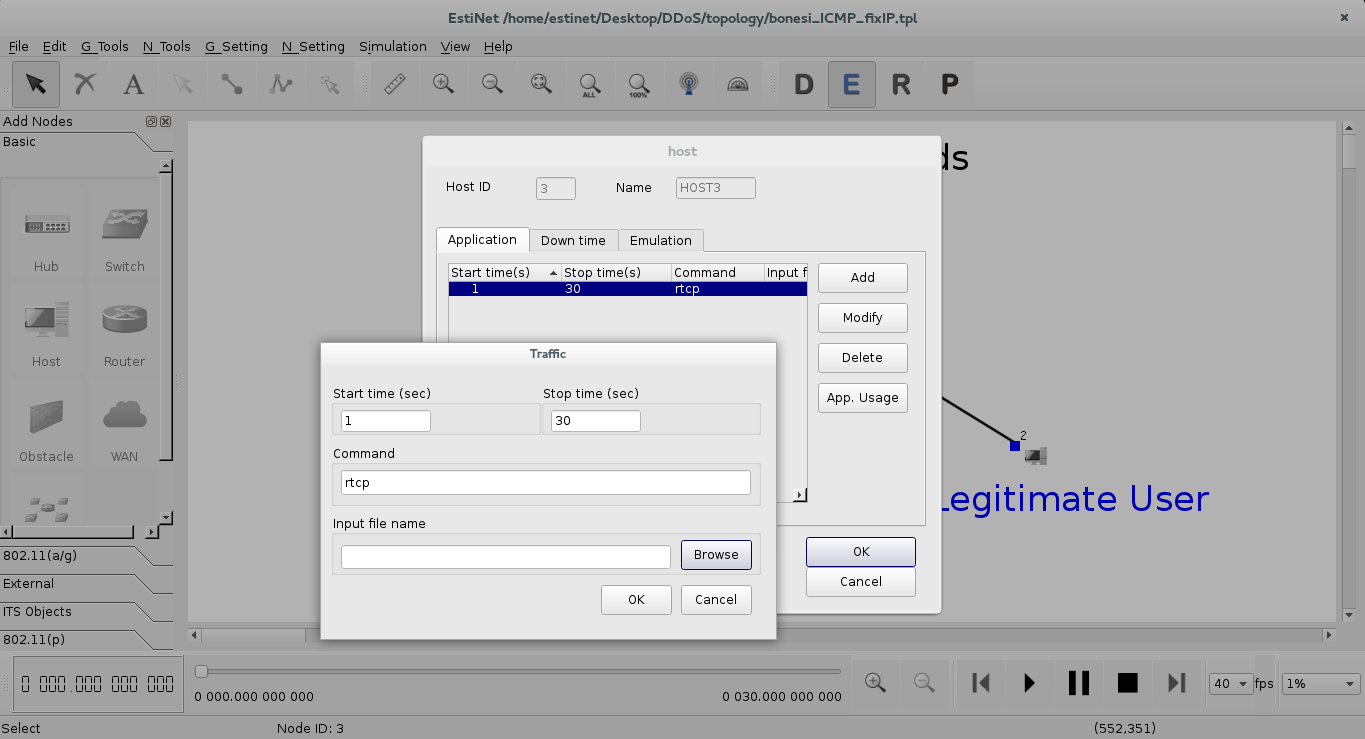
Double-click host3

Left-click "Add" button

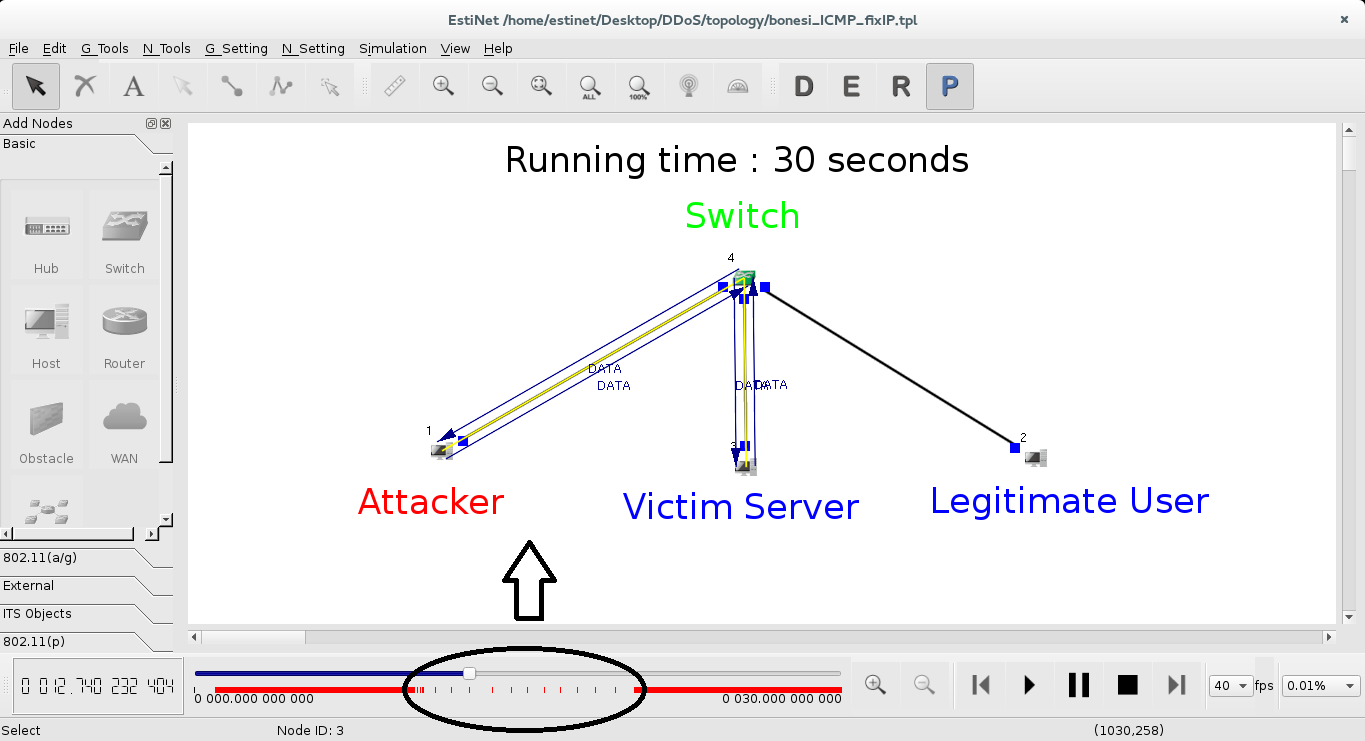
Enter rtcp

set rtcp's running time at 1~30 seconds

Left-click "OK" button to save and leave



**15. Run the simulation and see the results.**



Due to BoNeSi's UDP DDoS, TCP connections between Victim Server and Legitimate User were interrupted during 10~20 seconds in the total simulation running.

# Appendix B – Framework Confidence Analysis Application

1. import java.awt.Component;
2. import java.awt.EventQueue;
3. import javax.swing.JComboBox;
4. import javax.swing.JFileChooser;
5. import javax.swing.JFrame;
6. import javax.swing.JButton;
7. import java.awt.event.ActionListener;
8. import java.awt.event.ActionEvent;
9. import java.io.BufferedReader;
10. import java.io.File;
11. import java.io.FileInputStream;
12. import java.io.IOException;
13. import java.io.InputStream;
14. import java.io.InputStreamReader;
15. import java.math.RoundingMode;
16. import java.net.HttpURLConnection;
17. import java.net.URL;
18. import java.text.DecimalFormat;
19. import java.text.NumberFormat;
20. import java.util.ArrayList;
21. import java.util.List;
22. import javax.swing.JLabel;
23. import org.apache.poi.ss.usermodel.FormulaEvaluator;
24. import org.apache.poi.ss.usermodel.Row;
25. import org.apache.poi.xssf.usermodel.XSSFSheet;
26. import org.apache.poi.xssf.usermodel.XSSFWorkbook;
27. import org.omg.CORBA.NameValuePair;
28. import org.apache.commons.codec.binary.Base64;
29. import jxl.Cell;
30. import jxl.CellType;
31. import jxl.Sheet;
32. import jxl.Workbook;
33. import java.awt.BorderLayout;
34. import javax.swing.JTextField;
35. public class test {
36. private JFrame frame;
37. final JFileChooser fc = new JFileChooser();
38. Component c;
39. float bs, es;
40. private JTextField textField;
41. private JTextField textField\_1;
42. private JTextField textField\_2;
43. JLabel lblNewLabel\_4, lblNewLabel\_6, lblNewLabel\_8, lblNewLabel\_10, lblNewLabel\_12, lblNewLabel\_14;
44. float Spoofability, Measurability, Variablitiy, base\_score;
45. //In response to a button click: //int returnVal = fc.showOpenDialog(aComponent);
46. /\*\*\* Launch the application.\*/
47. public static void main(String[] args) {
48. EventQueue.invokeLater(new Runnable() {
49. public void run() {
50. try {
51. test window = new test();
52. window.frame.setVisible(true);
53. } catch (Exception e) {
54. e.printStackTrace();
55. }
56. }
57. });
58. }
60. /\*\*\* Create the application.\*/
61. public test() {
62. initialize();
63. }
64. /\*\*\* Initialize the contents of the frame.\*/
65. private void initialize() {
66. frame = new JFrame();
67. frame.setBounds(100, 100, 700, 650);
68. frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
69. frame.getContentPane().setLayout(null);
71. JLabel lblNewLabel = new JLabel("Pre");
72. lblNewLabel.setBounds(10, 76, 139, 14);
73. frame.getContentPane().add(lblNewLabel);
75. JLabel lblNewLabel\_1 = new JLabel("Actual");
76. lblNewLabel\_1.setBounds(10, 116, 139, 14);
77. frame.getContentPane().add(lblNewLabel\_1);
79. JLabel lblNewLabel\_2 = new JLabel("Post");
80. lblNewLabel\_2.setBounds(10, 156, 139, 14);
81. frame.getContentPane().add(lblNewLabel\_2);
83. textField = new JTextField();
84. textField.setBounds(418, 74, 270, 20);
85. frame.getContentPane().add(textField);
86. textField.setColumns(10);
88. textField\_1 = new JTextField();
89. textField\_1.setBounds(418, 114, 270, 20);
90. frame.getContentPane().add(textField\_1);
91. textField\_1.setColumns(10);
93. textField\_2 = new JTextField();
94. textField\_2.setBounds(418, 154, 270, 20);
95. frame.getContentPane().add(textField\_2);
96. textField\_2.setColumns(10);
98. JButton btnNewButton = new JButton("Calculate");
99. btnNewButton.setBounds(264, 201, 210, 23);
100. frame.getContentPane().add(btnNewButton);
101. JLabel lblNewLabel\_3 = new JLabel("Base\_Score\_Confidence\_Rating");
102. lblNewLabel\_3.setBounds(10, 258, 241, 14);
103. frame.getContentPane().add(lblNewLabel\_3);
104. lblNewLabel\_4 = new JLabel("New label");
105. lblNewLabel\_4.setBounds(418, 258, 270, 14);
106. frame.getContentPane().add(lblNewLabel\_4);
107. JLabel lblNewLabel\_5 = new JLabel("Base\_Score\_Confidence\_Rating 2");
108. lblNewLabel\_5.setBounds(10, 310, 241, 14);
109. frame.getContentPane().add(lblNewLabel\_5);
110. lblNewLabel\_6 = new JLabel("New label");
111. lblNewLabel\_6.setBounds(418, 310, 270, 14);
112. frame.getContentPane().add(lblNewLabel\_6);
113. JLabel lblNewLabel\_7 = new JLabel("Envir\_Score\_Confidence\_Rating");
114. lblNewLabel\_7.setBounds(10, 359, 241, 14);
115. frame.getContentPane().add(lblNewLabel\_7);
116. lblNewLabel\_8 = new JLabel("New label");
117. lblNewLabel\_8.setBounds(418, 359, 270, 14);
118. frame.getContentPane().add(lblNewLabel\_8);
119. JLabel lblNewLabel\_9 = new JLabel("Envir\_Score\_Confidence\_Rating 2");
120. lblNewLabel\_9.setBounds(10, 405, 241, 20);
121. frame.getContentPane().add(lblNewLabel\_9);
122. lblNewLabel\_10 = new JLabel("New label");
123. lblNewLabel\_10.setBounds(418, 408, 270, 14);
124. frame.getContentPane().add(lblNewLabel\_10);
125. JLabel lblNewLabel\_11 = new JLabel("Confidence\_Rating");
126. lblNewLabel\_11.setBounds(10, 459, 241, 14);
127. frame.getContentPane().add(lblNewLabel\_11);
128. lblNewLabel\_12 = new JLabel("New label");
129. lblNewLabel\_12.setBounds(416, 459, 272, 14);
130. frame.getContentPane().add(lblNewLabel\_12);
131. JLabel lblNewLabel\_13 = new JLabel("Confidence\_Rating 2");
132. lblNewLabel\_13.setBounds(10, 508, 193, 14);
133. frame.getContentPane().add(lblNewLabel\_13);
134. lblNewLabel\_14 = new JLabel("New label");
135. lblNewLabel\_14.setBounds(418, 508, 270, 14);
136. frame.getContentPane().add(lblNewLabel\_14);
137. JButton btnNewButton\_1 = new JButton("Connect");
138. btnNewButton\_1.addActionListener(new ActionListener() {
139. public void actionPerformed(ActionEvent arg0) {
140. testopendaylight();
141. }
142. });
143. btnNewButton\_1.setBounds(571, 12, 117, 25);
144. frame.getContentPane().add(btnNewButton\_1);
145. JButton btnNewButton\_2 = new JButton("Nodes in Network");
146. btnNewButton\_2.addActionListener(new ActionListener() {
147. public void actionPerformed(ActionEvent arg0) {
148. testopendayy.getNodes("admin", "admin", "http://127.0.0.1:8181");
149. }
150. });
151. btnNewButton\_2.setBounds(495, 47, 193, 25);
152. frame.getContentPane().add(btnNewButton\_2);
153. btnNewButton.addActionListener(new ActionListener() {
154. @Override
155. public void actionPerformed(ActionEvent arg0) {
156. // TODO Auto-generated method stub
157. System.out.println(textField.getText().toString());
158. System.out.println(textField\_1.getText().toString());
159. System.out.println(textField\_2.getText().toString());
160. testdo(Float.parseFloat(textField.getText().toString()), Float.parseFloat(textField\_1.getTex t().toString()), Float.parseFloat(textField\_2.getText().toString()), 1);
161. }
162. });
163. }
164. p rotected void testopendaylight() {
165. // TODO Auto-generated method stub
166. String user = "admin";
167. String password = "admin";
168. String baseURL = "http://127.0.0.1:8181/controller/nb/v2/flowprogrammer"; // String baseURL= "http://127.0.0.1:8181/index.html#/topology";
169. String containerName = "default";
171. try {
172. // Create URL = base URL + container
173. URL url = new URL(baseURL + "/" + containerName);
174. // Create authentication string and encode it to Base64
175. String authStr = user + ":" + password;
176. String encodedAuthStr = Base64.encodeBase64String(authStr.getBytes());
177. // Create Http connection
178. HttpURLConnection connection = (HttpURLConnection) url.openConnection();
179. // Set connection properties
180. connection.setRequestMethod("GET");
181. connection.setRequestProperty("Authorization", "Basic " + encodedAuthStr);
182. connection.setRequestProperty("Accept", "application/json");
183. // Get the response from connection's inputStream
184. InputStream content = (InputStream) connection.getInputStream();
185. BufferedReader in = new BufferedReader(new InputStreamReader(content));
186. String line = "";
187. while ((line = in .readLine()) != null) {
188. System.out.println(line);
189. System.out.println("---------------------------------------------------");
190. System.out.println("connection Succcesfull");
191. System.out.println("----------------------------------------------------");
192. }
193. } catch (Exception e) {
194. e.printStackTrace();
195. }
196. }
197. public void read(String inputfile) throws IOException {
198. File inputworkbook = new File(inputfile);
199. Workbook w;
200. try {
201. w = Workbook.getWorkbook(inputworkbook);
202. Sheet s = w.getSheet(0);
203. for (int j = 0; j < s.getColumns(); j++) {
204. for (int i = 0; i < s.getRows(); i++) {
205. Cell c = s.getCell(j, i);
206. CellType ct = c.getType();
207. //if(ct==CellType.LABEL)
208. System.out.println("Labrl" + c.getColumn());
209. // if(ct==CellType.NUMBER)
210. System.out.println("Number " + c.getColumn());
211. }
212. }
213. }
214. catch(Exception e) {
215. e.printStackTrace();
216. }
217. }
218. void testf(File file) throws IOException {
219. float a1 = 0, a2 = 0, a3;
220. FileInputStream fis = new FileInputStream(file);
222. XSSFWorkbook wb = new XSSFWorkbook(fis);
224. XSSFSheet sheet = wb.getSheetAt(0);
225. XSSFSheet sheet1 = wb.getSheetAt(2);
226. XSSFSheet sheet2 = wb.getSheetAt(3);
228. FormulaEvaluator fe = wb.getCreationHelper().createFormulaEvaluator();
229. int a = 0, check = 0;
231. for (Row row: sheet1) {
232. for (org.apache.poi.ss.usermodel.Cell c: row) {
233. //System.out.println("Cell is at "+c.getColumnIndex()+"
234. "+c.getRowIndex()+"
235. "+c.toString());
236. if (c.getColumnIndex() == 6 && c.getRowIndex() == 13) {
237. // System.out.println("Base Score is "+c.getNumericCellValue());
238. bs = (float) c.getNumericCellValue();
239. }
240. //switch (fe.evaluateInCell(c).getCellType())
241. }
242. }
243. for (Row row: sheet2) {
244. for (org.apache.poi.ss.usermodel.Cell c: row) {
245. if (c.getColumnIndex() == 7 && c.getRowIndex() == 13) {
246. // System.out.println("Environmental Score is "+c.getNumericCellValue());
247. es = (float) c.getNumericCellValue();
248. }
249. }
250. }
251. for (Row row: sheet) {
252. for (org.apache.poi.ss.usermodel.Cell c: row) {
253. switch (fe.evaluateInCell(c).getCellType()) {
254. case org.apache.poi.ss.usermodel.Cell.CELL\_TYPE\_STRING:
255. //System.out.println("String is"+c.getStringCellValue()+"\t\t");
256. if (c.getStringCellValue().matches("test
257. 1 "))
258. {
259. System.out.println("Matched");
260. a = 1;
261. } else if (c.getStringCellValue().matches("test 2")) {
262. a = 2;
263. } else if (c.getStringCellValue().matches("test 3")) {
264. a = 3;
265. } else if (c.getStringCellValue().matches("test 4")) {
266. a = 4;
267. } else if (c.getStringCellValue().matches("test 5")) {
268. a = 5;
269. }
271. break;
272. case org.apache.poi.ss.usermodel.Cell.CELL\_TYPE\_NUMERIC:
273. if (a == 1) {
274. //System.out.println("Values is "+c.getNumericCellValue()+"\t\t");
275. check++;
276. if (check == 1) {
277. a1 = (float) c.getNumericCellValue();
278. // System.out.println("Pre is "+a1);
279. }
280. if (check == 2) {
281. a2 = (float) c.getNumericCellValue();
282. // System.out.println("Actual is "+a2);
283. }
284. else
285. if (check == 3) {
286. a3 = (float) c.getNumericCellValue();
287. // System.out.println("Post is "+a3);
288. check = 0;
289. testdo(a1, a2, a3, 1);
290. }
291. } else if (a == 2) {
292. //System.out.println("Values is "+c.getNumericCellValue()+"\t\t");
293. check++;
294. if (check == 1) {
295. a1 = (float) c.getNumericCellValue();
296. // System.out.println("Pre is "+a1);
297. }
298. if(check == 2) {
299. a2 = (float) c.getNumericCellValue();
300. // System.out.println("Actual is "+a2);
301. } else if (check == 3) {
302. a3 = (float) c.getNumericCellValue();
303. //System.out.println("Post is "+a3);
304. check = 0;
305. testdo(a1, a2, a3, 2);
306. }
307. } else if (a == 3) {
308. //System.out.println("Values is "+c.getNumericCellValue()+"\t\t");
309. check++;
310. if (check == 1) {
311. a1 = (float) c.getNumericCellValue();
312. // System.out.println("Pre is "+a1);
313. }
314. if(check == 2) {
315. a2 = (float) c.getNumericCellValue();
316. // System.out.println("Actual is "+a2);
317. } else if (check == 3) {
318. a3 = (float) c.getNumericCellValue();
319. //System.out.println("Post is "+a3);
320. check = 0;
321. testdo(a1, a2, a3, 3);
322. }
323. } else if (a == 4) {
324. //System.out.println("Values is "+c.getNumericCellValue()+"\t\t");
325. check++;
326. if (check == 1) {
327. a1 = (float) c.getNumericCellValue();
328. // System.out.println("Pre is "+a1);
329. }
330. if(check == 2) {
331. a2 = (float) c.getNumericCellValue();
332. // System.out.println("Actual is "+a2);
333. } else if (check == 3) {
334. a3 = (float) c.getNumericCellValue();
335. //System.out.println("Post is "+a3);
336. check = 0;
337. testdo(a1, a2, a3, 4);
338. }
339. } else if (a == 5) {
340. //System.out.println("Values is "+c.getNumericCellValue()+"\t\t");
341. check++;
342. if (check == 1) {
343. a1 = (float) c.getNumericCellValue();
344. // System.out.println("Pre is "+a1);
345. }
346. if(check == 2) {
347. a2 = (float) c.getNumericCellValue();
348. // System.out.println("Actual is "+a2);
349. } else if (check == 3) {
350. a3 = (float) c.getNumericCellValue();
351. //System.out.println("Post is "+a3); //check=0;
352. testdo(a1, a2, a3, 5);
353. }
354. }
355. break;
356. default:
357. break;
358. }
359. }
360. }
361. }
362. ///direct working
363. void basescore(float s, float m, float v) {
365. float smv, impact;
366. smv = s \* m \* v \* 20;
367. System.out.println("SMV :" + smv);
369. impact = (float)(10.41 \* (1 - (1 - 0.66) \* (1 - 0.275) \* (1 - 0.275)));
370. System.out.println("Impact is :" + impact);
372. base\_score = (float)((0.6 \* impact + 0.4 \* smv - 1.5) \* 1.176);
373. System.out.println("Base score is " + base\_score);
374. }
375. //void environmentalscore(float )
376. private void testdo(float a1, float a2, float a3, int v) {
377. // TODO Auto-generated method stub
378. float m;
379. float add;
380. float sa1, sa2, sa3;
381. float sra1, sra2, sra3, tr;
382. if (v == 1) {
383. System.out.println("<---------------------------------------->");
384. System.out.println("Test 1");
385. System.out.println("<---------------------------------------->");
386. System.out.println("Pre is " + a1);
387. System.out.println("Actual is " + a2);
388. System.out.println("Post is " + a3);
389. System.out.println("<---------------------------------------->");
390. }
391. e lse
392. if (v == 2) {
393. System.out.println("<---------------------------------------->");
394. System.out.println("Test 2");
395. System.out.println("<---------------------------------------->");
396. System.out.println("Pre is " + a1);
397. System.out.println("Actual is " + a2);
398. System.out.println("Post is " + a3);
399. System.out.println("<---------------------------------------->");
400. }
401. e lse
402. if (v == 3) {
403. System.out.println("<---------------------------------------->");
404. System.out.println("Test 3");
405. System.out.println("<---------------------------------------->");
406. System.out.println("Pre is " + a1);
407. System.out.println("Actual is " + a2);
408. System.out.println("Post is " + a3);
409. System.out.println("<---------------------------------------->");
410. }
411. e lse
412. if (v == 4) {
413. System.out.println("<---------------------------------------->");
414. System.out.println("Test 4");
415. System.out.println("<---------------------------------------->");
416. System.out.println("Pre is " + a1);
417. System.out.println("Actual is " + a2);
418. System.out.println("Post is " + a3);
419. System.out.println("<---------------------------------------->");
420. }
421. e lse
422. if (v == 5) {
423. System.out.println("<---------------------------------------->");
424. System.out.println("Test 5");
425. System.out.println("<---------------------------------------->");
426. System.out.println("Pre is " + a1);
427. System.out.println("Actual is " + a2);
428. System.out.println("Post is " + a3);
429. System.out.println("<---------------------------------------->");
430. }
431. add = a1 + a2 + a3;
432. m = a1 + a2 + a3;
433. m = Float.parseFloat(format(m, 2));
434. System.out.println("Add is " + m);
435. m = m / 3;
436. m = Float.parseFloat(format(m, 2));
437. System.out.println("Mean is " + m);
438. //m=format(m);
439. sa1 = a1 - m;
440. sa2 = a2 - m;
441. sa3 = a3 - m;
442. sa1 = Float.parseFloat(format(sa1, 2));
443. sa2 = Float.parseFloat(format(sa2, 2));
444. sa3 = Float.parseFloat(format(sa3, 2));
445. System.out.println("First Value is " + sa1);
446. System.out.println("Second Value is " + sa2);
447. System.out.println("Third Value is " + sa3);
448. sra1 = (float) Math.pow(sa1, 2);
449. sra2 = (float) Math.pow(sa2, 2);
450. sra3 = (float) Math.pow(sa3, 2);
451. System.out.println("Square of first " + sra1);
452. System.out.println("Square of Second " + sra2);
453. System.out.println("Square of Third " + sra3);
455. tr = sra1 + sra2 + sra3;
456. tr = (float)(tr \* 0.4);
457. tr = (float) Math.sqrt(tr);
458. tr = (float)(tr + 0.0001);
459. System.out.println("Original Standard Devation is:" + tr);
460. tr = Float.parseFloat(format(tr, 5));
461. //System.out.println("Affter conversion SD "+tr);
462. if (tr == 0) {
463. System.out.println("Standard Devation(If Zero):" + 1);
464. } else {
465. //System.out.println("Standard Devation is:"+tr);
466. System.out.println("Affter conversion SD " + tr);
467. }
468. System.out.println("No. of Measures:" + 1);
470. System.out.println("Original Sum is:" + add);
471. add = Float.parseFloat(format(add, 5));
472. System.out.println("5 DP sum is " + add);
474. System.out.println("5 DP Ratio of Devaition:" + add / tr);
476. float rsf = Float.parseFloat(format(tr, 5));
477. System.out.println("5 DP Raw Score Factor :" + rsf);
478. float median = a1 + a3;
479. median = median / 2;
480. System.out.println("Raw Score Factor second column " + median);
481. median = Float.parseFloat(format(median, 5));
482. System.out.println("5 DP Raw Score Factor second column " + median);
483. float range\_lowest\_dev = (float)(rsf \* 0.25);
484. range\_lowest\_dev = (float)(median - range\_lowest\_dev);
485. System.out.println("Range Lowest Daevation " + range\_lowest\_dev);
486. range\_lowest\_dev = Float.parseFloat(format(range\_lowest\_dev, 5));
487. System.out.println("5 DP Range Lowest Daevation " + range\_lowest\_dev);
488. float range\_lowest\_dev2 = (float)(rsf \* .25);
489. range\_lowest\_dev2 = (float)(median + range\_lowest\_dev2);
490. System.out.println("Range Lowest Daevation second column " + range\_lowest\_dev2);
491. range\_lowest\_dev2 = Float.parseFloat(format(range\_lowest\_dev2, 5));
492. System.out.println("5 DP Range Lowest Daevation second column " + range\_lowest\_dev2);
493. Float range\_low\_dev = (float)(rsf \* 0.66);
494. range\_low\_dev = range\_lowest\_dev - range\_low\_dev;
495. System.out.println("Range\_Low\_Devation " + range\_low\_dev);
496. range\_low\_dev = Float.parseFloat(format(range\_low\_dev, 5));
497. System.out.println("5 DP Range\_Low\_Devation " + range\_low\_dev);
498. Float range\_low\_dev2 = (float)(rsf \* 0.66);
499. range\_low\_dev2 = range\_low\_dev2 + range\_lowest\_dev2;
500. System.out.println("Range Low Devation second column :" + range\_low\_dev2);
501. range\_low\_dev2 = Float.parseFloat(format(range\_low\_dev2, 5));
502. System.out.println("5 DP Range Low Devation second column :" + range\_low\_dev2);
503. Float range\_partial\_dev = rsf \* 1;
504. range\_partial\_dev = range\_low\_dev - range\_partial\_dev;
505. System.out.println("Range\_Partial\_Devation " + range\_partial\_dev);
506. range\_partial\_dev = Float.parseFloat(format(range\_partial\_dev, 5));
507. System.out.println("5 DP Range\_Partial\_Devation " + range\_partial\_dev);
508. Float range\_partial\_dev2 = rsf \* 1;
509. range\_partial\_dev2 = range\_partial\_dev2 + range\_low\_dev2;
510. System.out.println("Range\_Partial\_Devation second column " + range\_partial\_dev2);
511. range\_partial\_dev2 = Float.parseFloat(format(range\_partial\_dev2, 5));
512. System.out.println("5 DP Range\_Partial\_Devation second column " + range\_partial\_dev2);
513. Float range\_high\_dev = (float)(rsf \* 1.5);
514. range\_high\_dev = range\_partial\_dev - range\_high\_dev;
515. System.out.println("Range\_High\_Devation " + range\_high\_dev);
516. range\_high\_dev = Float.parseFloat(format(range\_high\_dev, 5));
517. System.out.println("5 DP Range\_High\_Devation " + range\_high\_dev);
518. Float range\_high\_dev2 = (float)(rsf \* 1.5);
519. range\_high\_dev2 = range\_partial\_dev2 + range\_high\_dev2;
520. System.out.println("Range\_High\_Devation second column " + range\_high\_dev2);
521. range\_high\_dev2 = Float.parseFloat(format(range\_high\_dev2, 5));
522. System.out.println("5 DP Range\_High\_Devation second column " + range\_high\_dev2);
523. float rawscorepree = raw\_score(a1, range\_high\_dev, range\_partial\_dev, range\_low\_dev, range\_lowest\_dev, ra nge\_high\_dev2, range\_partial\_dev2, range\_low\_dev2, range\_lowest\_dev2);
524. System.out.println("Raw Score of Pre " + rawscorepree);
525. rawscorepree = Float.parseFloat(format(rawscorepree, 5));
526. System.out.println("5 DP Raw Score of Pre " + rawscorepree);
527. float rawscoreactual = raw\_score(a2, range\_high\_dev, range\_partial\_dev, range\_low\_dev, range\_lowest\_dev, range\_high\_dev2, range\_partial\_dev2, range\_low\_dev2, range\_lowest\_dev2);
528. System.out.println("Raw Score of Actual " + rawscoreactual);
529. rawscoreactual = Float.parseFloat(format(rawscoreactual, 5));
530. System.out.println("5 DP Raw Score of Actual " + rawscoreactual);
531. float rawscorepost = raw\_score(a3, range\_high\_dev, range\_partial\_dev, range\_low\_dev, range\_lowest\_dev, ra nge\_high\_dev2, range\_partial\_dev2, range\_low\_dev2, range\_lowest\_dev2);
532. System.out.println("Raw Score of Post " + rawscorepost);
533. rawscorepost = Float.parseFloat(format(rawscorepost, 5));
534. System.out.println("5 DP Raw Score of Post " + rawscorepost);
535. //clear up
536. bs = (float) 5.31;
537. System.out.println("Base Score of Pre " + bs);
538. System.out.println("Base Score of Actual " + bs);
539. System.out.println("Base Score of Post " + bs);
540. bs = Float.parseFloat(format(bs, 5));
541. System.out.println("5 DP Base Score of Pre " + bs);
542. System.out.println("5 DP Base Score of Actual " + bs);
543. System.out.println("5 DP Base Score of Post " + bs);
544. es = (float) 3.53;
545. System.out.println("Environmental Score of Pre " + es);
546. System.out.println("Environmental Score of Actual " + es);
547. System.out.println("Environmental Score of Post " + es);
548. es = Float.parseFloat(format(es, 5));
549. System.out.println("5 DP Environmental Score of Pre " + es);
550. System.out.println("5 DP Environmental Score of Actual " + es);
551. System.out.println("5 DP Environmental Score of Post " + es);
552. Float final\_wighted\_measure1 = rawscorepree + bs + es;
553. System.out.println("Final Weighted Measure of Pre " + final\_wighted\_measure1);
554. final\_wighted\_measure1 = Float.parseFloat(format(final\_wighted\_measure1, 5));
555. System.out.println("5 DP Final Weighted Measure of Pre " + final\_wighted\_measure1);
556. Float final\_wighted\_measure2 = rawscoreactual + bs + es;
557. System.out.println("Final Weighted Measure of Actual " + final\_wighted\_measure2);
558. final\_wighted\_measure2 = Float.parseFloat(format(final\_wighted\_measure2, 5));
559. System.out.println("5 DP Final Weighted Measure of Actual " + final\_wighted\_measure2);
560. Float final\_wighted\_measure3 = rawscorepost + bs + es;
561. System.out.println("Final Weighted Measure of Post " + final\_wighted\_measure3);
562. final\_wighted\_measure3 = Float.parseFloat(format(final\_wighted\_measure3, 5));
563. System.out.println("5 DP Final Weighted Measure of Post " + final\_wighted\_measure3);
564. Float basescore\_confidence\_rating2 = ((rawscorepree + bs) + (rawscoreactual + bs) + (rawscorepost + bs)) / 3;
565. float basescore\_confidence\_rating1 = basescoreconfidencerating1(basescore\_confidence\_rating2);
566. System.out.println("Base Score Confidence Rating is " + basescore\_confidence\_rating1);
567. lblNewLabel\_4.setText("" + basescore\_confidence\_rating1);
568. System.out.println("BaseScore Confidence Rating second column is                      "+basescore\_confidence\_rating2);
569. lblNewLabel\_6.setText("" + basescore\_confidence\_rating2);
570. basescore\_confidence\_rating1 = Float.parseFloat(format(basescore\_confidence\_rating1, 5));
571. System.out.println("5 DP Base Score Confidence Rating is
572. "+basescore\_confidence\_rating1);
573. basescore\_confidence\_rating2 = Float.parseFloat(format(basescore\_confidence\_rating2, 5));   
     System.out.println("5 DP BaseScore Confidence Rating second column is
574. "+basescore\_confidence\_rating2);
576. Float envir\_score\_confidence\_rating2 = ((rawscorepree + es) + (rawscoreactual + es) + (rawscorepost + es)) / 3;
577. float envirscore\_confidence\_rating1 = envirscoreconfidencerating1(envir\_score\_confidence\_rating2);
578. System.out.println("EnvirScore Confidence Rating is
579. "+envirscore\_confidence\_rating1);
580. lblNewLabel\_8.setText("" + envirscore\_confidence\_rating1);
581. System.out.println("EnvirScore Confidence Rating second column is
582. "+envir\_score\_confidence\_rating2);
583. lblNewLabel\_10.setText("" + envir\_score\_confidence\_rating2);
584. envirscore\_confidence\_rating1 = Float.parseFloat(format(envirscore\_confidence\_rating1, 5));
585. System.out.println("5 DP EnvirScore Confidence Rating is
586. "+envirscore\_confidence\_rating1);
588. envir\_score\_confidence\_rating2 = Float.parseFloat(format(envir\_score\_confidence\_rating2, 5));
589. System.out.println("5 DP EnvirScore Confidence Rating second column is
590. "+envir\_score\_confidence\_rating2);
591. float confidencerating2 = (basescore\_confidence\_rating2 + envir\_score\_confidence\_rating2) / 2;
592. float confidencerating1 = confidencerating1(confidencerating2);
593. System.out.println("Confidence Rating is " + confidencerating1); lblNewLabel\_12.setText("" + confidencerating1);
594. System.out.println("Confidence Rating second column is " + confidencerating2);
595. lblNewLabel\_14.setText("" + confidencerating2); confidencerating1 = Float.parseFloat(format(confidencerating1, 5));
596. System.out.println("5 DP Confidence Rating is " + confidencerating1);
597. confidencerating2 = Float.parseFloat(format(confidencerating2, 5));
598. System.out.println("5 DP Confidence Rating second column is " + confidencerating2);
599. }
600. public String format(Number n, int a) {
601. NumberFormat format = DecimalFormat.getInstance();
602. format.setRoundingMode(RoundingMode.FLOOR);
603. format.setMinimumFractionDigits(0);
604. format.setMaximumFractionDigits(a);
605. return format.format(n);
606. }
607. public float raw\_score(float c4, float m4, float l4, float k4, float j4, float m5, float l5, float k5, float j5) {
608. Float C4, M4, L4, K4, J4, M5, L5, K5, J5;
609. C4 = c4;
610. M4 = m4;
611. L4 = l4;
612. K4 = k4;
613. J4 = j4;
614. M5 = m5;
615. L5 = l5;
616. K5 = k5;
617. J5 = j5;
619. if (C4 < M4) {
620. return 1;
621. } else {
622. if (C4 >= M4 && C4 < L4) {
623. return (float) 7.75;
624. } else {
625. if (C4 >= L4 && C4 < K4) {
626. return (float) 5.5;
627. } else {
628. if (C4 >= K4 && C4 < J4) {
629. return (float) 3.25;
630. } else {
631. if (C4 >= J4 && C4 <= J5) {
632. return 1;
633. } else {
634. if (C4 > J5 && C4 <= K5) {
635. return (float) 3.25;
636. } else {
637. if (C4 > K5 && C4 <= L5) {
638. return (float) 5.5;
639. } else {
640. if (C4 > L5 && C4 <= M5) {
641. return (float) 7.75;
642. } else {
643. return 10;
644. }
645. }
646. }
647. }
648. }
649. }
650. }
651. }
652. }
653. public Float basescoreconfidencerating1(Float r5) {
654. Float R5;
655. R5 = r5;
656. if (R5 > 9) {
657. return (float) 1;
658. } else {
659. if (R5 > 7 && R5 <= 8.9999999) {
660. return (float) 2;
661. }
662. if (R5 > 4 && R5 <= 6.9999999) {
663. return (float) 3;
664. }
665. if (R5 > 0 && R5 <= 3.99999999) {
666. return (float) 4;
667. } else {
668. return (float) 5;
669. }
670. }
671. }
672. public float envirscoreconfidencerating1(float s5) {
673. float S5 = s5;
674. if (S5 > 9) {
675. return 1;
676. } else {
677. if (S5 > 7 && S5 <= 8.99999999) {
678. return 2;
679. }
680. if (S5 > 4 && S5 <= 6.99999999) {
681. return 3;
682. }
683. if (S5 > 0 && S5 <= 3.99999999) {
684. return 4;
685. } else {
686. return 0;
687. }
688. }
689. }
690. public float confidencerating1(float t5) {
691. float T5;
692. T5 = t5;
693. if (T5 > 9) {
694. return 1;
695. }
696. e lse {
697. if (T5 > 7 && T5 <= 8.99999999999) {
698. return 2;
699. }
700. i f(T5 > 4 && T5 <= 6.999999999999) {
701. return 3;
702. }
703. i f(T5 > 0 && T5 <= 3.9999999999999) {
704. return 4;
705. }
706. e lse {
707. return 0;
708. }
709. }
710. }
711. }

# Appendix C – Framework OpenDaylight Controller Application

1. import java.io.BufferedReader;
2. import java.io.InputStream;
3. import java.io.InputStreamReader;
4. import java.net.HttpURLConnection;
5. import java.net.URL;
6. import java.util.concurrent.Future;
7. import java.util.concurrent.ScheduledExecutorService;
8. import java.util.concurrent.ScheduledFuture;
9. import org.apache.commons.codec.binary.Base64;
10. import org.codehaus.jettison.json.JSONObject;
11. import org.opendaylight.openflowplugin.applications.lldpspeaker.LLDPSpeaker;
12. import org.opendaylight.openflowplugin.applications.lldpspeaker.LLDPUtil;
13. import org.opendaylight.yang.gen.v1.urn.ietf.params.xml.ns.yang.ietf.yang.types.rev130715.MacAddre ss;
14. import org.opendaylight.yang.gen.v1.urn.opendaylight.flow.inventory.rev130819.FlowCapableNodeConne ctor;
15. import org.opendaylight.yang.gen.v1.urn.opendaylight.inventory.rev130819.NodeConnector;
16. import org.opendaylight.yang.gen.v1.urn.opendaylight.inventory.rev130819.NodeConnectorId;
17. import org.opendaylight.yang.gen.v1.urn.opendaylight.inventory.rev130819.NodeConnectorRef;
18. import org.opendaylight.yang.gen.v1.urn.opendaylight.inventory.rev130819.NodeId;
19. import org.opendaylight.yang.gen.v1.urn.opendaylight.inventory.rev130819.NodeRef;
20. import org.opendaylight.yang.gen.v1.urn.opendaylight.packet.service.rev130709.PacketProcessingServi ce;
21. import org.opendaylight.yang.gen.v1.urn.opendaylight.packet.service.rev130709.TransmitPacketInput;
22. import org.opendaylight.yang.gen.v1.urn.opendaylight.packet.service.rev130709.TransmitPacketInputBu ilder;
23. import org.opendaylight.yang.gen.v1.urn.opendaylight.params.xml.ns.yang.openflow.applications.lldp.speaker.rev141023.OperStatus;
24. import org.opendaylight.yangtools.yang.binding.InstanceIdentifier;
25. import org.opendaylight.yangtools.yang.common.RpcResult;
26. public class testopendayy extends LLDPSpeaker implements PacketProcessingService
27. {
28. // private static final FlowCapableNodeConnector fcnc; // private static final TransmitPacketInput packet;
29. static {
30. MacAddress mac = new MacAddress("01:23:45:67:89:AB");
31. }
32. private PacketProcessingService packetProcessingService;
33. private ScheduledExecutorService scheduledExecutorService;
34. private ScheduledFuture scheduledSpeakerTask;
35. private final MacAddress destinationMACAddress = null;
36. private LLDPSpeaker lldpSpeaker;
38. public testopendayy(PacketProcessingService packetProcessingService, MacAddress addressDestionation)
39. {
40. super(packetProcessingService, addressDestionation);
41. }
42. public static JSONObject getNodes(String user, String password, String baseURL) {
43. StringBuffer result = new StringBuffer();
44. try {
45. if (!baseURL.contains("http")) {
46. System.out.println("Not containing");
47. baseURL = "http://" + baseURL;
48. }
49. baseURL = baseURL + "/controller/nb/v2/switchmanager/default/nodes";
50. System.out.println(baseURL);
51. // Create URL = base URL + container
52. URL url = new URL(baseURL);
53. // Create authentication string and encode it to Base64
54. String authStr = user + ":" + password;
55. String encodedAuthStr = Base64.encodeBase64String(authStr.getBytes());
56. // Create Http connection
57. HttpURLConnection connection = (HttpURLConnection) url.openConnection();
58. // Set connection properties
59. connection.setRequestMethod("GET");
60. connection.setRequestProperty("Authorization", "Basic " + encodedAuthStr);
61. connection.setRequestProperty("Accept", "application/json");
62. System.out.println(connection + "");
63. // Get the response from connection's inputStream
64. InputStream content = (InputStream) connection.getInputStream();
65. BufferedReader in = new BufferedReader(new InputStreamReader(content));
66. // result.append("{");
67. String line = "";
68. while ((line = in .readLine()) != null) {
69. result.append(line);
70. System.out.println(result);
71. }
72. // System.out.println("org.codehaus.jettison.json.JSONException: A
73. JSONObject text must begin with '{'
74. at character 1 of "+result.toString());
75. JSONObject nodes = new JSONObject(result.toString());
76. System.out.println(nodes + "");
77. //Future<RpcResult<java.lang.Void>> transmitPacket(TransmitPacketInput input);
78. // JSONArray jsonArray=new JSONArray(result.toString())
79. return nodes;
80. } catch (Exception e) {
81. e.printStackTrace();
82. // e.getMessage();
83. }
84. return null;
85. }
86. void test()
87. {
88. lldpSpeaker = new LLDPSpeaker(packetProcessingService, scheduledExecutorService, destinationMACAddress);
89. lldpSpeaker.setOperationalStatus(OperStatus.RUN);
90. }
91. public void testStandBy() {
92. lldpSpeaker.setOperationalStatus(OperStatus.STANDBY); // Add node connector - LLDP packet should be transmitted through // packetProcessingService // lldpSpeaker.nodeConnectorAdded(id, fcnc); // // Execute one iteration of periodic task - LLDP packet should be // transmitted second time
93. lldpSpeaker.run();
94. // Check packet transmission // verify(packetProcessingService, times(1)).transmitPacket(packet); // verifyNoMoreInteractions(packetProcessingService);
95. }
96. public void testNodeConnectorAdd() { // Add node connector - LLDP packet should be transmitted through // packetProcessingService // lldpSpeaker.nodeConnectorAdded(id, fcnc); // Execute one iteration of periodic task - LLDP packet should be // transmitted second time
97. lldpSpeaker.run();
98. // Check packet transmission // verify(packetProcessingService, times(2)).transmitPacket(packet); // verifyNoMoreInteractions(packetProcessingService);
99. }
100. @Override
101. public Future < RpcResult < Void >> transmitPacket(TransmitPacketInput arg0) { // TODO Auto-generated method stub
102. return null;
103. }
104. }

# Appendix D – SMV Survey & Results

Survey of Network Security Metrics Utility

Research:   
We intend to design and implement a framework for SDN network security confidence analysis. In order to enhance traditional analysis, we utilize SDN in two key areas; route and destination verification and switch metrics analysis. This framework will allow SDN authentication applications to validate and verify the routing and destination of data as well as assess the network devices for unexpected behavior (i.e. data compromise, man-in the middle attacks, etc.)

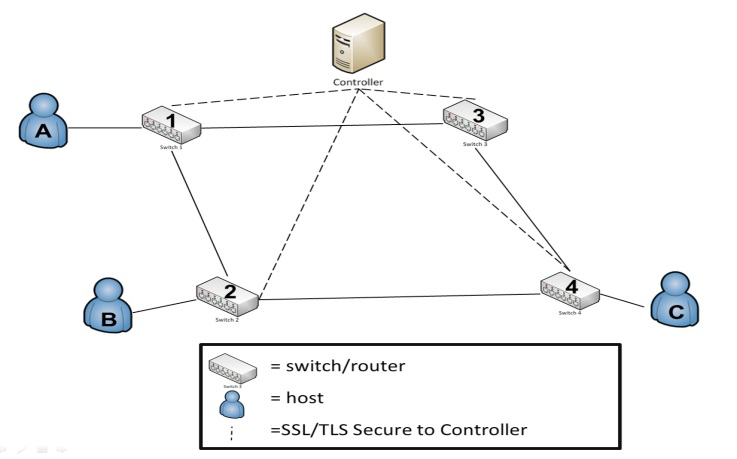
Purpose of Survey:

In order to better assess the value and qualities of these various networking metrics, we would like your evaluation.

Sample Scenario:

Use the Scenario and Figure below to answer the questions on this survey:

To better highlight the metrics collected and how they may be analyzed, reference the example network topology in the figure below and assume the host A is sending a sensitive message to host C.



General Criteria:

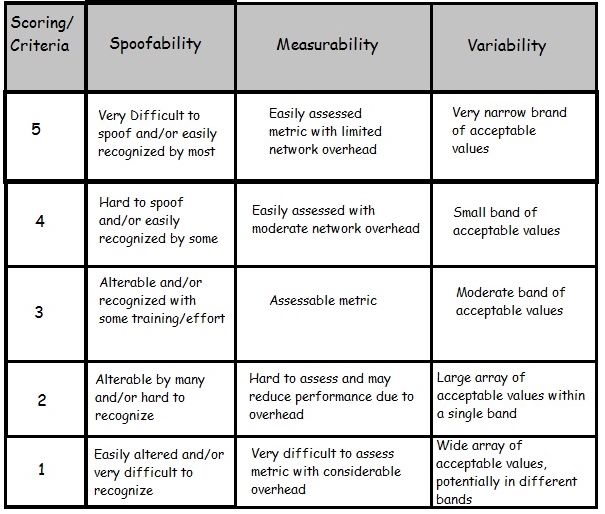
For assessing the meters/metrics in general, we propose the following criteria:

S: Spoofability--this measures the ability of the metric in general to be falsified in some manner.

M: Measurability – the degree of exactness that SDN allows for measure (whether subjective or objective metric)

V: Variability – measures the range of acceptable values that would be considered within bounds for a given metric.

General Criteria Grid:



Survey Questions

Example SDN Topology:

Variables:   
*P = Packet*

*Cr = SDN Controller  
F = Flow*

*T = Time  
x,y,z = Transmission Number*

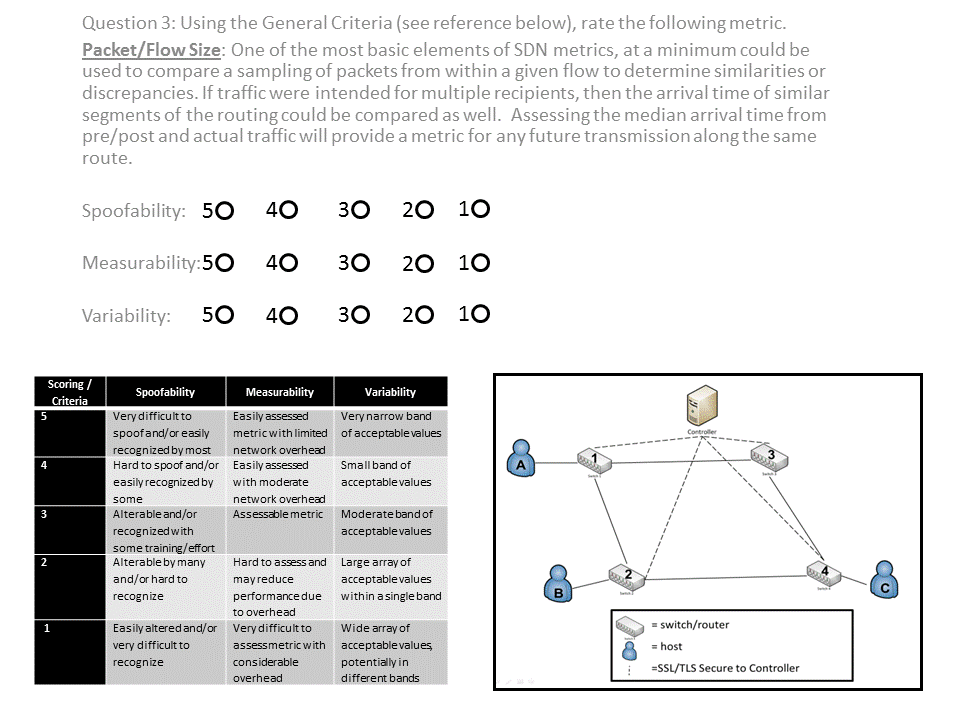
*n = Size/Bytes*

1. Route Verification

Utilizing SDN to provide and compare the expected route (from the controller's flow table) to the actual route that data flows from the switch forwarding rules. For example, a packet (Px) traveling from Host A to Host C (Reference the diagram above as needed).

Path: PRx, -> A, 1, Cr, 1, 2, 4, C

Verification Track: Px -> A,1,Cr,1,2,Cr,2,4,Cr,4,C

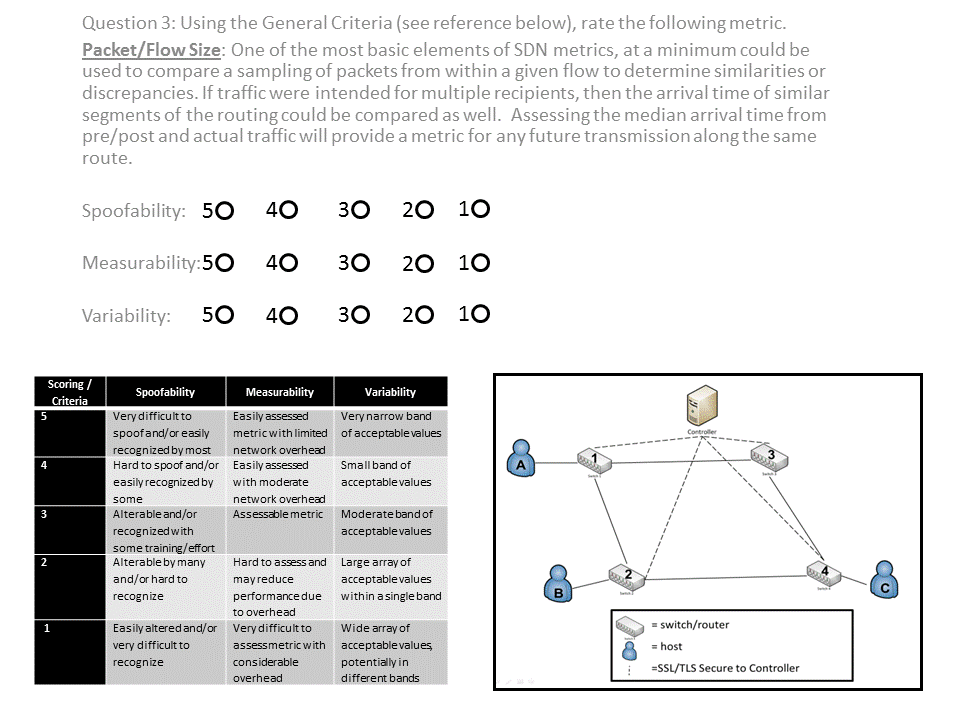
****

2. Packet/Flow Size

Measure the size of the first and last packets. Packet size can then be compared as the data flows from switch to switch and from each of the three elements of message traffic (pre/post and actual)

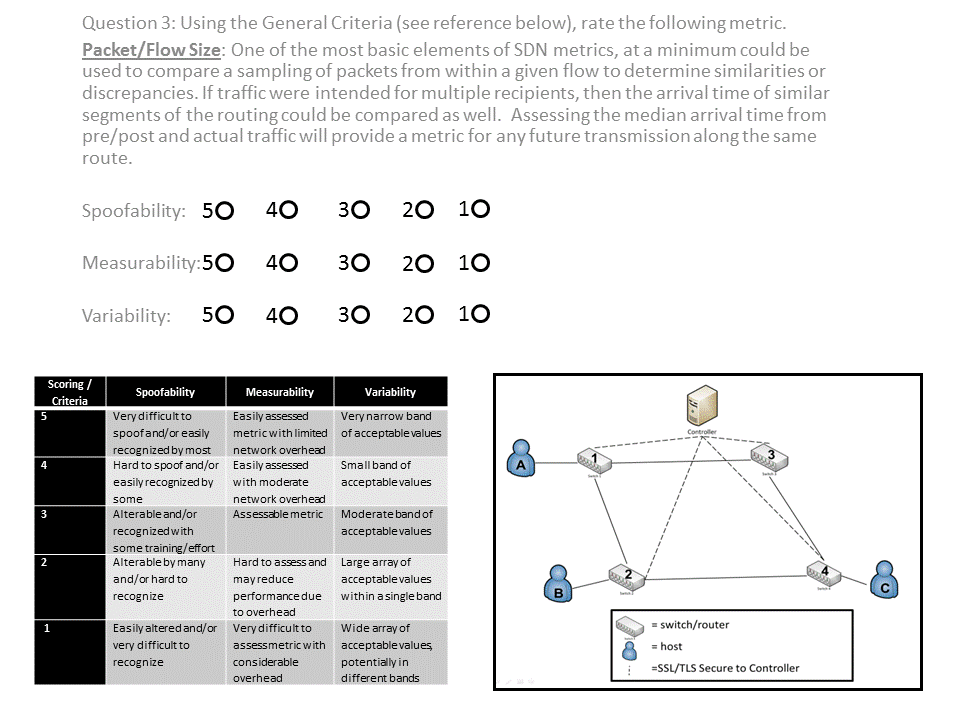
(Px1 + Px2 + Px4) , (Py1 + Py2 + Py4)U , (Px1 + Px2 + Px4)

(Py1 + Py2 + Py4) (Pz1 + Pz2 + Pz4) (Pz1 + Pz2 + Pz4)

****

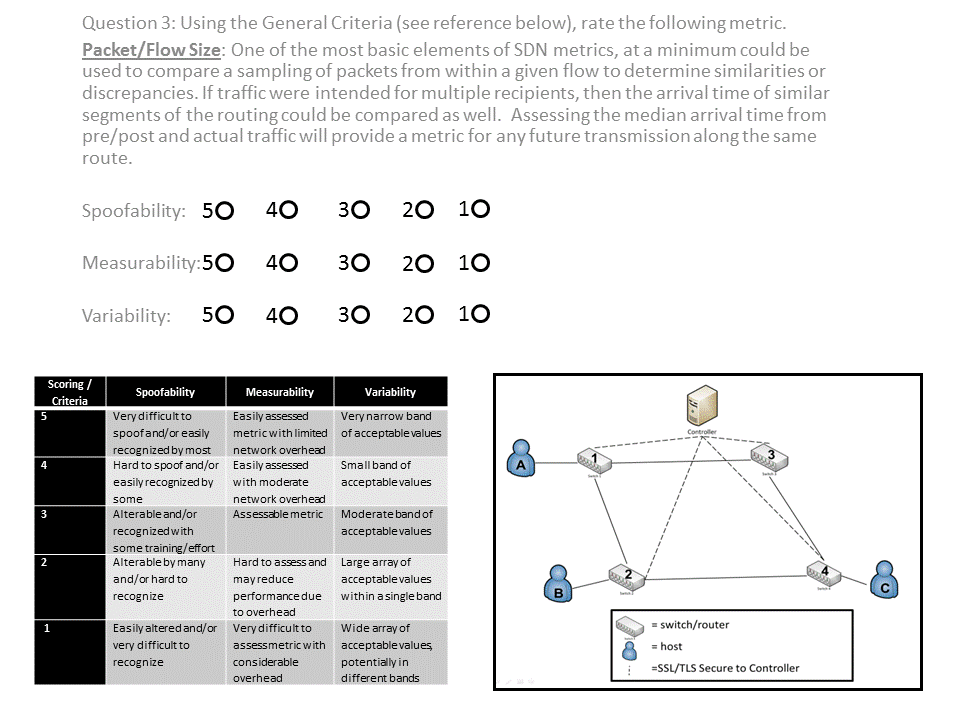
3. Packet Arrival Time to Controller

Assessing the median arrival time from pre/post and actual traffic will provide a metric for any future transmission along the same route. If traffic were intended for multiple recipients, then the arrival time of similar segments of the routing could be compared as well.

****

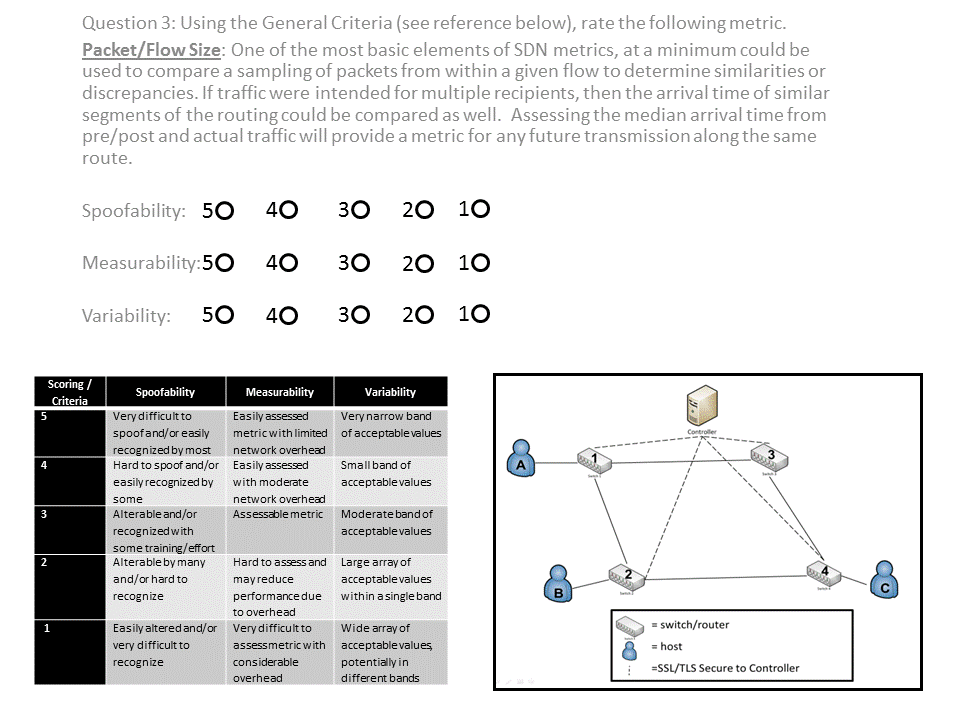
4. Packet/Flow Lapse Time

Measuring the packets arrival at two switches provides metrics that should be validated with performance traceroute to assess speed and detect man in the middle attacks.

****

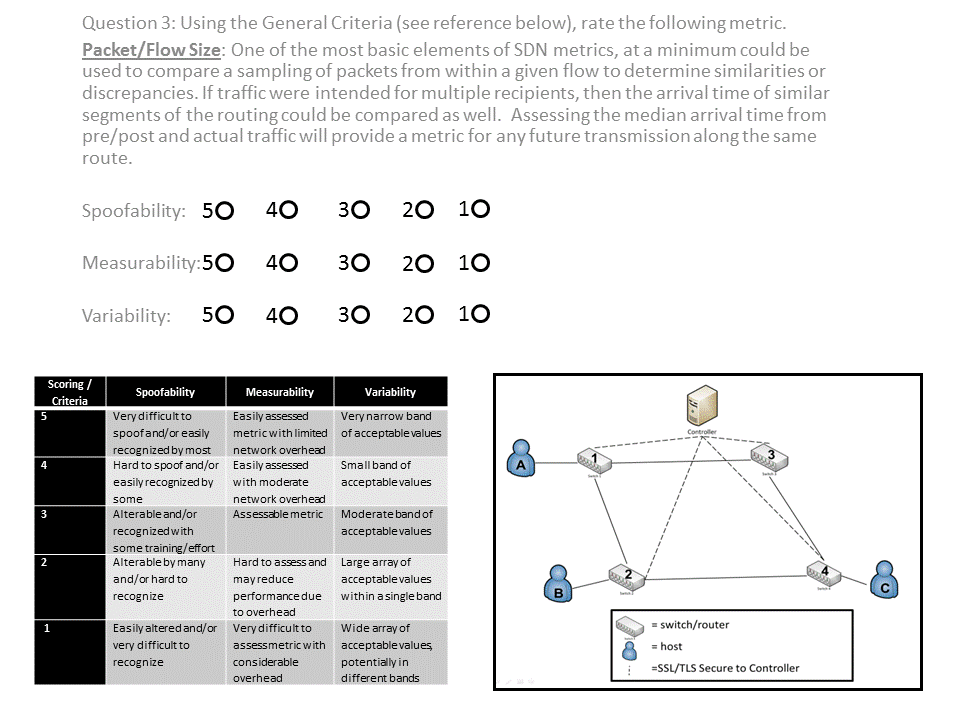
5. Packet/Flow Duration

The arrival time of the first packet until the flow exits the switch. Knowing flow sizes and types/protocol, we could compare the traffic against performance tests and standardized metrics to assess and create expectations for transport time and routing. Comparing the time a flow spends in the Flow Table may help determine if a high volume of malicious data is traveling utilizing the same flow table entry (i.e. many flows, but few packets).

****

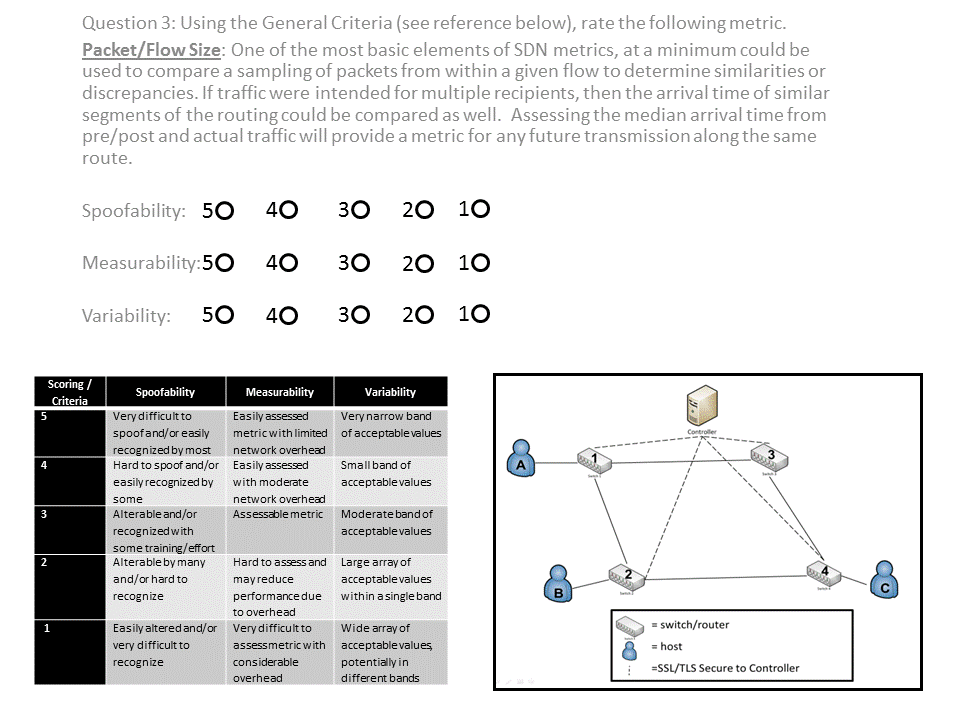
6. Hop Count

This is a simple metric and is available conventionally; however, the SDN controller could have a much more accurate estimation of the hop count per recipient. This estimation helps validate the path, eliminate routing to devices well outside of the network/system control, and would have limited overhead. It is important to note that hop count can be changed by the controller/installed Flow Entry Action.

****

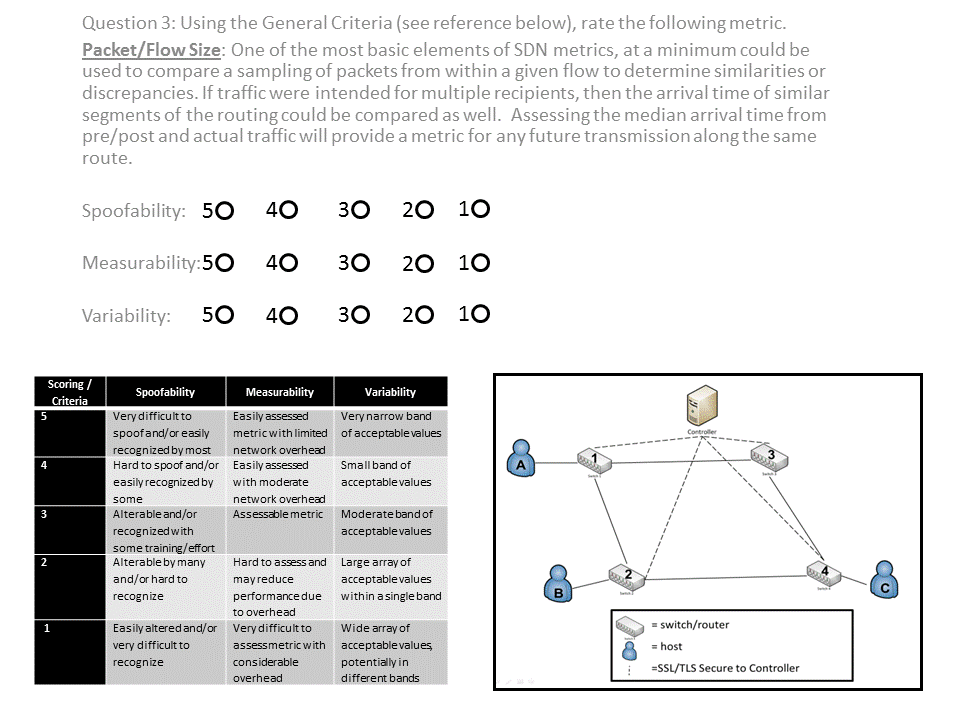
7. Switch/Device Location

This refers to the geospatial or at least time zone location of a device. This could be used with varying degrees of trust to assess the strength of a partial route. Some larger level (ISP) or company internal switches would have a higher level of trust versus the open internet. Location data such as an authorized IP range could identify a switch/network owner, combined with an external entity like IANA or a business IT department.

****

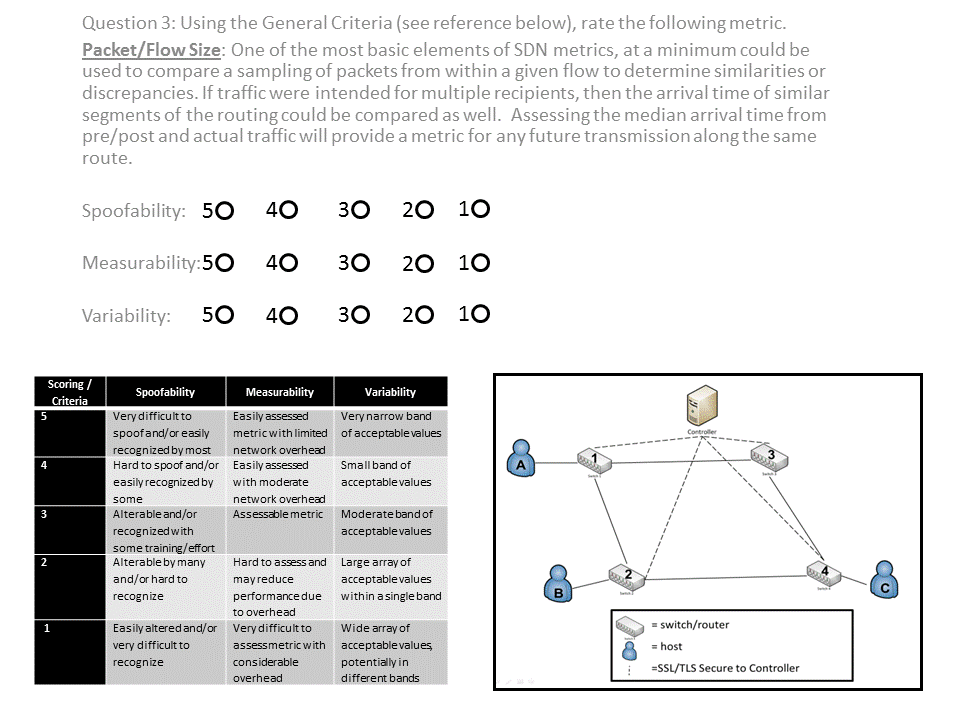
8. Switch/Device Characteristics

Knowing the type of switch, level within the LAN/WAN hierarchy, and switch owner could all be utilized to develop an algorithm for trust with the network. For example physical vs. virtual switches – typically a physical switch would have a higher degree of trust as it is harder to hack/spoof.

****

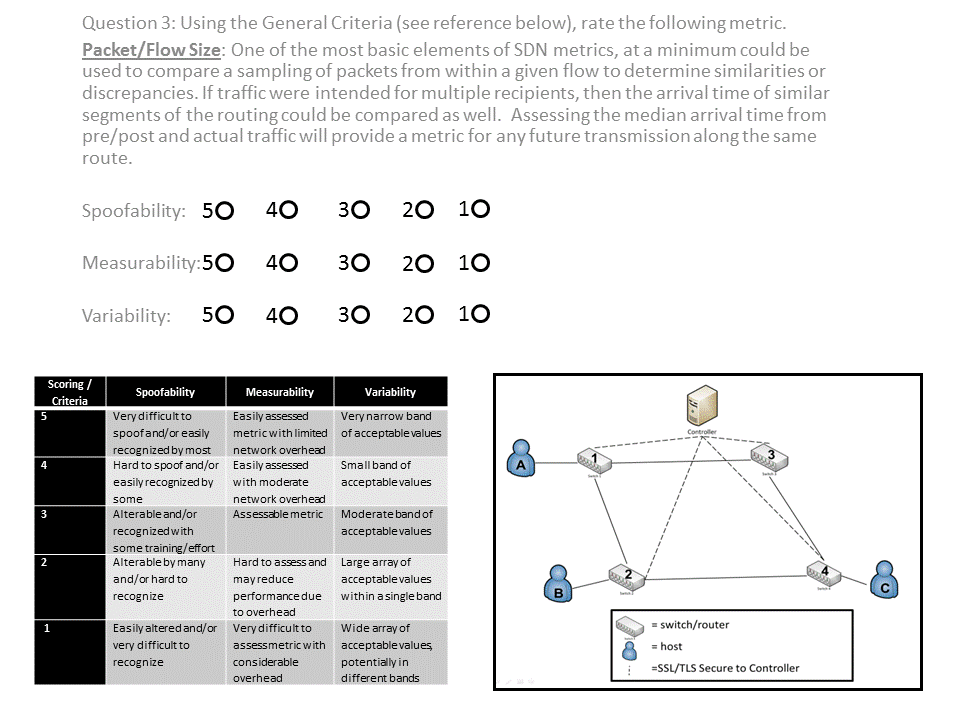
9. Sender/Receiver Role Based-Access

Linking Active Directory, RADIUS Server, etc., traffic could be validated by message type, access of the users and/or user group, pull in location data of the group and compare routing data. This data could also be held or queued if a user was identified as logged off, so it would not flow to the device until the user was logged in and available to receive it.

****

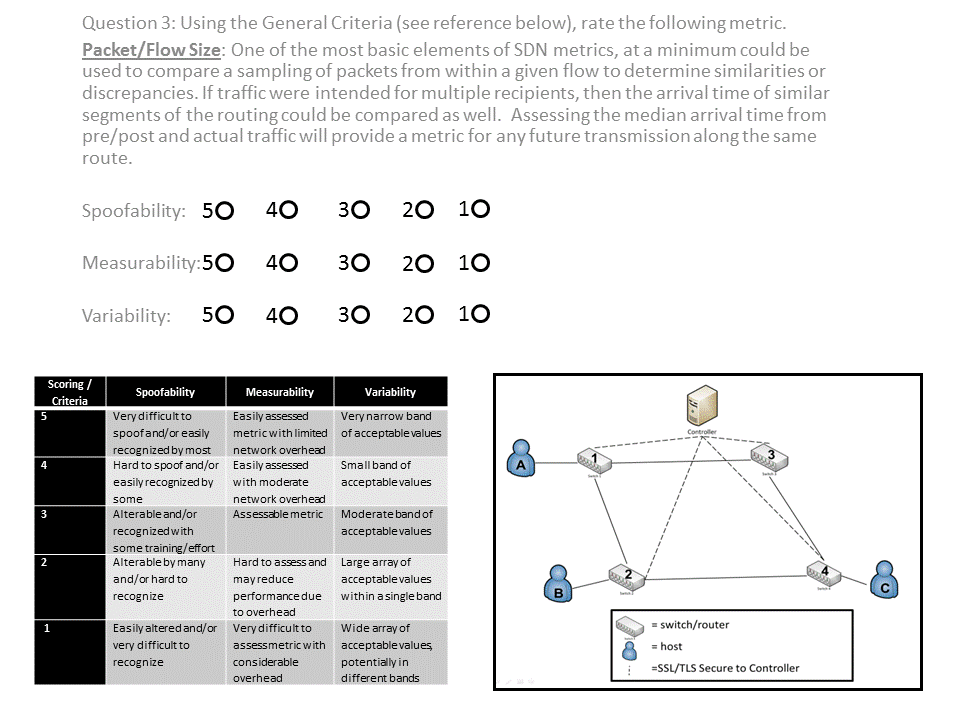
10. Average Number of Packets per Flow

Data transmission is a two-way street, so it is equally as important to ensure the safety of the receiving node from malicious attack. A side effect of malicious spoofing is the generation of flows with a small number of packets, i.e. about 3 packets per flow. Given that normal traffic usually involves a higher number of packets. If we can determine a median value for this, then we can assess confidence.

****

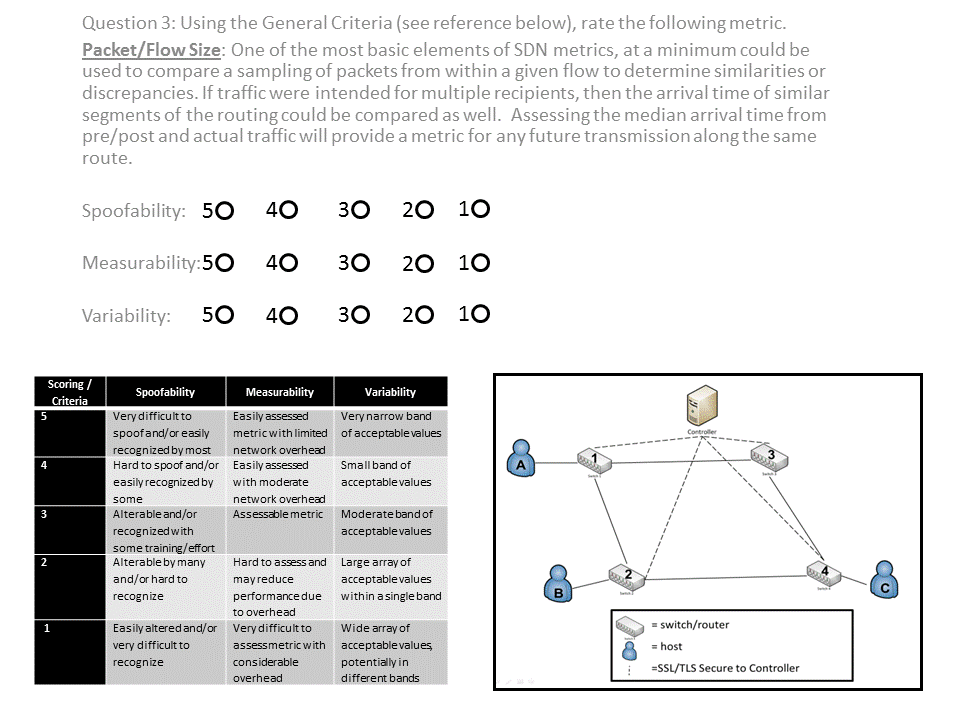
11. Median Bytes per Flow

Continuing on receiving node security, the attack payload size is often very small (for example TCP flooding attacks typically contain packets of ~100 bytes). If we can determine a median value for this, then we can assess confidence.

****

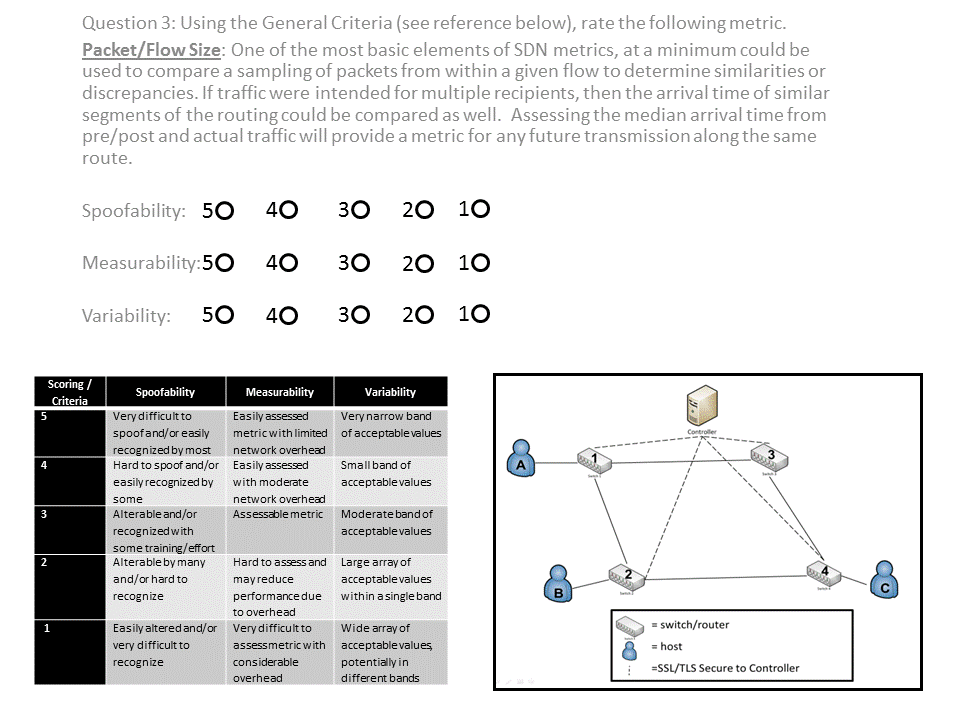
12. Growth of Single Flows

Verify how many pair-flows occur in the flow stream during a certain interval. Malicious activity often increases the number of single-flows into the network because they send packets with a fake IP.

****

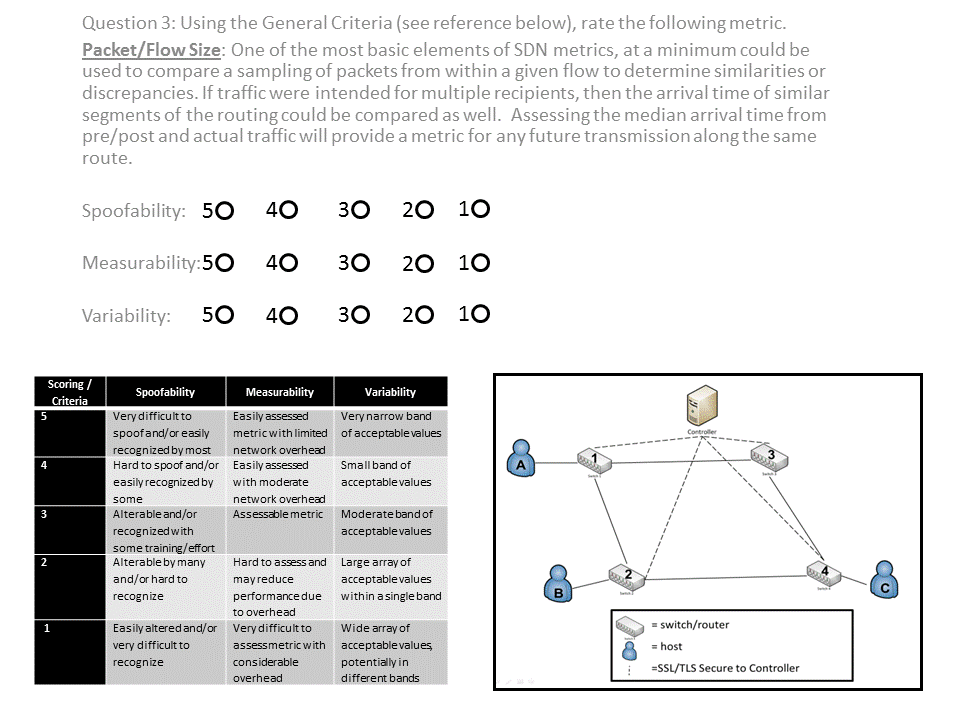
13. Packet Timestamp Comparison

By employing timestamping on the first packet in and last packet out of a given flow, we can assess exactly when traffic enters and exits SDN hybrid-networks. Based upon experience, we can identify if traffic is flowing at a different rate. Taking into account, any deviation would at a minimum degrade our confidence in the data path. Additionally, this could potentially hash the packet header and timestamp. Passing this hash to an authentication server (which knows the header) would validate the packet based upon the returned timestamp.

****

14. Percentage of Correlative Flows

The Destination has the capability/requirement to reply to packet request (whether legitimate or not). Under normal condition, the rate of traffic from the Destination back to the source is typically half, whereas illegitimate requests would see a near zero result.

****

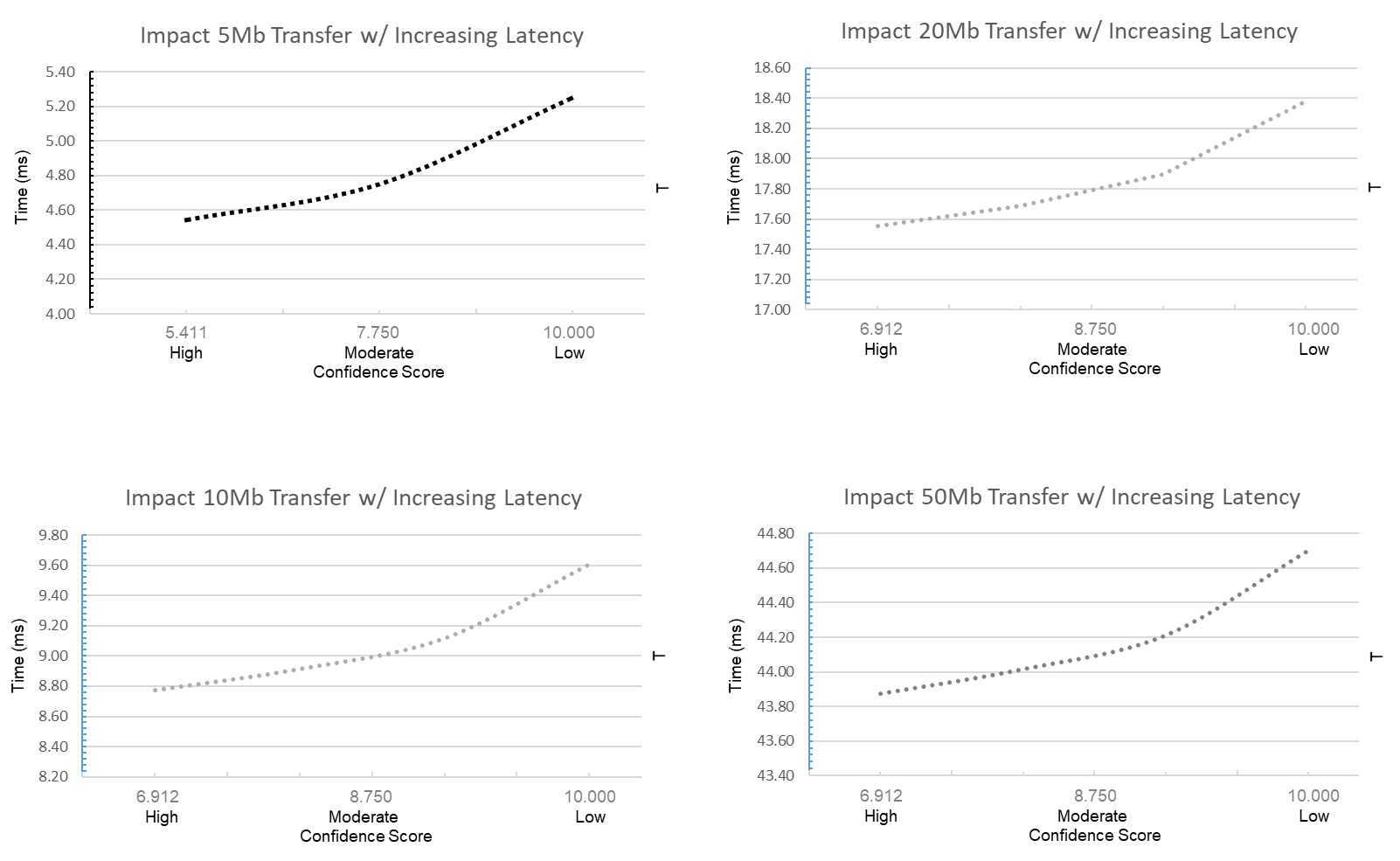
**Sample of the Survey Results**

# Appendix E – Framework Data & Results

A small sample of the data captured and analyzed by the SDN Confidence Analysis framework.



Using the data above, I calculated the flow duration for a series of varying sized data transfer. Throughout the transfers the latency was manually increased to simulate a man-in-the-middle attack.



# Appendix F – Large-Scale SDN Modeling and Simulation Setup

**Hierarchical SDN Model**

Simple Data Center:

Below is an example simple datacenter some number of racks, each with hosts and a single top-of-rack (ToR) switch. The ToR switches are connected to a central root switch. This represents a simple datacenter. This example includes the ability, from the command line, to specify how many hosts are in each rack and how many racks are in network [95].

"""

A simple datacenter topology script for Mininet.

[ s1 ]================================.

,---' | |

[ s1r1 ]=. [ s1r2 ]=. ... [ s1r# ]=.

[ h1r1 ]-| [ h1r2 ]-| ... [ h1r# ]-|

[ h2r1 ]-| [ h2r2 ]-| ... [ h2r# ]-|

... | ... | ... ... |

[ h#r1 ]-' [ h#r2 ]-' ... [ h#r# ]-'

"""

from mininet.topo import Topo

from mininet.util import irange

class DatacenterConfigurableTopo( Topo ):

"Configurable Datacenter Topology"

def build( self, numRacks=4, numHostsPerRack=4 ):

self.racks = []

rootSwitch = self.addSwitch( 's1' )

for i in irange( 1, numRacks ):

rack = self.buildRack( i, numHostsPerRack=numHostsPerRack )

self.racks.append( rack )

for switch in rack:

self.addLink( rootSwitch, switch )

def buildRack( self, loc, numHostsPerRack ):

"Build a rack of hosts with a top-of-rack switch"

dpid = ( loc \* 16 ) + 1

switch = self.addSwitch( 's1r%s' % loc, dpid='%x' % dpid )

for n in irange( 1, numHostsPerRack ):

host = self.addHost( 'h%sr%s' % ( n, loc ) )

self.addLink( switch, host )

# Return list of top-of-rack switches for this rack

return [switch]

# Allows the file to be imported using `mn --custom <filename> --topo dcconfig`

topos = {

'dcconfig': DatacenterConfigurableTopo

}

More Realistic Data Center:

In a real datacenter, there is often more than one root switch linked in a ring pattern with at least two links going to each ToR switch. This allows for a failure of one of the root switches without bringing down the entire network. To ensure full redundancy it is necessary to configure it so that every rack has two ToR switches, each connected with a single link to the root switches and provide every host with a connection to both ToR switches in that rack [95].

"""

A simple datacenter topology script for Mininet.

,----------------------------. Each root switch connected in ring.

[ s1 ]------[ s2 ]--- ... ---[ s# ]

| | | Each ToR switch connects to its

,==='==========='================' associated root switch. (s3r1 <-> s3)

|-[ s1r1 ]=. [ s1r2 ]=. ... [ s1r# ]=.

|-[ s2r1 ]=| [ s2r2 ]=| ... [ s2r# ]=|

| ... | ... | ... ... |

`-[ s#r1 ]=| [ s#r2 ]=| ... [ s#r# ]=|

| | |

[ h1r1 ]-| [ h1r2 ]-| ... [ h1r# ]-|

[ h2r1 ]-| [ h2r2 ]-| ... [ h2r# ]-|

... | ... | ... ... |

[ h#r1 ]-' [ h#r2 ]-' ... [ h#r# ]-'

"""

from mininet.topo import Topo

from mininet.util import irange

class DatacenterHAFullTopo( Topo ):

"Configurable Datacenter Topology"

def build( self, numRacks=4, numHostsPerRack=4, numHASwitches=2 ):

# This configuration only supports 15 or less root switches

if numHASwitches >= 16:

raise Exception( "Please use less than 16 HA switches" )

self.racks = []

rootSwitches = []

lastRootSwitch = None

# Create and link all the root switches

for i in irange( 1, numHASwitches ):

rootSwitch = self.addSwitch( 's%s' % i )

rootSwitches.append( rootSwitch )

# If we have initialized at least two switches, make sure to

# connect them. This handles s1 -> s2 -> ... -> sN

if lastRootSwitch:

self.addLink( lastRootSwitch, rootSwitch )

lastRootSwitch = rootSwitch

# Make the final link from the last switch to the first switch

if numHASwitches > 1:

self.addLink( lastRootSwitch, rootSwitches[0] )

for i in irange( 1, numRacks ):

rack = self.buildRack( i, numHostsPerRack=numHostsPerRack,

numHASwitches=numHASwitches )

self.racks.append( rack )

# For every HA switch, add a link between the rack switch and root

# switch of the same ID

for j in range( numHASwitches ):

self.addLink( rootSwitches[j], rack[j] )

def buildRack( self, loc, numHostsPerRack, numHASwitches ):

"Build a rack of hosts with a top-of-rack switch"

switches = []

for n in irange( 1, numHASwitches ):

# Make sure each switch gets a unique DPID based on the location

# in the rack for easy decoding when looking at logs.

dpid = ( loc \* 16 ) + n

switch = self.addSwitch( 's%sr%s' % (n, loc), dpid='%x' % dpid )

switches.append( switch )

for n in irange( 1, numHostsPerRack ):

host = self.addHost( 'h%sr%s' % ( n, loc ) )

# Add a link from every top-of-rack switch to the host

for switch in switches:

self.addLink( switch, host )

# Return list of top-of-rack switches for this rack

return switches

# Allows the file to be imported using `mn --custom <filename> --topo dafull`

topos = {

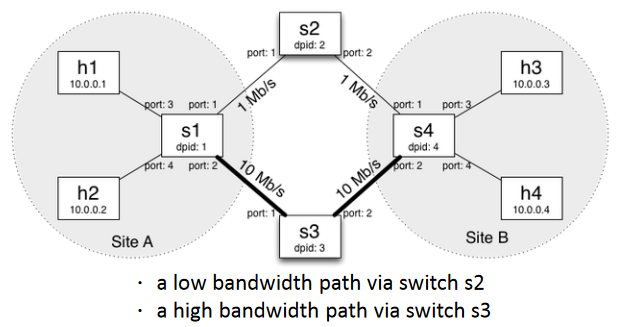
'dchafull': DatacenterHAFullTopo

}

**Multi-Tenancy SDN Model**

To gain an understanding of the basics of SDN network slicing and how to control different slices with different controller, I created the sample topology below. I will create a network application that create multiple layer-2 network slices for different portions of the flowspace [63].

The figure below illustrates a single SDN network with two different bandwidth paths (through switches s2 and s3) [96]. The script below will generate this topology:



#!/usr/bin/python

from mininet.topo import Topo

class FVTopo(Topo):

    def \_\_init\_\_(self):

        # Initialize topology

        Topo.\_\_init\_\_(self)

        # Create template host, switch, and link

        hconfig= {'inNamespace':True}

        http\_link\_config= {'bw': 1}

        video\_link\_config= {'bw': 10}

        host\_link\_config= {}

        # Create switch nodes

        for i in range(4):

            sconfig= {'dpid': "%016x" % (i+1)}

            self.addSwitch('s%d' % (i+1), \*\*sconfig)

        # Create host nodes

        for i in range(4):

            self.addHost('h%d' % (i+1), \*\*hconfig)

    #Add switch links

    self.addLink('s1', 's2', \*\*http\_link\_config)

    self.addLink('s2', 's4', \*\*http\_link\_config)

    self.addLink('s1', 's3', \*\*video\_link\_config)

    self.addLink('s3', 's4', \*\*video\_link\_config)

    # Add host links

    self.addLink('h1', 's1', \*\*host\_link\_config)

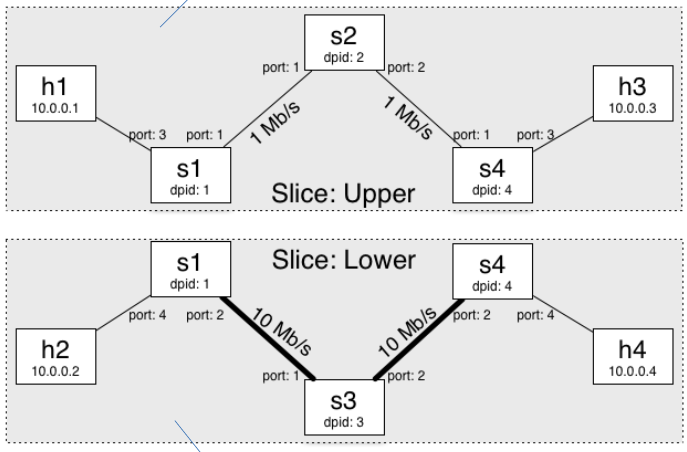
    self.addLink('h2', 's1', \*\*host\_link\_config)

    self.addLink('h3', 's4', \*\*host\_link\_config)

    self.addLink('h4', 's4', \*\*host\_link\_config)

topos= { 'fvtopo': ( lambda: FVTopo() )}

To understand how an SDN “slice” and multi-tenancy intersect, I will explain how to divide the network into two separate slices, upper and lower. To implement this isolation, there is a need to block communication between hosts in different slices. This done by inserting drop rules on certain switches.



#### Creating Network Slices [97]

Each slice will be managed by a separate controller which will control all the traffic in its slice. To create a slice named upper connecting to a controller listening on tcp:localhost:10001 by running the following command:

$ fvctl -f /dev/null add-slice upper tcp:localhost:10001 admin@upperslice

Similarly, create a slice named lower connecting to a controller listening on tcp:localhost:10002.

$ fvctl -f /dev/null add-slice lower tcp:localhost:10002 admin@lowerslice

Ensure that the slices were added by running the following command:

$ fvctl -f /dev/null list-slices

#### 

#### Creating the Flowspaces [97]

Flowspaces associate packets of a particular type in the network to specific slices. When a packet matches multiple flowspaces, FlowVisor assigns it to the flowspace with the highest priority number.

* match describes a flow or flows. Such flow descriptions comprise a series field=value assignments, separated by commas.
* slice-perm is a comma-separated list of slices that have control over a specific FlowSpace. slice-perm is of the form "slicename1=perm[slicename2=perm[...]]". Each slice can have three types of permissions over a flowspace: *DELEGATE*, *READ*, and *WRITE*. Permissions are a bitmask specified as an integer, with *DELEGATE*=1, *READ*=2, *WRITE*=4.

To create a flowspace named dpid1-port1 (with priority value 1) that maps all the traffic on port 1 of switch s1 to the upper slice by running the following command:

$ fvctl -f /dev/null add-flowspace dpid1-port1 1 1 in\_port=1 upper=7

Similarly, create a flowspace named dpid1-port3 that maps all the traffic on port 3 of switch s1 to the upper slice:

$ fvctl -f /dev/null add-flowspace dpid1-port3 1 1 in\_port=3 upper=7

Now create a flowspace for all the traffic at a switch by using the match value of any. Use that technique to add switch s2 to the upper slice:

$ fvctl -f /dev/null add-flowspace dpid2 2 1 any upper=7

Next, create flowspaces to add ports 1 and 3 of switch s4 to the upper slice:

$ fvctl -f /dev/null add-flowspace dpid4-port1 4 1 in\_port=1 upper=7

$ fvctl -f /dev/null add-flowspace dpid4-port3 4 1 in\_port=3 upper=7

Ensure that the flowspaces are correctly added:

$ fvctl -f /dev/null list-flowspace

Then repeat to create flowspaces for the lower slice:

$ fvctl -f /dev/null add-flowspace dpid1-port2 1 1 in\_port=2 lower=7

$ fvctl -f /dev/null add-flowspace dpid1-port4 1 1 in\_port=4 lower=7

$ fvctl -f /dev/null add-flowspace dpid3 3 1 any lower=7

$ fvctl -f /dev/null add-flowspace dpid4-port2 4 1 in\_port=2 lower=7

$ fvctl -f /dev/null add-flowspace dpid4-port4 4 1 in\_port=4 lower=7

Ensure that the flowspaces are correctly added:

$ fvctl -f /dev/null list-flowspace

Start a controller instance for each slice. The controllers will reactively install routes based on the destination MAC address, and it is provided as an executable. Open two fresh terminal tabs and run the following:

* Terminal 1

$ ./pox.py openflow.of\_01 –port=10001 flowvisor\_lab1\_upper

* Terminal 2

$ ./pox.py openflow.of\_01 –port=10002 flowvisor\_lab1\_upper

This will launch two instances of the controller, listening on port 10001 and 10002 respectively.

Verify that h1 can ping h3 but not h2 and h4 (and vice versa). In the Mininet console run the following commands:

mininet> h1 ping -c1 h3

mininet> h1 ping -c1 -W1 h2

mininet> h1 ping -c1 -W1 h4

Verify that h2 can ping h4 but not h1 and h3 (and vice versa). In the mininet console run the following commands:

mininet> h2 ping -c1 h4

mininet> h2 ping -c1 -W1 h1

mininet> h2 ping -c1 -W1 h3

**SDN Emulation**

Step 1: Connect guest OS (i.e. Mininet) to the internet.

In the VirtualBox network setting, make sure the NAT interface is enabled to allow connection to the internet. The ip address should look like this : 10.0.3.15 (a class A address). Test connectivity by pinging an internet (www.google.com) to make sure the connection to internet from the guest OS (i.e. Mininet) is functioning [65].  
[](https://1.bp.blogspot.com/-Pe5bZ1uz25Y/VIYTVI6m81I/AAAAAAAAJmk/4sT7ch665ho/s1600/mininet+config.JPG)

Step 2: Start the network  
Start a Mininet. At a minimum create a network with a switch and two hosts:

sudo mn  --switch ovsk --mac --topo single,2

Step 3: Connect the guest interface to the OpenvSwitch bridge  
Use the ovs-vsctl command to configure openvswitchd(this is a process of OpenvSwitch).  
Do this by opening a terminal window for switch 1, s1, as this command does not run directly on mininet. Check the OpenvSwitch configuration using the command: ovs-vsctl show  
Run the following command to connect eth1 to s1: ovs-vsctl add-port s1 eth1  
Check the configuration again using ovs-vsctl show. The new interface that is added has been highlighted in red.  
root@mininet-vm:~# ovs-vsctl show  
d27a9060-3edf-4ee7-a4cf-09e705c93f56  
    Bridge "s1"  
        Controller "ptcp:6634"  
        Controller "tcp:127.0.0.1:6633"  
            is\_connected: true  
        fail\_mode: secure  
        Port "eth1"  
            Interface "eth1"  
        Port "s1-eth1"  
            Interface "s1-eth1"  
        Port "s1-eth2"  
            Interface "s1-eth2"  
        Port "s1"  
            Interface "s1"  
                type: internal  
    ovs\_version: "2.0.1"  
  
Step 4: Run dhclient on hosts.  
Open terminal windows for host 1, h1, and host 2, h2, and run the following commands to remove the ip from h1-eth0 and the second command gets the ip address for h1-eth0 from dhcp server.   
root@mininet-vm:~# ifconfig h1-eth0 0  
root@mininet-vm:~# dhclient h1-eth0  
root@mininet-vm:~# ifconfig  
h1-eth0   Link encap:Ethernet  HWaddr 00:00:00:00:00:01   
          inet addr:10.0.3.16  Bcast:10.0.3.255  Mask:255.255.255.0  
          inet6 addr: fe80::200:ff:fe00:1/64 Scope:Link  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:3304 (3.3 KB)  TX bytes:1764 (1.7 KB)  
  
lo        Link encap:Local Loopback   
          inet addr:127.0.0.1  Mask:255.0.0.0  
          inet6 addr: ::1/128 Scope:Host  
          UP LOOPBACK RUNNING  MTU:65536  Metric:1  
          RX packets:1252 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:1252 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:0  
          RX bytes:151432 (151.4 KB)  TX bytes:151432 (151.4 KB)  
  
Next check the internet connectivity using ping.  
root@mininet-vm:~# ping www.google.com  
PING www.google.com (216.58.216.164) 56(84) bytes of data.  
64 bytes from sea15s02-in-f4.1e100.net (216.58.216.164): icmp\_seq=14 ttl=54 time=61.9 ms  
64 bytes from sea15s02-in-f4.1e100.net (216.58.216.164): icmp\_seq=15 ttl=54 time=60.7 ms  
^C  
--- www.google.com ping statistics ---  
15 packets transmitted, 2 received, 86% packet loss, time 14065ms  
rtt min/avg/max/mdev = 60.707/61.336/61.965/0.629 ms

# Appendix G – University Small SDN Testbed

Tutorial on VMware Esxi SDN Testbed Setup

1. Within esxi, set IP info (static IP, mask, gateway, DNS, and host name).
2. Install vSphere Client on local PC
3. In local PC, modify “Drivers/etc/hosts” to include IP and name for esxi and vcenter
4. Launch vSphere Client
5. In vSphere Client, Deploy VSphere Appliance
   1. File – Deploy OVF Template, Thin Provision
6. In vSphere Client, do basic configurations
   1. Set Time: Configuration – Time Configuration – Properties – Add NTP Server “Start and Stop with Host”
   2. Enable SSH: Security Profile – Properties – SSH “Start and Stop with Host”
   3. Prevent SSH Error: Advanced Settings – UserVars – change Boolean for ShellAccess
7. In vSphere Client, setup vCenter
   1. Edit Settings: Set RAM to 4GB
   2. Console View and Start VM, let is boot to bluescreen
8. On local PC (connected to esxi server router), Log on to provide IP and configure: modify time, “admin” for password, and network – address for static ip
9. On local PC, surf to sdnvcenter:xxxx
   1. Verify Time still set
10. WinSCP into vCenter to modify host file to remove host name from first line and add esxi and vcenter to bottom of file.
11. On local PC on sdnvcenter site, launch “setup wizard” with default settings.
12. On local PC on sdnvcenter site, reboot server (under system tab) & open in new console window
13. On local PC in vSphere Client, select esxi and VM Startup & Shutdown, modify to:
    1. Allow vm start & stop automatically
    2. Startup in 15 seconds and continue with VM tools
    3. Shutdown in 15 seconds and “guest shutdown”
    4. Move vCenter App into “Auto Startup” group
14. On local PC open new vSphere Client, log in to vCenter to verify connectivity.
15. On local PC goto https://vcenter and select “log in to vSphere Web Client”
16. On local PC, surf to sdnvcenter:xxxx
    1. Verify Time still set
17. On vSphere Web Client, select vCenter Servers – vcenter and then “create datacenter”
    1. Select “add a host” and type esxi, then next to finish
18. Optional ~ adding more storage (i.e. a data store) in Datacenters – Datacenter, select “add a datastore”
    1. Create “vmfs datastore”, select host “esxi” and select storage and give it a name
    2. Select vmfs 5 (allows for >2TB drives), select next to finish
19. Setup ISO Storage: Click add folder for ISO storage, then download and install “VMware Client Integration Plug-In” Note: You might need to enable plug-ins.
20. Uploading ISOs and existing VMs

# Appendix H – Low-Cost SDN Router Setup

A Low-Cost SDN Router can be created using various older Linksys or D-Link WiFi routers and OpenWRT. Below I will describe the steps to download, format, and deploy such a router. The following assumes the user is working in a Linux environment [98][99]**.**

**1. Install the dependencies.**

* sudo apt-get install build-essential binutils flex bison autoconf gettext texinfo sharutils \ subversion libncurses5-dev ncurses-term zlib1g-dev gawk

**2. Download OpenWRT source and feeds**

* mkdir ~/openwrt
* cd ~/openwrt
* sudo apt-get update
* svn co svn://svn.openwrt.org/openwrt/branches/backfire
* cd backfire
* ./scripts/feeds update -a
* ./scripts/feeds install -a

**3. Get OpenWRT config file**

* make menuconfig

**4. Configure OpenWRT with dependencies and build**

* make prereq
* make

**5. Set up OpenFlow and select package.**

* sudo apt-get install git
* git clone https://github.com/CPqD/openflow-openwrt.git
* cd ~/openwrt/backfire/packages/
* ln -s ~/openwrt/openflow-openwrt/openflow-1.3/
* cd ~/ofwrt/backfire/
* ln -s ~/openwrt/openflow-openwrt/openflow-1.3/files
* cd ~/ofwrt/
* make menuconfig
* make kernel\_menuconfig
* make

# Appendix I – SDN OTG Parts List

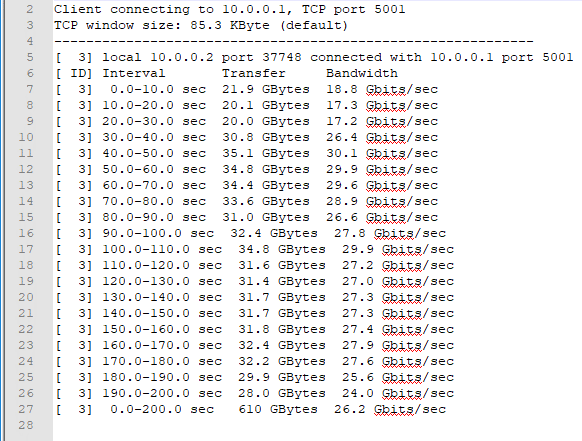
|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Quantity** | **Cost/Per** | **Sub-Total** |
| Zodiac FX 100Mb 4-Port SDN Switch | 4 | 63.75 | 255.00 |
| Raspberry Pi 3 Complete Starter Kit by Vilros | 4 | 69.99 | 279.96 |
| Ethernet Cables (multi-color), 3ft length, 8-pack | 1 | 14.98 | 14.98 |
| Ethernet Cables (orange), 3ft length, 4-pack | 1 | 15.95 | 15.95 |
| Belkin 12-port Power Strip (Model: BP112230-08) | 1 | 27.49 | 27.49 |
| Tek Republic USB Sharing Switch - 4 Port Manual Switch (Model: TUS-400) | 1 | 16.99 | 16.99 |
| USB Hub, 4-port by Dlink (Model: DUB-H4) | 1 | 19.99 | 19.99 |
| USB LAN Ethernet Adapter, 10/100Mb by Manhattan (Model: 506731) | 4 | 16.99 | 67.96 |
| A IO Gear HDMI Video KVM (Model: GHDSW4K4) | 1 | 21.89 | 21.89 |
| 3-foot HMDI Cables | 4 | 5.09 | 20.36 |
| TP-LINK WiFi Router (Model: TL-WR1043ND) | 1 | 49.99 | 49.99 |
| InFocus Kangaroo Plus | 1 | 169.99 | 169.99 |
|  |  | **Total** | **$960.55** |

# Appendix J – Application Data & Results

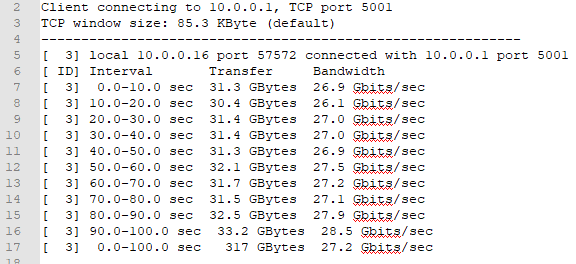
**General Application Testing and Experimentation**

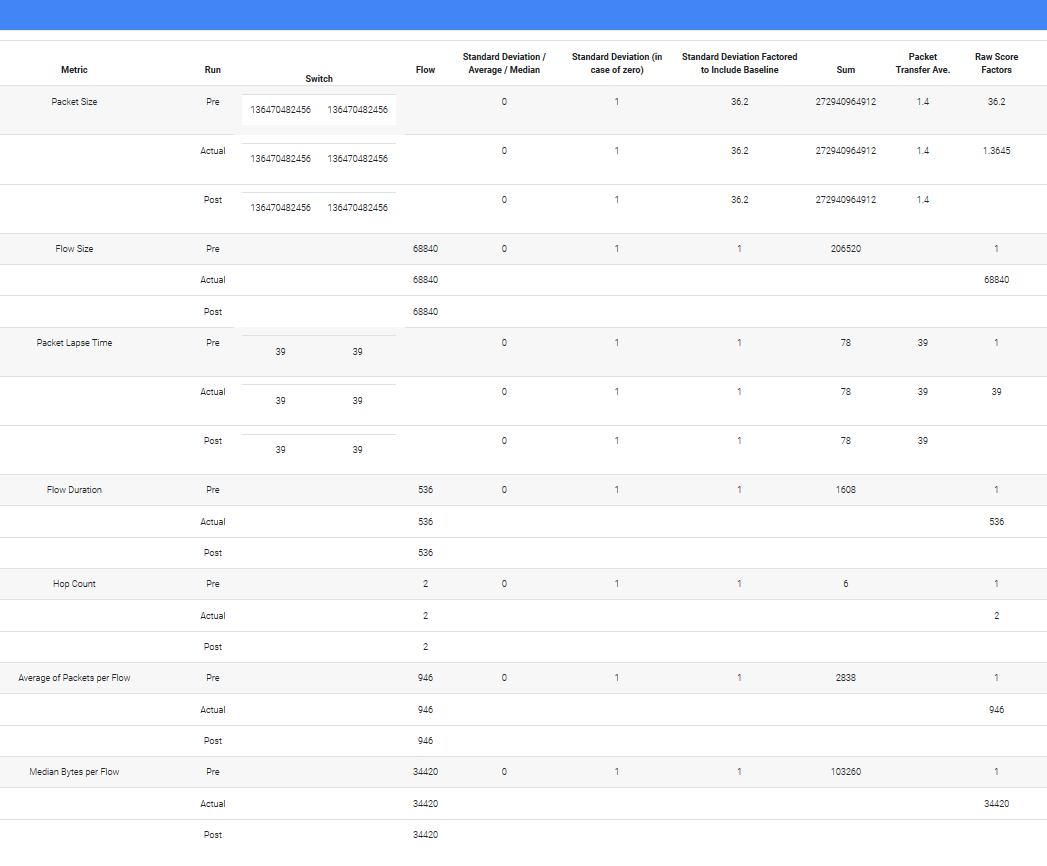
Below is a sampling of the data collected during the testing and experimentation phases of Chapter 8.

**- Baseline Simulation**

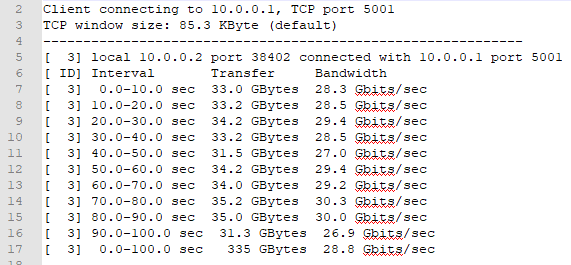


**- Data Center Simulation**: In this sample you can see a comparison of the traffic generation and the assessment tool at 40.0 seconds into the run. Total throughput at this point according to the traffic generator is 134.8GBytes and according the tool is 136GBytes.





**- OTG Physical Testbed**: In this sample you can see a comparison of the traffic generation and the assessment tool at 70.0 seconds into the run. Total throughput at this point according to the traffic generator is 233.5GBytes and according the tool is 227.1GBytes.



**Man in the Middle Testing and Experimentation**

Below is a sampling of the data collected during the testing and experimentation phases of Chapter 8.

**- Data Center Simulation**: In the samples below, I broke out each of the scoring groups (base, environmental, and overall confidence). It is easier to see the linear relationship and the cross over points of the data.

# Appendix K – Acronym List

| Abbreviation | Term |
| --- | --- |
| API | Application Programming Interface |
| AUP | Acceptable Use Policy |
| CRC | Cyclic Redundancy Check |
| CSU | Colorado State University |
| CVSS | Common Vulnerability Scoring System |
| DDoS | Distributed Denial of Service |
| DMZ | Demilitarized Zone |
| DNS | Domain Name System |
| DoS | Denial of Service |
| DSC | IEEE Conference on Dependable and Secure Computing |
| DSN | International Conference on Dependable Systems and Networks |
| GUI | Graphical User Interface |
| HTTP(S) | Hypertext Transfer Protocol (Secure) |
| IANA | Internet Assigned Numbers Authority |
| ICCCN | International Conference on Computer Communications and Networks |
| ICMP | Internet Control Message Protocol |
| IDS | Intrusion Detection System |
| IOT | Internet of Things |
| IP | Internet Protocol |
| ISP | Internet Service Provider |
| IT | Information Technology |
| KVM | Kernel-based Virtual Machine |
| LAN | Local Area Network |
| M&S | Modeling and Simulation |
| MAC | Media Access Control |
| MitM | Man in the Middle |
| MTU | Maximum Transmission Unit |
| NAT | Network Address Translation |
| NTP | Network Time Protocol |
| ODL | OpenDaylight |
| ONF | Open Networking Foundation |
| OOB | Out-of-Band |
| OS | Operating System |
| OTG | On-The-Go |
| PVM | Path Verification Mechanism |
| QoS | Quality of Service |
| RTT | Round Trip Time |
| SDN | Software-Defined Network |
| SMV | Spoofability, Measurability, Variability |
| SSL | Secure Sockets Layer |
| TCP | Transmission Control Protocol |
| TLS | Transport Layer Security |
| TOS | Type of Service |
| TSDR | Time Series Data Repository |
| UCCS | University of Colorado, Colorado Springs |
| UDP | User Datagram Protocol |
| UI | User Interface |
| URL | Uniform Resource Locator |
| VLAN | Virtual Local Area Network |
| VPN | Virtual Private Network |
| WAN | Wide Area Network |