Link Failover for Resilient Cyber-Physical Systems

In-Hee Park and Kyung-Joon Park
Department of Information and Communication Engineering,
Daegu Gyeongbuk Institute of Science & Technology (DGIST), Daegu, Republic of Korea
Email: {inhee, kjp}@dgist.ac.kr

***Abstract*—In this work, we propose a software-defined networking approach to recover from network failures for resilient cyber-physical systems (CPS). Based on the software-defined networking technology, the proposed scheme finds alternative paths when the stability of the physical system is threatened due to link disconnection. We carry out experiments to measure the network recovery time from physical link disconnection. Our results show that the SDN approach is promising for providing the stability of the physical system, which is critical to the resilience of CPS.**

Keywords—CPS; SDN; link failover; real-time, resilience

I. Introduction

Recently, cyber-physical systems (CPS) have emerged as an enabler for the 4th industrial revolution [1, 2]. CPS focus on the interaction between the physical and the cyber systems, in which various form of information is exchanged through communication networks. In addition, while the physical system evolves in real-time, the cyber system operates in accordance with the logic flow according to the computation results, independent of the time flow. Therefore, if the information between the cyber and the physical systems is not properly transmitted in real time due to network failures, the physical system becomes unsecured and the entire system fails to operate in a proper manner.

In order to guarantee the reconstruction and restoration of the network topology within the time constraint for resilient CPS, it is necessary to monitor each network link to detect failures. Therefore, in this work, we design link failover experiments to secure the stability of the physical system by implementing a testbed environment using software-defined networking [3].

II. Link Failover in CPS

A. Testbed Configuration Environment

In this section, we explain our testbed designed for the link failover experiment of CPS. The testbed consists of 4 switches, 2 host PCs and an SDN controller as shown in Fig. 1. The underlying idea for building this test environment is to find out how much recovery time is required after network link failures, where messages are exchanged between the PCs, which plays the role of physical systems.

In addition, in order to take account of a mixed environment of wired and wireless networks, we consider that one switch operates as a wireless AP, one host PC is wirelessly connected, and the other host PC is connected by wire, respectively.

As for the equipment used to build the testbed, OpenvSwitch installed in raspberry pi 3 to support OpenFlow 1.3 version is used for 4 switches. The ONOS SDN controller is used as the network controller [4, 5]. The host PCs adopt the Window 10 environment and in the Linux environment.

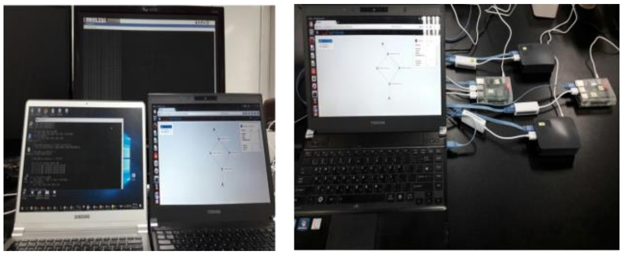


Fig. 1. Testbed environment.

B. Link Failover Experimental Results

In this section, we describe the scenario and experimental results based on the testbed.

The experiment scenario is as follows: One of the two host PCs connected to the testbed periodically transmits an ICMP packet with a period of 1ms, and the other one monitors the packet. During this transmission, if the physical link disconnection occurs between S3 and S4 in the original path S1-S3-S4 as shown in Fig. 2, the link failure is detected by the SDN controller and the traffic is rerouted to the alternative path of S1-S2-S4.

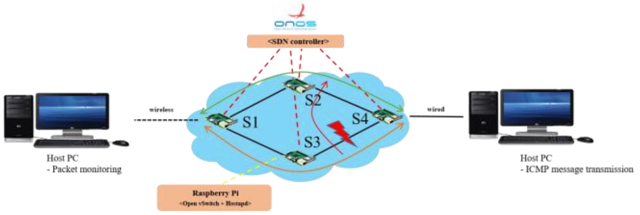


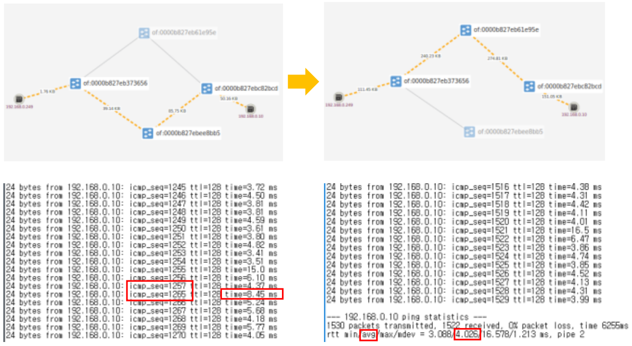
Fig. 2. Link failover scenario.

978-1-5386-1898-1/17/$31.00 ©2017 IEEE

The recovery time is measured by monitoring the number of packets lost in link disconnection, round trip time (RTT) of the packet immediately after recovery, and average RTT. More specifically, the recovery time is calculated by using the following equation:

$$\text{Recovery Time} = \text{Packet Cycle} * \text{Number of Packet Losses} + \text{RTT} - \text{Average RTT}$$

The experimental results are shown in Fig. 3. When the link disconnection occurs, the switch sends port state change information to the SDN controller, and the flow information changed in the SDN controller is transmitted to switches based on this information. So it is possible to move from the original path to the alternative path to provide resilient real-time communication.



```
24 bytes from 192.168.0.10: icmp_seq=1245 ttl=128 time=3.72 ms
24 bytes from 192.168.0.10: icmp_seq=1246 ttl=128 time=4.50 ms
24 bytes from 192.168.0.10: icmp_seq=1247 ttl=128 time=3.81 ms
24 bytes from 192.168.0.10: icmp_seq=1248 ttl=128 time=3.81 ms
24 bytes from 192.168.0.10: icmp_seq=1249 ttl=128 time=4.59 ms
24 bytes from 192.168.0.10: icmp_seq=1250 ttl=128 time=3.61 ms
24 bytes from 192.168.0.10: icmp_seq=1251 ttl=128 time=3.80 ms
24 bytes from 192.168.0.10: icmp_seq=1252 ttl=128 time=4.82 ms
24 bytes from 192.168.0.10: icmp_seq=1253 ttl=128 time=3.41 ms
24 bytes from 192.168.0.10: icmp_seq=1254 ttl=128 time=3.51 ms
24 bytes from 192.168.0.10: icmp_seq=1255 ttl=128 time=15.0 ms
24 bytes from 192.168.0.10: icmp_seq=1256 ttl=128 time=6.10 ms
24 bytes from 192.168.0.10: icmp_seq=1257 ttl=128 time=4.37 ms
24 bytes from 192.168.0.10: icmp_seq=1265 ttl=128 time=8.45 ms
24 bytes from 192.168.0.10: icmp_seq=1266 ttl=128 time=5.24 ms
24 bytes from 192.168.0.10: icmp_seq=1267 ttl=128 time=5.68 ms
24 bytes from 192.168.0.10: icmp_seq=1268 ttl=128 time=4.18 ms
24 bytes from 192.168.0.10: icmp_seq=1269 ttl=128 time=5.77 ms
24 bytes from 192.168.0.10: icmp_seq=1270 ttl=128 time=4.05 ms
```

```
24 bytes from 192.168.0.10: icmp_seq=1516 ttl=128 time=4.38 ms
24 bytes from 192.168.0.10: icmp_seq=1517 ttl=128 time=4.31 ms
24 bytes from 192.168.0.10: icmp_seq=1518 ttl=128 time=4.42 ms
24 bytes from 192.168.0.10: icmp_seq=1519 ttl=128 time=4.11 ms
24 bytes from 192.168.0.10: icmp_seq=1520 ttl=128 time=4.01 ms
24 bytes from 192.168.0.10: icmp_seq=1521 ttl=128 time=16.5 ms
24 bytes from 192.168.0.10: icmp_seq=1522 ttl=128 time=6.47 ms
24 bytes from 192.168.0.10: icmp_seq=1523 ttl=128 time=3.86 ms
24 bytes from 192.168.0.10: icmp_seq=1524 ttl=128 time=4.74 ms
24 bytes from 192.168.0.10: icmp_seq=1525 ttl=128 time=3.85 ms
24 bytes from 192.168.0.10: icmp_seq=1526 ttl=128 time=4.52 ms
24 bytes from 192.168.0.10: icmp_seq=1527 ttl=128 time=4.13 ms
24 bytes from 192.168.0.10: icmp_seq=1528 ttl=128 time=4.31 ms
24 bytes from 192.168.0.10: icmp_seq=1529 ttl=128 time=3.99 ms

--- 192.168.0.10 ping statistics ---
1530 packets transmitted, 1522 received, 0% packet loss, time 6255ms
rtt min/avg/max/mdev = 3.088/4.026/16.578/1.213 ms, pipe 2
```

Fig. 3. Experimental results of link failover.

To obtain the average recovery time, the experiment described above is repeated 15 times. The recovery time of each ICMP packet is calculated by using the recovery time formula. Table I summarizes the experimental results.

The results show that the link is restored within the average recovery time of 30.20 ms, which is sufficient to deliver control information within the typical time constraint of the physical systems for resilient CPS.

TABLE I. RECOVERY TIME.

Number	Packet loss	RTT	Avg. RTT	Recovery time
1	6	7.89ms	4.427ms	9.462ms
2	7	8.31ms	4.039ms	11.271ms
3	4	7.91ms	3.912ms	7.998ms
4	4	7.86ms	3.847ms	8.013ms
5	5	8.44ms	4.192ms	9.248ms
6	5	8.51ms	4.080ms	9.42ms
7	309	8.43ms	5.681ms	311.749ms
8	5	8.00ms	4.185ms	8.815ms
9	7	8.45ms	4.026ms	11.424ms
10	7	6.17ms	4.237ms	8.933ms
11	5	10.5ms	4.462ms	14.038ms
12	6	9.66ms	4.154ms	11.506ms
13	5	9.63ms	4.640ms	9.99ms
14	5	9.00ms	4.167ms	9.836ms
15	7	8.75ms	4.466ms	11.284ms
Avg.	25.8	8.501ms	4.301ms	30.20ms

However, as shown in Table Ⅰ, there exists a case when substantial packets are lost. This problem is caused by errors in the communication between the switch equipment and the SDN controller. In order to resolve this issue, it is necessary to further improve the conventional SDN function for detection and rerouting of failed links. We are currently working in the direction.

III. CONCLUTIONS

In this work, we carry out research on link failover for providing resilience in CPS. Our experimental results show that the proposed SDN approach is promising for the stability of the physical system. In addition, we will conduct experiments in real physical system by applying a testbed to a train system. And we will research various network environments using testbed.

ACKNOWLEDGMENT

This work was partly supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2016R1C1B2007899) and Institute for Information and communication Technology Promotion (IITP) grant funded by the Korea government (MSIP; 2014-0-00065, Resilient Cyber-Physical Systems Research).

REFERENCES

[1] K.-J.Park, R. Zheng, and X. Liu, “Cyber-physical systems: Milestones and research challenge,” Computer Communicatnio, vol.36, issue 1, pp. 1-7, Dec. 2012.

[2] K.-J.Park, J. Kim, H. Lim, and Y. Eun, “Robust path diversity for network quality of service in cyber-physical systems,” IEEE Transaction on Industrial Informatics, vol.10, no. 4, pp. 2204-2215, November. 2014.

[3] S. Sezer et al., “Are we ready for SDN? Implementation challenges for software-defined networks, ” IEEE Communications Magazine, vol. 51, issue 7, pp.36-42, July 2013.

[4] ONOS website, https://wiki.onosproject.org.

[5] OpenvSwitch website, openvswitch.org.