A Two-Layer Network Orchestrator Offering Trustworthy Connectivity to a ROS-Industrial Application

Behzad Mirkhanzadeh, Chencheng Shao, Ali Shakeri, Takehiro Sato*, Miguel Razo-Razo, Marco Tacca, Andrea Fumagalli, and Naoaki Yamanaka*

OpNeAR Lab, Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas, TX, USA
*Department of Information and Computer Science, Keio University, Japan
e-mail: {behzad, chencheng.shao, alishakeri, mrazo, mtacca, andreaf}@utdallas.edu, takehiro.sato@yamanaka.ics.keio.ac.jp, yamanaka@ics.keio.ac.jp

ABSTRACT
This paper describes an experiment carried out to demonstrate robustness and trustworthiness of an orchestrated two-layer network test-bed (PROnet). A Robotic Operating System Industrial (ROS-I) distributed application makes use of end-to-end flow services offered by PROnet. The PROnet Orchestrator is used to provision reliable end-to-end Ethernet flows to support the ROS-I application required data exchange. For maximum reliability, the Orchestrator provisions network resource redundancy at both layers, i.e., Ethernet and optical. Experimental results show that the robotic application is not interrupted by a fiber outage.

Keywords: network orchestrator, trustworthy network, ROS, ROS-I, industrial robotics.

1. INTRODUCTION
Robots have been widely used in industry to reduce the risk of severe injuries to human operators. Robots can operate using either local or remote (in the Cloud) computing resources. The latter case has several advantages over the former, including resource sharing, cost effectiveness, ease of maintenance, and scalability. In a typical scenario, data sensed by or for the robot need to be quickly and reliably transferred to one or more remote servers for further processing. Particularly important to the application is to rely on a reliable connection between the robot/sensor and the remote server, in order to guarantee uninterrupted data transmission even in the presence of network outages. A robust and trustworthy network infrastructure is therefore needed to fulfill this mission-critical requirement.

In this study, we demonstrate the trustworthiness of PROnet, a reliable two-layer network test-bed with a Software Defined Networking (SDN) orchestrator [1]. PROnet can be programmed to provision two (primary and secondary) disjoint Ethernet flows (paths) of data transmission between the industrial robot/sensor and the remote server. The secondary path is in standby, ready to provide an alternative robot-server connection in the presence of a network outage affecting the primary path. The experimental results of this paper demonstrate that the self-healing and prompt response of the Ethernet layer prevent a ROS-I application from being disrupted. A restoration lightpath is also concurrently provisioned at the optical layer to quickly prepare the network against the possibility of a second fiber outage. The experiments also reveal that, subsequent to the fiber outage, the TCP protocol slowly recovers to its maximum throughput when the round trip time of the secondary path is significantly larger than the primary’s.

2. INDUSTRIAL ROBOTIC APPLICATION GODEL
The chosen industrial robot application performs “metal surface blending.” An industrial robot is attached to two end effectors: a 3D camera and a blending tool. The chosen robot (e.g., Kinect) is responsible for scanning the environment looking for metal surfaces, and the blending tool (e.g., ATI sander) is used to blend the surface selected by the operator through a graphical user interface (GUI). The (Godel) software that carries out this task works as follows [2]. Metal parts that need to be blended are laid out at arbitrary locations and with arbitrary orientations on the robot’s work bench. The Kinect (an RGB-depth camera) installed on the robot arm is automatically positioned by the control software to scan the objects on the work bench. The control software is configured to move the robot’s arm (consequently the Kinect) to specific multiple positions above the work bench to scan all the parts. The Kinect-collected data is transmitted to the application software (residing at the server) for further processing. Each Kinect image is about 60 MB. After successfully completing this data transmission, the control software uses a segmentation algorithm to extract and detect the boundaries of metal parts’ surfaces. A GUI displays such boundaries to a human operator, who can select which part needs to be blended next. A path planner ROS-I software module (node) makes use of the Descartes planner (boost graph solver) [2] to compute a physical path for the blending tool. The path planner node is responsible for finding the valid and optimal solutions as specified by the robot model. These solutions are computed through a sequence of trajectory points that avoid collisions while accounting for joint limits and singularities.
2.1 Godel Software Architecture
This Section describes the key ROS nodes of the Godel application. Godel consists of multiple ROS nodes, which can be distributed across and executed at multiple servers. These nodes are now described.

Kinect Publisher: This node publishes the Kinect data with different resolutions. Each resolution is published using a different topic. In the Godel application the only subscriber for this node is the Surface Detection node.

MoveIt: This node is a ROS package used for collision aware path planning [4]. This package is used to move the robot arm to reach specified points. It is initialized with the robot’s physical parameters, such as the robot arm size, joint type and other physical limitations.

RViz: This node is a ROS visualizer package (GUI), used for visualizing on a monitor a digitally reconstructed image of the robot and other parts. This package visualizes the 3D graphical image of the robot and is used to monitor the robot state and the sensor values.

Surface Detection: This node is the central module of the application. The Surface Detection node accepts service calls and sends control messages to other ROS nodes. It uses the MoveIt node to move the robot arm to specific locations as requested by the human operator thought the RViz GUI. The Kinect sensor (attached to the arm) captures multiple images and the Kinect Publisher node publishes them as topics. The Surface Detection node uses these published images to compute the blending path.

Path Planner: This ROS node receives as input the path computed by the Surface Detection node. It then computes the trajectories for the robot arm making sure that the blending tool moves along the previously computed path.

Industrial Robot Simulator: It is always safe to test-simulate the computed paths before executing them with the actual industrial robot. This node simulates what the robot would do upon receiving the move-instructions as they are prepared by the application software. The operator can visually verify that the outcome is consistent with expectation.

2.2 ROS Topic Transport
ROS makes use of a data-centric model for unidirectional steaming communication, in which data exchange between nodes takes the form of topics. Topics are named buses over which nodes exchange data. Nodes that generate data publish to the relevant topic. Nodes that receive data subscribe to the relevant topic. Before publishing to a topic for the first time, a node advertises its intent to publish to that topic by informing the Master node using XMLRPC. The provided information includes the message type, the topic name, and the publishing node’s URI. The Master node maintains this information in the publisher table. When a node subscribes to that topic, it sends the topic information to the Master, via XMLRPC. The Master node updates the subscriber table accordingly. In return, the subscriber is given the current list of publishers URIs for that topic. The subscribing node is then ready to initiate transport-specific connections [5]. In essence, the Master node provides a name service.

The diagram in Fig. 1 shows the procedure that is used in Godel to create a TCP connection between the Kinect Publisher and the Surface Detection node. The Kinect Publisher node receives point cloud data from the Kinect sensor and publishes point cloud messages to a topic called /sensor_point_cloud. All that the Kinect Publisher node does is to publish point cloud messages, without knowledge of whether any node is subscribed to that topic. The Surface Detection node subscribes to the /sensor_point_cloud topic. After subscription, the Surface Detection node begins receiving point cloud messages. All that the Surface Detection does is to subscribe to the /sensor_point_cloud topic, without knowledge of whether any node is publishing to that topic. The two nodes can be started, terminated, and restarted in any order, without inducing any error conditions. The Surface Detection and Kinect Publisher nodes identify each other (before starting a TCP data transfer) through the name service provided by the Master node. The Master node has a XMLRPC URI that is known to all ROS-I nodes.

2.3 Application Experimental Setup
Figure 2 shows the Godel ROS nodes distribution that is used in our experiment. The Kinect Publisher node runs on Host 1 and the rest of the application ROS-I nodes run on Host 2. Connectivity between the two hosts is provided by the trustworthy network depicted in Fig. 3 and described in Section 3.1.
As described previously, the Surface Detection node must make use of the captured 3D images that are published by the Kinect Publisher node. These images must be transferred from Host 1 to Host 2 over a reliable and high-speed connection using the trustworthy network test-bed briefly described next. A more detailed description of the network test-bed is available in [1].

3. NETWORK INFRASTRUCTURE
A software defined networking (SDN) is applied to obtain a robust and trustworthy network that provides the connectivity required by the Godel application. In particular, the PROnet test-bed and the SDN orchestrator presented in [1] are leveraged to provision trustworthy connectivity between ROS nodes. The PRONet Orchestrator communicates with both Ethernet and optical network controllers through REST APIs. REST is offered by the SDN controller to discover the optical network topology. Through the orchestrator frontend, end-to-end connections between the two hosts can be readily provisioned.

3.1 PROnet Network Test-bed
The drawing in Fig. 3 depicts the two-layer network test-bed (PROnet) that was used to run the Godel application experiments. In this figure, an optical network (grey boxes represent optical ROADMS) is connected to two distinct Ethernet sub-networks (blue boxes represent Ethernet switches). The networks are jointly managed by the PROnet Orchestrator, which runs on an external server. As already mentioned, the application ROS-I nodes run on both Host 1 and Host 2. Two SRLG disjoint Ethernet (end-to-end) paths are computed and provisioned by the PROnet Orchestrator to provide reliable connectivity between the two hosts. The primary path is shown in green and the secondary in red. Note that the provisioning procedure requires that (1) two distinct and disjoint lightpaths be established across the optical network, and (2) two Ethernet flows be established end-to-end between the two hosts and over the two lightpaths. In addition, the PROnet Orchestrator computes and provisions Ethernet detour flow rules (some shown in yellow) connecting Ethernet switches of the primary path to one Ethernet switch of the secondary path [1]. The detour path is automatically activated by the switch on the primary path upon detection of a link disruption (label 1 in the figure indicates the case of a fiber → lightpath → link cascade outage). The swift detour flow rule activation enables the TCP data transfer to continue over part of the secondary path, and consequently the application is not interrupted. As soon as the grey lightpath is operational, the green path is reconstructed and the detour flow rule is deactivated. The network is now prepared to overcome a second fiber failure.

4. TCP PERFORMANCE ANALYSIS
To test the network trustworthiness, one of the fiber connectors was unplugged to disrupt the primary path (green) while TCP is transferring the second of three Kinect-generated images (about 60 MB each) from Host 1 to Host 2. The round trip time (RTT) of the primary path is set to 20 ms, while the secondary has a 60 ms RTT. CUBIC TCP is applied, which makes use of an optimized congestion control algorithm for high bandwidth networks with high latency. Figure 4a shows the TCP data transfer measured using Wireshark. The first 3D image is entirely transmitted over the primary path, with a transfer rate that quickly reaches close to 1 Gbps (i.e., the network interface card rate at the hosts). For the second image data transfer, a zoomed-in chart is provided. During the initial part of the TCP transfer, the data rate quickly reaches close to 1 Gbps. The data transfer momentarily goes down to zero as soon as the fiber connector is unplugged, slowly resuming about 300 ms later.
when data is transmitted over the secondary path with 60 ms RTT. The TCP average transfer rate does not have
time to reach 1 Gbps before completion of the image transfer. The third image is entirely transmitted over the
secondary path with a transfer rate that quickly reaches close to 1 Gbps. One of the possible causes for the
relatively low rate data transfer experienced immediately after the outage (but not when the third image is
transmitted over the secondary path) could be the size of the TCP receiver advertised window (RWND), which
is chosen by the protocol at the beginning of the data transfer. When RTT = 20 ms, RWDN is set to approximately
10 MB. However, when RTT = 60 ms, RWDN is set to approximately 30 MB. The fiber outage changes the
RTT from 20 to 60 ms in the middle of the second image data transfer, however, such data transfer must be
completed using RWDN = 10 MB, which is not optimized for the 60 ms RTT. Figure 4b shows another cycle of
three images being transmitted. The first image is transmitted over the secondary path, i.e., before the disrupted
lightpath is restored (shown in grey in Fig. 3). The restored lightpath becomes operational while the second
image is being transmitted. A zoomed-in chart of the second image data transfer is provided, in which the effect
of the RTT change (from 60 ms to 20 ms) is clearly visible (about 68.8 second into the experiment). The TCP
transfer rate then quickly reaches 1 Gbps. The last image is entirely transmitted over the restored primary path.

Figure 4. TCP data transfer rate for 3 Kinect images (each about 60MB) as required by the Godel application.

5. CONCLUSIONS
This study demonstrates that a well-orchestrated SDN two-layer network can provide a responsive and
trustworthy infrastructure for completing a robotic application without data-loss despite a single fiber outage.
Redundancy in the network is provided by two mechanisms, one at the Ethernet layer and one at the optical
layer. The TCP transfer rate is only in part affected by the outage, when the secondary path RTT is considerably
larger than the primary path RTT.

ACKNOWLEDGEMENTS
This research was supported by NSF Grants No. CNS-1405405, CNS-1409849, ACI-1541461,
and CNS-1531039.

REFERENCES
a MoveIt Pkg for an Industrial Robot