Analysis of In-order Packet Delivery Network Policy Enforcement Function

Stuart Baxley
University of Houston
Houston, TX
smbaxley@uh.edu

Nick Bastin
University of Houston
Houston, TX
nick.bastin@gmail.com

Deniz Gurkan
University of Houston
Houston, TX
dgurkan@uh.edu

I. Introduction

Industrial Control System (ICS) networks face novel challenges in risk management, feature agility, and deployment flexibility. Essential hardware control systems may have a lifetime of decades while the need for business features and the network security landscape evolve on a daily basis. Even the mix of common protocols for network connectivity is likely to undergo significant market disruption over the 50+ year lifetime of a large industrial complex. Given this reality, the University of Houston Networking Lab [4] has embarked upon an effort, facilitated by the Department of Energy CREDC[2] program, to decouple the long development cycles of hardened industrial equipment from the ever-changing realities of both the local and wide-area networks they must use to transport essential sensor data and control messages.

We are developing a specification for a Network Function (NF) with the express purpose of defining a standard behavior for per-flow policy enforcement, allowing operators to specify varying policies (or combinations of policies) for packet flows being transmitted between two trusted and/or reliable enclaves via an untrusted or unreliable segment. These policies allow features – like packet signatures, sequence numbers, local timestamping, etc – to be added without hardware downtime or vendor firmware availability. For example, devices incapable of running complex network state machines for protocols such as TCP can nonetheless be relied on to provide reliable data delivery given the application of appropriate policy. Similarly sensors and control mechanisms designed decades ago can benefit from modern HMAC [3] signatures to guarantee data integrity, without ever having to be upgraded or replaced.

While our objective is a specification that will be handed over to existing ICS vendors – and not a performant NF implementation – we need to be able to quantify both the utility of given policies and also their resource requirements such that vendors can make reasonable business decisions about which policies to support and what hardware will be required to do so. The experiment outlined in this paper is one example of how we are breaking up the specification evaluation into discrete pieces (as partially exposed in Figure 1) and using the flexibility and efficiency of GENI [7] to provide reproducible methods, data and analytics for each draft policy in a wide variety of configurable network conditions.

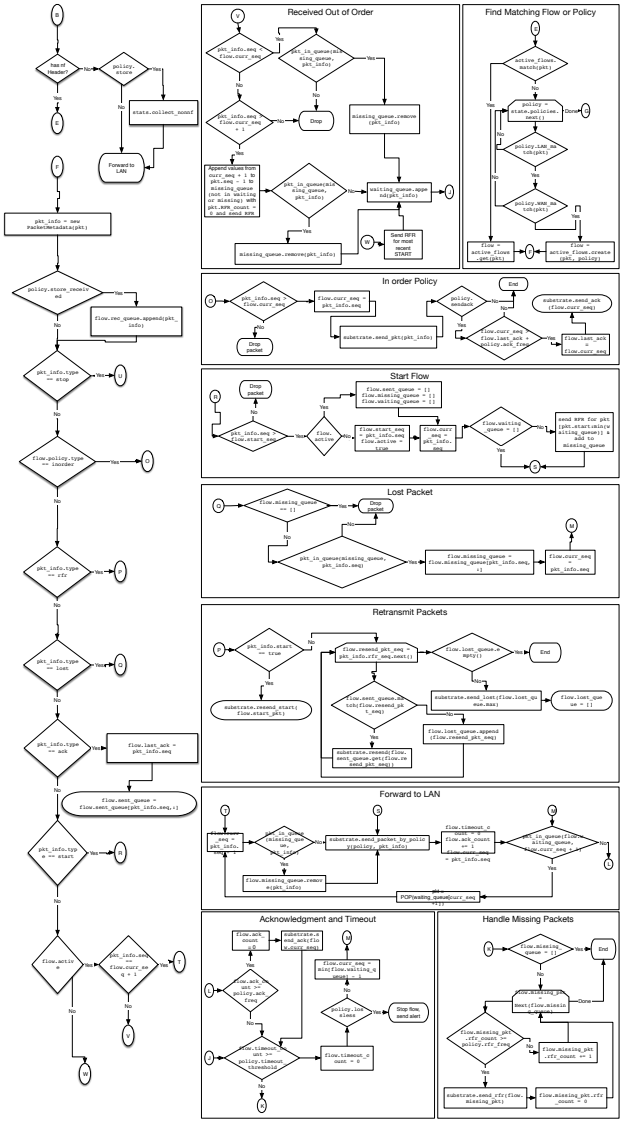

Fig. 1. Draft specification flow chart of WAN-side packet handling for several concurrent policies

II. Experiment Goals

The experiment described here evaluates a guaranteed in-order delivery policy for existing connectionless packet flows, expected to be used in commodity transport environments where satellite or long range MIMO radio links [6, 10] alter the order of packet delivery. The same baseline configuration (number of sites, sensors, etc.) is run under varying network conditions to explore how an increasing percentage of re-ordered packets will impact final message forwarding delay in the control application, as well as resource requirements for buffered packets at the Network Function.

III. Experiment Details

The experiment topology deployed on the GENI VTS testbed [8] consists of groups of sensor sites with a configurable number of sensor nodes transmitting emulated control data across an unreliable network to a management site. To quantify resource usage we must track the queue depth at the network function as well as recording the timing in which events (such as queuing or forwarding packets) occur under a wide range of network conditions (in this case, increasing amounts of reordered packets), so the deployed NF endpoints record a log of these events. For validation purposes the experiment also orchestrates packet capture at various points in the network – as seen in Figure 2 – in order to provide reference timing data and allow for debugging and investigating any unexpected behavior.

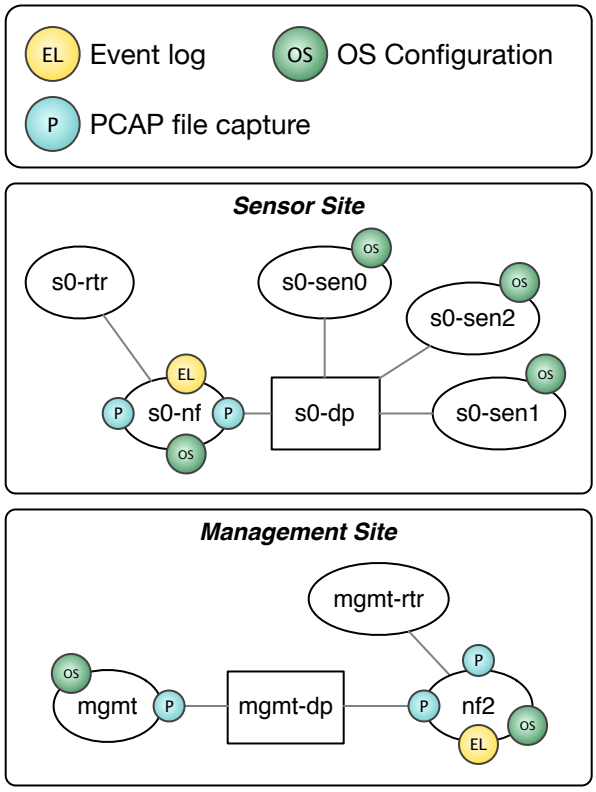

Fig. 2. Data capture points in component topology sites

Each network function relies on an ICS system operator to set per flow polices defining the actions the network function should take to buffer, drop, or forward packets. Packets are matched via a filter that passes matching packets to the appropriate handler to enforce policy decisions. In the case of our experiment we configure the experiment orchestration system [5] to allocate a lossless policy function for each unique sensor. The orchestration system utilizes the input from the GENI Manifest RSpec to extract the number of sensors at each site, as well as their interface MAC addresses for policy matching.

In the case of in-order policy, control data leaving the sensor site is encapsulated in a header containing a sequence number (incurring an additional packet overhead of four bytes). The sequence number identifies the ordering of the flow as it left the originating facility and allows for flow reassembly on the receiving side in the original order.

The ”commodity transport” core between the sites is where network impairments (delay and reordering) are applied. The degree and character of unreliability is easily managed by the VTS aggregate manager through the GENI Request RSpec, as well as dynamically during experiment runtime via the GENI Perform Operational Action API [1]. The configuration for the experiment results in this paper is 1000ms of delay (common for geosynchronous satellite transport) and 6-22% packet reordering in 2% increments. The number of sensor sites and number of sensors within each site, as well as the core impairments, are fully programmable resulting in quick scalability as VTS allows for rapid buildup and teardown of entire test networks in minutes.

The series of experiment runs are performed through a set of purpose built scripts, utilizing a custom experiment orchestration system written at the University of Houston, as well the standard geni-lib [9] Python library. A reservation script creates the topology at your site of choice, configurable with the number of sites and sensors at each site, as well as the network impairments. The topology is configurable to exclude the network functions in order to collect baselining data as well. A second script orchestrates the experiment by connecting to the remote devices via SSH, installing the necessary files, writing the packet filters, and starting the necessary services (network functions at each site, sensor emulators, etc). After the designated sensor run duration, the orchestrator stops the processes, terminates the connections, and uploads generated data files (event logs, pcaps, OS data, etc.) to the user’s Dropbox.

IV. Data Analysis

Network Function event data uploaded to Dropbox from within each experiment run is extracted and passed into an analysis script. This script produces two panel plots: the first shows a histogram of the frequency in which packets are delayed by the network function as shown in Figure 5, the second shows the total amount of time each network function contains the specified number of packets in the queue as shown in Figure 4. Note the broken y-axis scale on both panel plots, allowing for minor changes in additional delay and queue depth to be assessed. Each subplot of the figure is generated using data from the six sensors running for each experiment allowing for quick turn-around time while providing reproducibility. The queue depth bar chart is generated by taking the average of the total time spent per queue depth of each

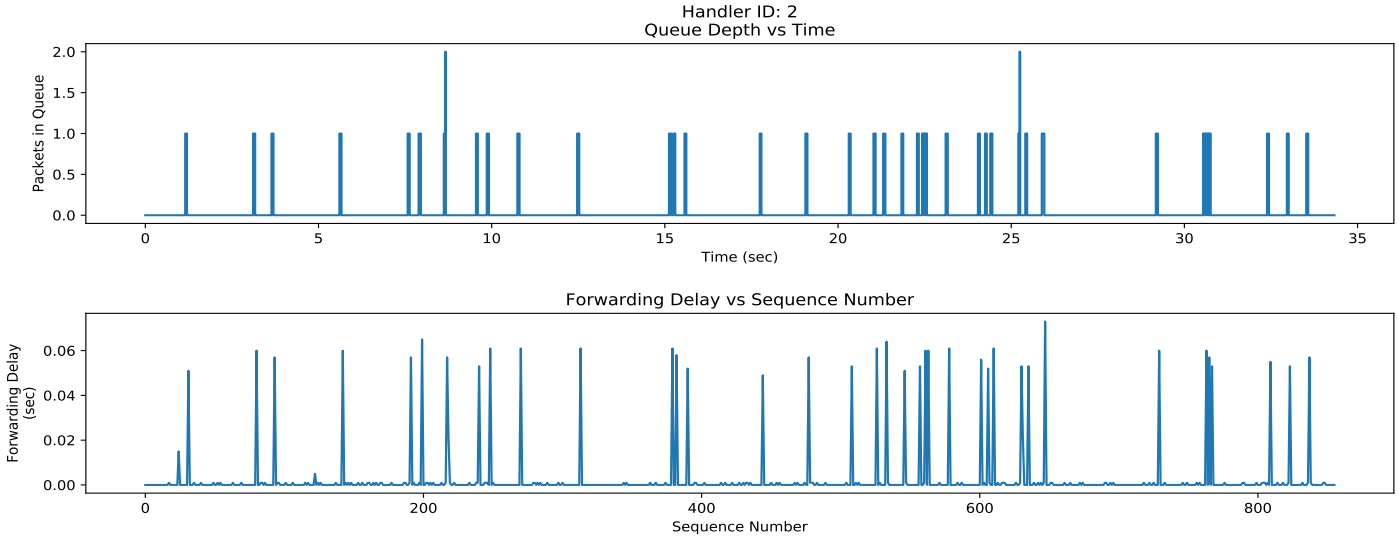

Fig. 3. Combined plot of a single sensor queue depth and forwarding time with 12% reordering

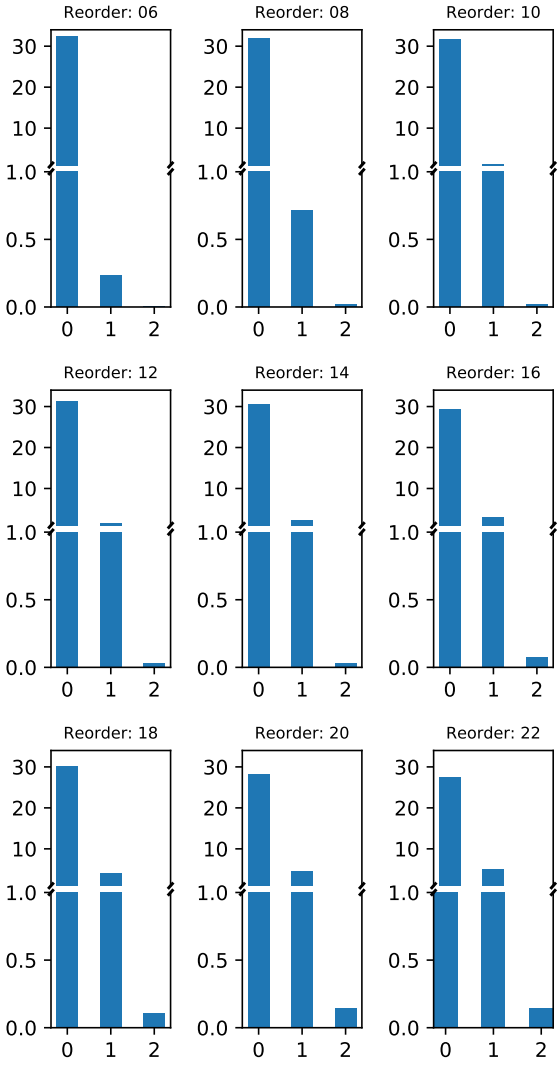

Fig. 4. Time spent per queue depth

of the six runs. The histogram data combines all of the six packet streams received and processed by the NF during the experiment run. The analysis script is configurable to generate arbitrary panel dimensions (NxM plots) and can also generate individual plots of instantaneous queue depth and per-packet forwarding delay as seen in Figure 3.

V. Conclusion

Both panels confirm the assumption that increasing reordering on the network results in longer delay times for out-of-order packets and higher utilization of Network Function buffer space. Still, the delay histogram reveals that in networks with large amounts of existing delay the additional overhead is minimal even with large amounts of reordered packets. Throughout all of the experiment runs up to twenty-two percent reordering the maximum additional packet delay incurred by the Network Function is under one hundred milliseconds and is more likely closer to fifty milliseconds. Furthermore, the queue depth bar charts show that the additional buffer required by a Network Function operating in these conditions is at most three packets per flow, and is likely bounded to a reasonable amount even in extremely poor network conditions.

This data is extremely valuable to operators in understanding where to apply certain policies – if the value of in-order delivery is worth minor jitter in extraordinary environments – as well as to vendors when determining how much overhead they will need to build into their products to support these policies. A similarly thorough evaluation of each policy is essential to vendor and operator acceptance of our specification, and the scripting allowing reproducible results on GENI resources provides for rigorous expansion of analysis as novel questions arise from real-world use of this function.

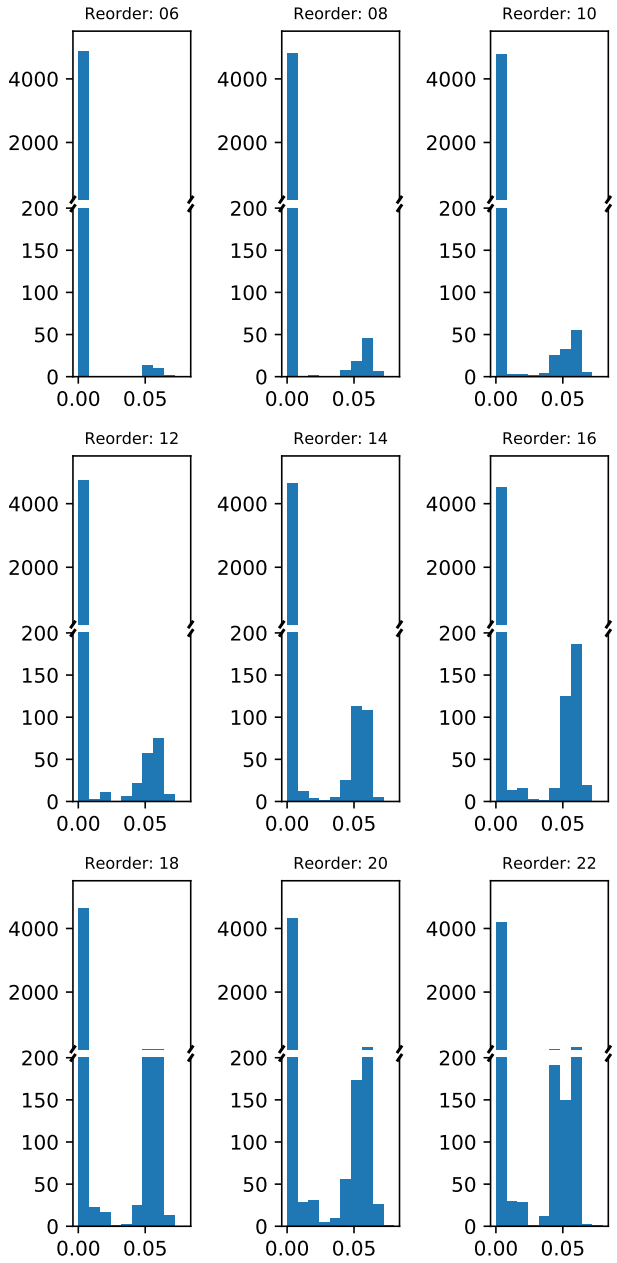

Fig. 5. Distribution of additional forwarding delay

VI. Acknowledgments

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000780. [1]

[1]Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions ofauthors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] GENI Community. *GENI AM API Specification - POA*. URL: http://groups.geni.net/geni/wiki/GAPI_AM_API_V3#PerformOperationalAction (visited on 01/22/2018).

[2] Cyber Resilient Energy Delivery Consortium. URL: https://cred-c.org/ (visited on 01/22/2018).

[3] et. al. H. Krawczyk. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104. RFC Editor, Feb. 1997. URL: http://www.rfc-editor.org/rfc/rfc2104.txt.

[4] UH Networking Laboratory. URL: http://www.uh.edu/tech/netlab/ (visited on 01/22/2018).

[5] UH Networking Laboratory. *UH Experiment Orchestration Library*. URL: https://bitbucket.org/UH-netlab/uhexp (visited on 01/22/2018).

[6] J. Li et al. "A survey of packet disordering existing in networked control systems". In: *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. June 2015, pp. 1797–1801. DOI: 10.1109/CYBER.2015.7288219.

[7] Rick McGeer et al., eds. *The GENI Book*. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-33767-8. DOI: 10.1007/978-3-319-33769-2. URL: http://dx.doi.org/10.1007/978-3-319-33769-2.

[8] Barnstormer Softworks. *GENI Virtual Topology Service*. URL: http://geni-vts.readthedocs.io (visited on 01/22/2018).

[9] Barnstormer Softworks. *geni-lib*. URL: http://geni-lib.readthedocs.io (visited on 01/22/2018).

[10] A. L. Toledo and Xiaodong Wang. "TCP performance over wireless MIMO channels with ARQ and packet combining". In: *IEEE Transactions on Mobile Computing* 5.3 (Mar. 2006), pp. 208–223. ISSN: 1536-1233. DOI: 10.1109/TMC.2006.37.