Poster Abstract: All-to-all Communication in Multi-hop Wireless Networks with Mixer

Fabian Mager*, Johannes Neumann*, Carsten Herrmann†, Marco Zimmerling*, Frank Fitzek†
* Networked Embedded Systems Group, TU Dresden, Germany
†Deutsche Telekom Chair of Communication Networks, TU Dresden, Germany
firstname.lastname@tu-dresden.de johannes.neumann@mailbox.tu-dresden.de

ABSTRACT

Cyber-physical systems (CPS) use distributed feedback loops to control physical processes. Designing practical distributed CPS controllers often benefits from a logically centralized approach, where each node computes the control law locally based on global knowledge of the system state. We present Mixer, an all-to-all communication scheme that enables all nodes in a multi-hop low-power wireless network to exchange sizable packets with one another. Mixer's design integrates synchronous transmissions with random linear network coding, harnessing the broadcast nature of the wireless medium. Results from testbed experiments with an early Mixer prototype show that our design reduces latency by 1.1–2.6× for 16–96-byte packets compared with the state of the art, while providing a reliability above 99.9 % in most settings we test.

1. OVERVIEW

Wireless cyber-physical systems (CPS) bring unprecedented opportunities by integrating sensing, control, and actuation into feedback loops for controlling physical processes. Computing the control law locally at each node rather than at a centralized controller offers better scalability and fault tolerance. To ease the design of practical distributed CPS controllers, it is often beneficial if one can assume that all nodes have global knowledge of the current system state [5]. To this end, a communication scheme is needed that supports exchanging sizable packets among all nodes in a multi-hop wireless network in a fast, reliable, and efficient manner.

State-of-the-art solutions, such as Chaos [2], partially meet these requirements. However, Chaos was primarily designed for data aggregation, and scales sub-optimally when nodes need to exchange larger chunks of raw data, called *payloads*.

Mixer in a nutshell. We present Mixer, an all-to-all communication scheme for multi-hop wireless networks. By integrating synchronous transmissions with random linear network coding [1], Mixer nodes combine several payloads before relaying. To exploit the former, we built Mixer on top of Chaos [2], but using the latter, Mixer needs fewer transmissions than Chaos to convey the same amount of information, significantly reducing latency especially for larger payloads.

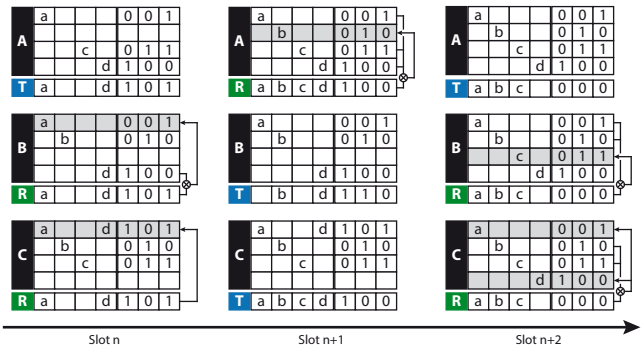


Figure 1: Example execution of Mixer in a 2-hop network with 4 nodes A, B, C, and D. *Using network coding, Mixer exchanges four different payloads within three slots—without network coding, at least one more slot would be needed.*

Network coding. The basic idea of network coding is to apply coding operations on incoming payloads at every node in the network. In Mixer, a coding operation is a linear combination of received payloads. This translates into a system of linear equations, which is maintained by every node.

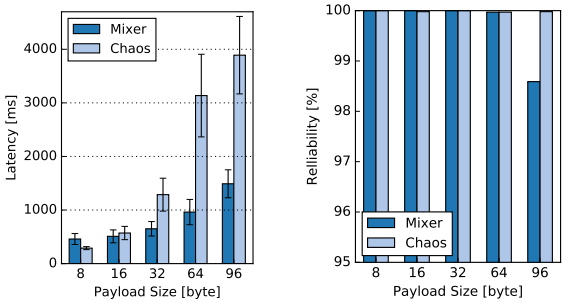
$$\underbrace{\begin{pmatrix} e_{11} & \dots & e_{1n} \\ \vdots & \ddots & \vdots \\ e_{n1} & \dots & e_{nn} \end{pmatrix}}_{\text{Encoding Matrix}} * \underbrace{\begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}}_{\text{Payloads}} = \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}}_{\text{Encoded Payloads}} \quad (1)$$

Each row in the *encoding matrix* E represents an *encoding vector* e, determining which payloads are combined to create an *encoded payload*. The elements of E must be from a finite field, in our case GF(2), so encoding means to XOR several payloads. But how should a node choose e? One way, known as random linear network coding [1], is to choose e randomly each time before relaying. This approach is advantageous as it does not require the nodes to know the network topology.

After choosing a random encoding vector e, a node computes an encoded payload and puts it together with e into a packet. According to (1), nodes receiving the packet store the encoded payload in vector c and e in matrix E if this increases the rank of E. The system of linear equations has a single unique solution when E reaches full rank; that is, a

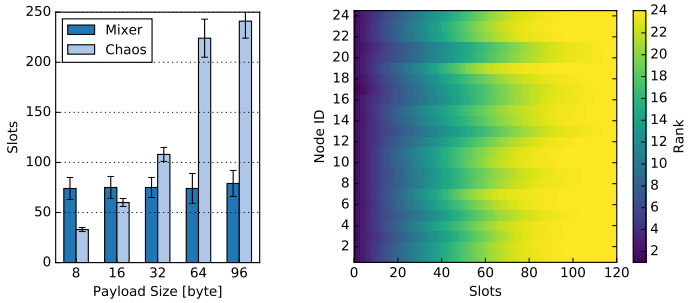
SenSys '16 November 14-16, 2016, Stanford, CA, USA

ACM ISBN 978-1-4503-4263-6/16/11…$15.00
DOI: http://dx.doi.org/10.1145/2994551.2996706



(a) Latency. (b) Reliability.

Figure 2: Performance of Mixer and Chaos against payload size. *Mixer scales better than Chaos with increasing payload size, while providing very high reliability.*



(a) Slots until completion. (b) Rank of each node over time.

Figure 3: Closer look at the functioning of Mixer and Chaos. *Chaos needs more and more slots to exchange increasingly larger payloads, whereas Mixer needs 70–80 slots irrespective of the payload size.*

node has received n linearly independent encoding vectors. As a result, a node can retrieve the payloads of all nodes.

Example execution. Efficiently integrating network coding with the distributed protocol operation is a key challenge we face in Mixer. To get a feel for our solution and the benefits of network coding, Figure 1 shows an example execution of Mixer in a 2-hop network with 4 nodes A, B, C, and D.

Mixer retains the basic protocol operation of Chaos: communication occurs in *rounds* divided into *slots*. In each slot, a node either receives or transmits a packet, and performs some processing. Each node maintains a matrix where rows correspond to received packets. Every packet consists of two parts: in the example shown in Figure 1, the first part is the encoding vector, and the second part is the 3-bit payload. Entries in the encoding vector indicate which position refers to which node. Rows below matrices show which packets are transmitted (T) or received (R) by the nodes in each slot.

In slot n, node A transmits a randomly chosen combination of already received packets: the first row XOR-ed with the last row. B receives the packet and uses its prior knowledge of d to decode and retrieve a. Since B has not yet received a packet containing a, the rank of B's matrix grows. Instead, C cannot decode as it knows neither a nor d. Nevertheless, C stores the received payload, which also increases its rank. In slot $n+1$, B and C both transmit, yet due to the capture effect [3] A receives the packet from C, decodes, and retrieves b. The matrix of A has now full rank, meaning that A knows the payloads of all other nodes. In slot $n+2$, A transmits a packet that makes B and C also reach full rank. The example shows that using network coding, Mixer exchanges four different payloads within three slots—without network coding at least one more slot would be needed.

2. PRELIMINARY RESULTS

Settings. We compare the performance of our Mixer prototype against Chaos by running experiments on FlockLab [4], using a multi-hop network of 24 TelosB nodes. We consider key CPS performance metrics: *latency* as the time from the start of a round until a node has received all payloads, or the end of a round is reached; and *reliability* as the fraction of correctly received payloads in a round. We compute average and standard deviation across all nodes and 600 rounds.

Results. Figure 2 plots performance of Mixer and Chaos for different payload sizes. We see that Mixer scales significantly better than Chaos while providing the same high reliability.[1] By coding multiple payloads instead of concatenating them, Mixer reduces latency compared to Chaos, for example, from 571 ms to 509 ms (1.1×) and from 3890 ms to 1490 ms (2.6×) for 16-byte and 96-byte payloads, respectively.

To understand the decrease in latency, especially for larger payloads, we plot in Figure 3a the number of slots needed to exchange all payloads. Because the maximum IEEE 802.15.4 packet size of 128 bytes limits the number of payloads Chaos can fit into a packet, Chaos needs more and more rounds and hence more and more slots as the payload size increases. By contrast, payload size has no impact on the number of slots in Mixer; the increase in Mixer's latency is mainly due to an increase in the time required to process larger payloads. In doing so, most nodes reach full rank after 70–80 slots as visible in Figure 3b, which plots the rank of each of the 24 nodes throughout a typical Mixer round in our experiments.

3. ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG) within the Cluster of Excellence "Center for Advancing Electronics Dresden (cfaed)" and through Priority Programs 1798 and 1914.

4. REFERENCES

[1] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *IEEE ISIT*, 2003.

[2] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale. In *ACM SenSys*, 2013.

[3] K. Leentvaar and J. H. Flint. The capture effect in FM receivers. *IEEE Trans. Commun.*, 24(5), 1976.

[4] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel. FlockLab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *ACM/IEEE IPSN*, 2013.

[5] S. Trimpe and R. D'Andrea. The balancing cube: A dynamic sculpture as test bed for distributed estimation and control. *IEEE Control Syst. Mag.*, 32(6), 2012.

[1]The slight drop in Mixer's reliability for 96-byte payload is due to a bug in our prototype implementation.