Light-weight Network Health Monitoring

Yi-Hsuan Chiang* Matthias Keller† Roman Lim† Polly Huang* Jan Beutel†
*Graduate Institute of Communication Engineering, National Taiwan University, Taiwan
†Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland
{junctionQQ, huang.polly}@gmail.com {kellmatt, lim, beutel}@tik.ee.ethz.ch

ABSTRACT

As the application of WSNs for long-term monitoring purposes becomes real, the issue of WSN system health monitoring grows increasingly important. Manually understanding the root causes of an observed behavior is time-consuming and difficult, often knowledge of prior behavior is necessary for understanding the potential risk on the long-term system performance. The challenges lie in the balance between the amount of system data collected and the level of detail in which state can be inferred from this data. In this paper, we propose a lightweight runtime logging and corresponding network state inference mechanism that enables scalable WSN health monitoring. Concretely, we propose that nodes only report their internal state on the occurrence of important events. Having a very low computational complexity and message overhead within the sensor network, reported events are analyzed at a less constrained network sink.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

Keywords

Wireless Sensor Networks, Environmental Monitoring, Health Monitoring, Long Term, Data Analysis

1. INTRODUCTION

Wireless sensor networks are well-suited for long-term data collection applications [4]. Not only being challenged by highly unpredictable, fast-changing conditions for wireless communication, the system performance is also at risk due to many other external factors [1], *e.g.*, mechanical damage. While short-term dynamics in the structure of a network are tolerable subject to the latency requirements of an application, a severe decrease in system performance, *e.g.*, long-lasting node isolations, might even need human intervention, *e.g.*, a hardware replacement.

Detecting and localizing anomalies is essential for a timely assessment of potential risks. Given a snapshot of the current system state, understanding the underlying root causes for the observed behavior can be a difficult problem [5].

In this paper, we propose a novel runtime health monitoring system that runs at the sink of a wireless sensor network.

IPSN'12, April 16–20, 2012, Beijing, China.
ACM 978-1-4503-1227-1/12/04.

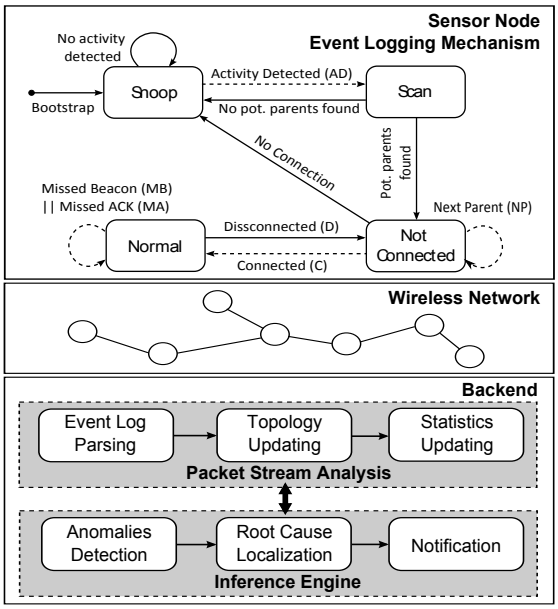

Figure 1: Health monitoring architecture

Based on a logging mechanism that only generates data on the occurrence of important events, internal network state is efficiently captured and transported to the sink. Monitoring algorithms running at the sink continuously analyze reported events to understand the system behavior, to infer root causes of undesired behavior, and to notify a network operator if a situation is severe. The usefulness of the proposed system is demonstrated using long-term data that has been collected at a real-world deployment.

2. SYSTEM ARCHITECTURE

The proposed event logging mechanism is tightly integrated into the Dozer [2] cross-layer communication stack used. The operation of Dozer on a sensor node can be described as a state machine with four states [4], see Figure 1. Here, dashed lines denote state transitions in which an event is emitted. The occurrences of two classes of events are recorded: 1) Internal transitions of the network protocol, *e.g.*, a transition from connected to disconnected operation. 2) Communication errors, *e.g.*, failed packet transmissions. An event has three properties: 1) Event_ID: denoting the type of event, 2) Event_Time: the time of event occurrence and 3) Event_Value: additional information, *e.g.*, the address of a new child node. Available event types are listed

Event_ID	meaning	Event_Value
10	Connected (C)	Parent ID
11	Disconnected (D)	*N/A*
13	Missed Beacon (MB)	#. times
14	Missed ACK (MA)	#. times
16	Next Parent (NP)	#. pot. parents

Table 1: Event types

in Table 1. Up to seven events are transmitted within a single event log packet. In contrast to periodically generated application data packets, *aperiodic* event log packets are only produced on the occurrence of interesting events.

3. HEALTH MONITORING

We propose two components for monitoring incoming traffic at the backend, see Figure 1. According to the event feedback, the *Packet Stream Analysis* component rebuilds the network topology and reassembles a global state composed of each node's current status. Those accumulated events of each node provide valuable diagnostic information. Hence, the *Packet Stream Analysis* component also extracts events from the packet stream and statistically updates high-level metrics. Three of them are listed as follows:

- **Disconnection Duration**: Time difference between event *D* and the next event *C*. It reflects the time needed for the node returning to the network.
- **Number of Potential Parents**: The Event_Value returned by the event *NP* reflects the node's tolerance to topology changes.
- **Number of Handshaking Attempts**: Since *NP* would be recorded when a node attempts to establish a connection with a potential parent node while traversing the list of potential parents, the number of reported *NP* events is used as an index of connection recovering ability.

The *Inference Engine* is responsible for detecting anomalies and inferring the root cause using above mentioned diagnostic metrics. In detail, when the application data throughput decreases, the *Inference Engine* is triggered to localize isolated nodes in the network. Here, a node is diagnosed as isolated if the symptom persists longer than the learned 90th percentile of its *Disconnection Duration*. If the throughput from several sensor nodes decreases at the same time, the *Inference Engine* tries to locate a potentially faulty link including potentially affected nodes. If the throughput of all nodes dropped at once, there is likely a failure at the sink.

4. CASE STUDY

The data used here originates from a real-world PermaSense [3] deployment that is located at the Matterhorn. The WSN consists of 22 nodes and a base station. We extract one month of data from Oct. 2010 for evaluating our proposed health monitoring mechanism.

Depending on the sensor configuration, every 120 seconds each node generates up to six kinds of application data packets. During the analyzed month, we find 500 event log packets out of in total 65,000 received packets. Thus, the introduced overhead is less than 1%.

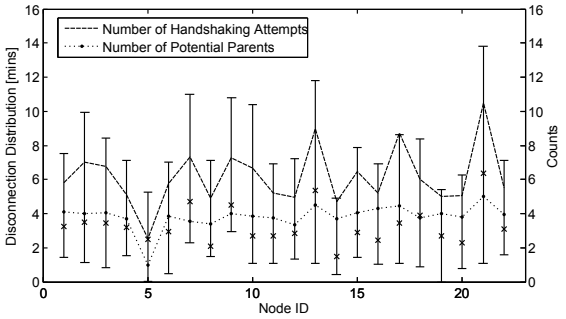

Figure 2: Diagnostic metrics that have been inferred from packets recorded at 22 nodes in Oct. 2010

In Figure 2, we show the statistics of the used diagnostic metrics. The y-axis on the left side represents the *Disconnection Duration* of each node. The 30th, 50th, and 70th percentile of the *Disconnection Duration* are plotted respectively along the error bar. The dashed and dotted curves denote the average counts of found potential parents and needed handshaking attempts until a node successfully (re)connected to the network, respectively.

Distinctively, we are able to diagnose node 5 being isolated repeatedly in Oct. 2010 due to the observations that (1) there is a persistent application data throughput drop, and (2) that the drop in throughput persists longer than the 90th percentile of the historical *Disconnection Duration* of node 5. Additionally, we can also observe that the communication ability of node 5 is dependent on the availability of only one available potential parent.

5. CONCLUSIONS

This preliminary results demonstrate that we can benefit from the light-weight logging mechanism. Our intention is to further evaluate available metrics for improving our algorithms for automated diagnosis at the backend.

6. ACKNOWLEDGEMENTS

The work presented was supported by NCCR-MICS, a center supported by the Swiss National Science Foundation under grant number 5005-67322, and from the Swiss NanoTera.ch initiative.

7. REFERENCES

[1] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli. The hitchhiker's guide to successful wireless sensor network deployments. In *Proc. of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys'08)*, 2008.

[2] N. Burri, P. von Rickenbach, and R. Wattenhofer. Dozer: ultra-low power data gathering in sensor networks. In *Proc. 6th Int'l Conf. Information Processing Sensor Networks (IPSN '07)*, pages 450–459, 2007.

[3] A. Hasler, I. Talzi, J. Beutel, C. Tschudin, and S. Gruber. Wireless sensor networks in permafrost research – concept, requirements, implementation and challenges. In *Proc. 9th Int'l Conf. on Permafrost (NICOP)*, 2008.

[4] M. Keller, M. Woehrle, R. Lim, J. Beutel, and L. Thiele. Comparative performance analysis of the permadozer protocol in diverse deployments. In *Proc. 6th IEEE Int'l Workshop on Practical Issues in Building Sensor Network Applications (SenseApp '11)*, pages 969–977, 2011.

[5] Y. Liu, K. Liu, and M. Li. Passive diagnosis for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 18(4):1132–1144, 2010.