Troubleshooting Multihop Wireless Networks

Lili Qiu
lili@cs.utexas.edu
Univ. of Texas at Austin

Paramvir Bahl
bahl@microsoft.com
Microsoft Research

Ananth Rao
ananthar@cs.berkeley.edu
UC Berkeley

Lidong Zhou
lidongz@microsoft.com
Microsoft Research

ABSTRACT

Effective network troubleshooting is critical for maintaining efficient and reliable network operation. Troubleshooting is especially challenging in multihop wireless networks because the behavior of such networks depends on complicated interactions between many unpredictable factors such as RF noise, signal propagation, node interference, and traffic flows. In this paper we propose a new direction for research on fault diagnosis in wireless networks. Specifically, we present a diagnostic system that employs trace-driven simulations to detect faults and perform root cause analysis. We apply this approach to diagnose performance problems caused by packet dropping, link congestion, external noise, and MAC misbehavior. In a 25 node multihop wireless network, we are able to diagnose over 10 simultaneous faults of multiple types with more than 80% coverage. Our framework is general enough for a wide variety of wireless and wired networks.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations – Network management

General Terms

Algorithms, Management, Measurement, Performance, Experimentation.

Keywords

Multihop wireless networks, network diagnosis, network management, simulation.

1. INTRODUCTION

Network management in multihop wireless networks is a necessary ingredient for providing high-quality, reliable communications among the networked nodes. Unfortunately, it has received little attention until now. In this paper, we focus on network troubleshooting, the component of network management responsible for maintaining the "health" of the network and ensuring its smooth and continued operation [1].

SIGMETRICS'05, June 6–10, 2005, Banff, Alberta, Canada.
ACM 1-59593-022-1/05/0006.

Troubleshooting a network, may it be wired or wireless, is a difficult problem. This is because a network is a complex system with many inter-dependent factors that affect its behavior. The factors include networking protocols, traffic flows, hardware, software, different faults, and most importantly the interactions between them. Troubleshooting a multihop wireless network is even more difficult due to unreliable physical medium, fluctuating environmental conditions, complicated wireless interference, and limited network resources. We know of no heuristic or theoretical technique that captures these interactions and explains the behavior of such networks.

To address these challenges, we propose a novel troubleshooting framework that integrates a network simulator into the management system for detecting and diagnosing faults occurring in an operational network. We collect traces, post-process them to remove inconsistency, and then use them to recreate in the simulator the events that have taken place inside the real network.

For our system to work, we must solve two problems: (i) accurately reproduce inside the simulator what just happened in the operational network; and (ii) use the simulator to perform fault detection and diagnoses.

We address the first problem by taking an existing network simulator (e.g., Qualnet [5], a commercially available packet-level network simulator) and identify the traces to drive it with. (Note: although we use Qualnet in our study, our technique is equally applicable to other network simulators, such as ns-2 [2], OPNET [3] etc.).

We concentrate on physical and link layer traces, including received signal strength, and packet transmission and retransmission counts. We replace the lower two networking layers in the simulator with these traces to remove the dependency on generic theoretical models that do not capture the nuances of the hardware, software, and radio frequency (RF) environment.

We address the second problem with a new fault diagnosis scheme that works as follows: the performance data emitted by the trace-driven simulator is considered to be the expected baseline ("normal") behavior of the network and any significant deviation indicates a potential fault. When a network problem is reported/suspected, we selectively inject a set of possible faults into the simulator and observe their effect. The fault diagnosis problem is therefore reduced to efficiently searching for the set of faults that, when injected into the simulator, produce network performance that matches the observed performance. This approach is significantly different from the traditional signature based fault detection schemes.

Our system has the following three benefits. First, it is flexible. Since the simulator is customizable, we can apply our fault detection and diagnosis methodology to a large class of networks operating under different environments. Second, it is robust. It can

capture complicated interactions within the network and between the network and the environment, as well as among the different faults. This allows us to systematically diagnose a wide range and combination of faults. Third, it is extensible. New faults are handled independently of the other faults as the interaction between the faults is captured implicitly by the simulator.

We have applied our system to detect and diagnose performance problems that arise from the following four faults:

- Packet dropping. This may be intentional or may occur because of hardware and/or software failure in the networked nodes. We care about persistent packet dropping.
- Link congestion. If the performance degradation is because of too much traffic on the link, we want to be able to identify this.
- External noise sources. RF devices may disrupt on-going network communications. We concern ourselves with noise sources that cause sustained and/or frequent performance degradation.
- MAC misbehavior. This may occur because of hardware or firmware bugs in the network adapter. Alternatively, it may be due to malicious behavior where a node deliberately tries to use more than its share of the wireless medium.

The above faults are more difficult to detect than fail-stop errors (e.g., a node turns itself off due to power or battery outage), and they have relatively long lasting impact on performance, In this paper, we focus only on identifying the faults, and not on the corrective actions one might take.

We demonstrate our systems ability to detect random packet dropping and link congestion in a small multihop IEEE 802.11a network. We demonstrate detection of external noise and MAC misbehavior via simulations because injecting these faults into the testbed in a controllable manner is difficult. In a 25 node multihop network, we find that our troubleshooting system can diagnose over 10 simultaneous faults of multiple types with more than 80% coverage and very few false positives.

To summarize, our primary contribution is to show that a trace-driven simulator can be used as a real-time analytical tool in a network management system for detecting, isolating, and diagnosing faults in an operational multihop wireless network. In the context of this system, we make the following three contributions:

- We identify traces that allow a simulator to mimic the multihop wireless network being diagnosed.
- We present a generic technique to eliminate erroneous trace data.
- We describe an efficient search algorithm and demonstrate its effectiveness in diagnosing multiple network faults.

2. OUR APPROACH

Our management system consists of two types of software modules. An *agent*, running on every node, gathers information from various protocol layers and reports it to a management server, called *manager*. The manager analyzes the data and takes appropriate actions. The manager may run on a single node (centralized architecture), or may run on a set of nodes (decentralized architecture) [6].

The monitoring and diagnosis process starts by agents continuously collecting and transmitting their (local) view of the network's behavior to the manager(s). Examples of the information sent include traffic statistics, received packet signal strength on various links, and re-transmission counts on each link.

It is possible that the data the manager receives from the different agents results in an inconsistent view of the network. Such inconsistencies could be the result of topological and environmental changes, measurement errors, or misbehaving nodes. The *Data Cleaning* module of the manager resolves inconsistencies before engaging the analysis model.

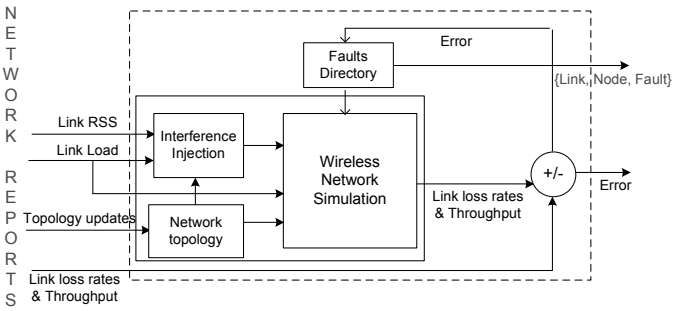

Figure 1: Root cause analysis module

After the inconsistencies have been resolved, the cleaned trace data is fed into the root-cause analysis module which contains a modified network simulator (see Figure 1). The analysis module drives the simulator with the cleaned trace data and establish the expected normal performance for the given network configuration and traffic patterns. Faults are detected when the expected performance does not match the observed performance. Root cause for the discrepancy is determined by efficiently searching for the set of faults that results in the best match between the simulated and observed network performance. The detailed search algorithm is described in our technical report [4].

3. EVALUATION

We evaluate our approach using both simulation and experimentation. We summarize the results below, and the detailed results are available in [4].

- We evaluate the overhead of collecting traces, which will be used as inputs to diagnose faults in a network. We show that the data collection overhead is low and has little effect on application traffic in the network.
- Our extensive evaluation shows that the simulation-based fault diagnosis approach is effective: around 80% of faults can be correctly identified, with close to 0 false positive.
- To handle imperfect data, we apply the inconsistency detection scheme to the raw data before feeding it to a simulator for fault diagnosis. Our results show that the scheme can accurately resolve the inconsistency.

4. ACKNOWLEDGMENTS

We are grateful to Atul Adya, Sharad Agarwal, Mike Jones, Jitendra Padhye, Venkata N. Padmanabhan, Dan Simon, Helen Wang, Alec Wolman, and anonymous reviewers for their valuable comments.

5. REFERENCES

[1] D. D. Clark, C. Patridge, J. C. Ramming, and J. T. Wroclawski. A knoweldge plane for the internet. In *Proceedings of ACM SIGCOMM*, Karlsruhe, Germany, August 2003.

[2] The network simulator – ns-2. http://www.isi.edu/nsnam/ns/.

[3] OPNET modeler. http://www.opnet.com.

[4] L. Qiu, P. Bahl, A. Rao, and L. Zhou. Fault detection, isolation, and diagnosis in multi-hop wireless networks. In *Microsoft Technical Report TR-2004-11*, 2004.

[5] The Qualnet simulator from Scalable Networks Inc. http://www.scalable-networks.com/.

[6] C.-C. Shen, C. Jaikaeo, C. Srisathapornphat, and Z. Huang. The Guerrilla management architecture for ad hoc networks. In *Proc. of IEEE MILCOM*, Anaheim, California, October 2002.