RFDump: An Architecture for Monitoring the Wireless Ether

Kaushik Lakshminarayanan, Samir Sapra, Srinivasan Seshan, Peter Steenkiste
Carnegie Mellon University
Pittsburgh, PA 15213
{kaushik, ssapra, srini, prs}@cs.cmu.edu

Abstract

Networking researchers have been using tools like *wireshark* and *tcpdump* to sniff packets on physical links that use different types of datalink protocols, e.g. Ethernet or 802.11, allowing them to monitor higher level protocols sharing these links. However, monitoring wireless links is more challenging, since the transmission medium is shared by flows using diverse datalink protocols (e.g. 802.11, Bluetooth) and physical layer schemes (e.g. QPSK and GFSK). To this end, we propose *RFDump*, a software architecture for monitoring packets on heterogeneous wireless networks. The key idea underlying our architecture is the use of a fast detection stage which can tentatively map signals to protocols very efficiently. As a result, RFDump can scale up to a modest number (5-10) of wireless technologies.

We implemented RFDump on the GNU Radio and USRP platforms. This is, to our knowledge, the first inexpensive software-based infrastructure for simultaneously analyzing multiple wireless protocols in real-time. Using traces from the real world and from a wireless emulator testbed, we show that our implementation is efficient and accurate. Further, we demonstrate that our system is extensible and scales with the addition of new protocols.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: [Measurement Techniques]; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*; C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*

General Terms

Measurement, Experimentation, Performance

Keywords

software defined radio, wireless networks, monitoring, Wi-Fi, Bluetooth, tcpdump

CoNEXT'09, December 1–4, 2009, Rome, Italy.

1. INTRODUCTION

Tcpdump, Wireshark/Ethereal and similar applications have become a critical part of the tool collections used by networking researchers, networking administrators and application developers. These tools expose the operation of a network in a detailed, cross-layer fashion. Based on this exposed information, users are able to monitor and analyze the interactions between different nodes, different protocols, different protocol layers and different applications in the network. This has enabled activities such as diagnosing network protocols, optimizing network performance and even teaching network protocol operation.

Unfortunately, applying these tools in wireless networks fails to provide the same level of insight into the operation of the network. There are two reasons for this problem. First, these tools operate at the link-layer and above. In wireless settings, the behavior of the physical layer is critical to the operation of the network. Second, these tools are limited to operation over a single network interface card (NIC), such as an 802.11 NIC. As a result, they can only report on the detailed operation of the associated network link technology. However, unlike wired networks, the physical medium over which the network operates is shared by many link technologies. For example, the 2.4 GHz unlicensed spectrum band is shared by 802.11, Bluetooth, ZigBee, cordless phones and a wide range of other link technologies. Making observations on a single link technology hides many of the node, protocol and application interactions that users are attempting to observe with such tools. In this paper, we describe the design of RFDump, a tool that extends the monitoring capabilities below the link layer and enables more effective monitoring of the wireless ether.

In order to be practical, a monitoring tool for wireless networks must meet two key requirements. First, we must be able to monitor packets that use a wide variety of protocols, so the tool must efficiently support multiple protocols and it must be easy to add new protocols in the future. Second, the tool must run in real-time so it can be used for runtime analysis and troubleshooting. Note that we do not expect our system to interact with the monitored links (i.e., it does not need to implement the link-layer protocol). As a result, our system can process transmissions after some delay (e.g., a second) but the processing must keep up with the rate of packet transmissions. In addition, while core functions, such as identifying packets and the technology they use, must occur in real time, more complex functions, such as full decoding of payloads or deep packet inspection, may only be feasible for a subset of the traffic in the ether.