Passive Online Rogue Access Point Detection Using Sequential Hypothesis Testing with TCP ACK-Pairs

Wei Wei[†], Kyoungwon Suh[†], Bing Wang[‡], Yu Gu[†], Jim Kurose[†]

[†]Department of Computer Science
University of Massachusetts, Amherst, MA 01003

[‡]Computer Science & Engineering Department
University of Connecticut, Storrs, CT 06269

UMass Computer Science Techincal Report 2006-60
Nov-5-2006

ABSTRACT

Rogue access points (unauthorized wireless access points) pose serious security threats to local networks. In this paper, we propose online algorithms to detect rogue access points using sequential hypothesis tests applied to passively-measured packet header data collected at a gateway router. The online algorithms utilize the TCP ACK-pair technique that we investigated earlier to passively differentiate wired and wireless LAN TCP traffic. We motivate and develop our sequential hypothesis tests by analyzing $M/D/1$ queues and the medium access mechanisms of 802.11. Depending on whether training sets are available, we propose two sequential hypothesis tests: one with training and the other without training. Extensive experimental validations in various scenarios and over hosts with various operating systems demonstrate the superb performance of our approach: the test with training is prompt (the detection time is mostly within 10 seconds) and extremely accurate (very low false positive and false negative ratios are obtained). Without training, the algorithm detects 60%-76% of the wireless hosts without false positives. The success of our approach lies in the fact that it leverages knowledge about the behavior of the 802.11 CSMA/CA MAC protocol, and the half duplex nature of wireless channels. Our approach is scalable and non-intrusive, requiring little deployment cost and effort, and is easy to manage and maintain.

1. INTRODUCTION

The deployment of IEEE 802.11 wireless networks (WLANs) has been growing at a remarkable rate during the past several years. The presence of a wireless infrastructure within a network, however, raises various network management and security issues. One of the most challenging issues is rogue access points (APs), i.e., wireless access points that are installed without explicit authorization from a local network management [9, 1, 3, 4]. Although usually installed by innocent users for convenience or higher productivity, rogue APs pose serious security threats to the local network. First, they potentially open up the network to unauthorized parties, who may utilize the resources of the network, steal sensitive information or even launch attacks to the network. Furthermore, rogue APs may interfere with nearby well-planned APs and lead to performance problems inside the network.

Due to the above security and performance threats, detecting rouge APs is one of the most important tasks for an IT department to manage a local network. Surprisingly, there is little research work on rogue AP detection. In contrast, a proliferation of commercial products have been developed to detect rogue APs [2, 8, 9, 1, 3, 4]. These commercial products typically monitor the RF airwaves, and may further exploit information gathered at routers and switches to detect rogue APs. Although effective under certain circumstances, they suffer various drawbacks in terms of scalability, deployment cost and accuracy, as to be discussed in more detail in Section 1.1. We are aware of only three research efforts on detecting rogue APs [10, 16, 15]. The study of [10] requires instrumenting a large number of clients (e.g., laptops) and possibly APs to monitor nearby APs. Their approach assumes that rogue APs beacon and respond to probe messages according to the IEEE 802.11 standards. As pointed out by the authors, this assumption may not always hold. In [15], the authors proposed a framework for monitoring wireless networks through wireless devices attached to desktop machines. Their method improves upon [10] by adding more tests to reduce false positives and false negatives. However, it has the same limitation as [10] that it relies on some specific characteristics of IEEE 802.11 which can be easily violated by wireless devices. The study of [16] is based on temporal characteristics of wireless traffic. It, however, does not provide a systematic or automated procedure to detect rogue APs and suffers from a number of limitations as detailed in Section 1.1.

In this paper, we propose a novel approach for *passive online* rouge AP detection based on measurements collected passively at the edge of a network. This approach roughly works as follows. We first determine online whether a host uses wireless or Ethernet connection as packets arriving at the measurement point. Once a host is determined as using wireless connection, we further check whether the host is authorized to use the wireless network by looking it up from an authorization list. If it is not in the authorized list, the AP attached to by this host is detected as a rogue AP.

The core of our rogue AP detection scheme is online determination of a host's connection type (either WLAN or Ethernet). For this purpose, we develop two sequential hypothesis tests that determine a host's connection type as packets coming in. These two tests are motivated by analyzing the intrinsic characteristics of TCP ACK-pairs from Ethernet and WLAN hosts[1]. The first test requires training data while the second one does not have such a requirement.

Our scheme for online passive rogue AP detection has the following advantages. First, it is based on the fundamental CSMA/CA MAC protocol of IEEE 802.11 and the half-duplex nature of wireless channels. Hence, it is more effective and reliable than those relying on specific characteristics of IEEE 802.11 (e.g., periodic beacon message, identities information such as MAC address or SSID). Second, our detection scheme only utilizes the characteristics of wireless traffic and hence can be equally applied to detect rogue APs, rogue NAT (Network Address Translation) box, rogue routers and rogue ad-hoc networks[2], while approaches based on RF sensing fails to detect certain types of rogues or need different mechanisms for different types of rogues [9, 4, 10, 15]. Third, since our scheme is based on online passive measurements at a single monitoring point, it is scalable, non-intrusive (to both the airwave and traffic in the network), requiring little deployment cost and effort, and easy to manage and maintain.

We apply our online scheme at a monitoring point placed at the gateway router of a campus network. Extensive experiments in various scenarios and over hosts with various operating systems demonstrate the superb performance of our approach: the sequential hypothesis test with training is prompt (the detection time is mostly within 10 seconds) and extremely accurate (very low false positive and false negative ratios are obtained). The sequential hypothesis test without training detects 60%-76% of the wireless hosts without false positives. Furthermore, our scheme can detect connection switchings and wireless networks behind a NAT box. Last, our scheme remains effective for hosts with high CPU, hard disk or network utilizations.

The rest of the paper is organized as follows. Section 1.1 describes related work. Section 2 presents the problem setting and a high-level description of our approach. Section 3 presents the analytical foundation of our scheme. Section 4 describes the core of our rogue AP detection scheme: two sequential hypothesis tests, with and without training data. Section 5 presents our online detection scheme. Section 6 describes experimental validation conducted over a campus network. Finally, Section 7 concludes the paper and describes future work.

1.1 Related work

Most existing commercial products detect rogue APs through RF sensing. The first approach is to manually scan the wireless network using sniffers on laptops or handheld devices (e.g., AirMagnet [2], and NetStumbler [8]). This manual process is expensive, time-consuming, and too intermittent to provide continuous protection of the network. To automate the scanning process and ensure continuous vigilance to rogue APs, the second approach employs large number of sensors to monitor the airwaves. The sensors can be specialized hardware devices (e.g.,[1]), wireless clients or APs [9, 4]. The basic idea is that the sensors continuously listen to (even actively scan) nearby APs and report signatures of discovered APs to a central management site. This central management site maintains a list of authorized APs and raises an alarm if a discovered AP is not in the authorization list. This approach becomes ineffective when a rogue AP spoofs signatures (e.g., MAC address, SSID, etc.). Furthermore, it is difficult to guarantee complete coverage of the network to ensure effective rogue AP detection. Last, it fails to detect rogue ad-hoc networks [15].

Prior research studies [10, 15] take a similar approach as commercial products to detect rogue APs through RF sensing. Their focus is to provide a framework for network fault diagnosis and security respectively, which leads naturally to rogue AP detection. In [10], wireless clients are instrumented to collect information about nearby APs and send the information to a centralized server. When the server receives information about an AP, it checks whether this AP is registered to determine whether it is a rogue AP. This approach is similar to those taken by commercial products of [9, 4] and has similar limitations as described above. Furthermore, it assumes that rogue APs use standard beacon messages in IEEE 802.11 and respond to probes from the clients, which may not hold in practice. Last, all unknown APs (including those in the vicinity networks) are flagged as rogue APs, which may lead to large number of false positives. The main idea of [15] is to enable dense RF monitoring through wireless devices attached to desktop machines. This study improves upon [10] by providing more accurate and comprehensive rogue AP detection. However, it has a similar limitation as [10] that it heavily relies on certain specific features of IEEE 802.11, which can be easily turned off or violated.

The work of [16] takes a completely different approach from others. It detects rogue APs through temporal characteristics of wireless networks. This scheme is based on the intuition that inter-packet arrival times of wireless traffic are more random than those of wired traffic. This study, however, suffers from a number of limitations. First, it requires the wireless APs to be directly attached or one-hop away from the monitoring point. Secondly, the detection is effective only when wireless hosts are uploading data. Third, the approach is based on visual inspection, unable to be carried out automatically.

Our study has a similar flavor as [16] in the sense that we also utilize the temporal characteristics of wireless traffic. However, our work differs significantly from [16] in several important aspects. First, our approach is based on a rigorous analysis of Ethernet and wireless traffic characteristics. Second, based on the analysis, we provide two sequential hypothesis tests to automatically detect rogue APs in real time. Third, in our approach, the placement of measurement point is much more flexible (we place it at the university gateway router in our experiments). Last, we employ the inter-arrival time of TCP ACK-pairs (instead of data packets), which is suitable for the more common scenarios that wireless devices download data from outside servers.

The core of our scheme is determining whether a host is via a wireless connection in real time. There are several

[1]The concept of TCP ACK-pair is introduced in our previous work [14]. Informally, an ACK-pair refers to two ACKs generated in response to data packets that arrive close in time to the measurement point.

[2]In a rogue ad-hoc network, an unauthorized device attached to the wired network operates in ad-hoc mode and acts as a forwarder [10].

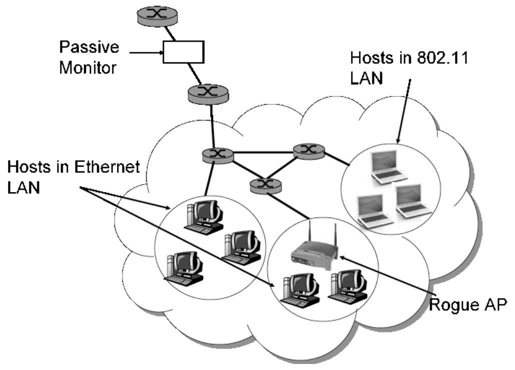

Figure 1: Problem setting: a monitoring point is located at the gateway router of a local network, capturing traffic coming in and going out of the network. The end hosts within this network are behind wired Ethernet or 802.11 WLAN.

prior studies on determining connection types. Our previous work [14] proposes an iterative Bayesian inference technique to identify wireless traffic based on passive measurements. This iterative approach is not a light-weight online algorithm and hence is not suitable for online rogue AP detection. In other studies, differentiating connection types is based on active measurements [11] or certain assumptions about wireless links (such as very low bandwidth and high loss rates) [18], which do not apply to our scenario.

Last, sequential hypothesis testing [24] provides an opportunity to make decisions as data come in, and thus is a suitable technique for our purpose. It is also used for prompt detection of portscan in [20].

2. PROBLEM SETTING AND APPROACH

We now state the problem setting and describe, at a high-level, our approach towards solving this problem. Consider a local network, e.g., a university campus or an enterprise network, as illustrated in Fig. 1. End hosts within this network use either wired Ethernet or 802.11 WLAN to access the network. An end host not authorized to use WLAN may install a rogue AP to connect to the network. Our goal is to detect rogue APs through passive measurements in real time.

For this purpose, we place a monitoring point at the gateway router of this local network, capturing traffic coming in and going out of the network. This monitoring point determines whether an end host uses Ethernet or WLAN connection. After detecting a WLAN host, the monitoring point checks whether this host is authorized to use WLAN by looking it up from an authorization list (stored locally). If this host is not in the authorization list, the monitoring point raises an alarm that this host is connected through a rogue AP.

In the above detection approach, a critical task is how to determine *online* whether a host uses WLAN based simply on *passive* measurements at the monitoring point. This is a challenging task since the monitoring point is at the edge of the network, in the middle of the path between senders and receivers. Therefore, the measurements collected at the monitoring point may not provide accurate information on the characteristics of the sender and the receiver. Furthermore, since we are interested in online detection, the detection scheme needs to be light-weight, incurring little storage and computation overhead.

We solve the above challenge by developing two sequential hypothesis tests based on intrinsic characteristics of WLAN and Ethernet connections. These sequential hypothesis tests make decisions as TCP packets (since TCP carries majority of the Internet traffic) arriving at the measurement point and roughly operate as follows. For each host being monitored inside the local network, we identify *TCP ACK-pairs* (i.e., a pair of ACKs corresponding to two data packets that arrive at the monitoring point close in time) generated by this host at the monitoring point. As will be shown in Section 3, the inter-arrival times of ACK-pairs at the monitoring point differ significantly if the host uses WLAN as compared to using Ethernet. As ACK-pairs arriving at the monitoring point, the hypothesis tests calculate the likelihoods of the host using WLAN and Ethernet. When the ratio of the two likelihoods exceeds a certain threshold, the monitoring point determines that the host uses WLAN.

In the next section, we present the analytical basis of our scheme, which demonstrates that the inter-arrival times of ACK-pairs of Ethernet and WLAN are distinguishable. We then describe the two sequential hypothesis tests along with the approach for online rogue AP detection.

3. ANALYSIS OF TCP ACK-PAIRS

In this section, we carry out an analytical study that forms the foundation of the sequential hypothesis tests in our online detection scheme. The goal of this analysis is to answer a key question: Is TCP ACK-pair a good technique to differentiate WLAN and Ethernet hosts?

To answer this question, we consider an arbitrary TCP flow in which an outside server sends data to a receiver residing in the local network, as shown in Fig. 2. The access link of the receiver is either Ethernet (Fig. 2(a)) or WLAN (Fig. 2(b)). The monitoring point lies between the sender and the receiver, at the edge of the local network. Let Δ denote the inter-arrival time of two data packets that arrive close in time at the monitoring point. The receiver returns a pair of ACKs corresponding to these two data packets, i.e., an ACK-pair, to the sender. Let Δ_A denote the inter-arrival time of this ACK-pair at the monitoring point, referred to as an *inter-ACK time*. We conjecture that, when the receiver uses a WLAN, Δ_A is larger than when the receiver uses Ethernet. Intuitively, this is due to two reasons. Firstly, in a WLAN, even if there is no contention in the channel, the receiver must wait for a random backoff interval after a previous successful transmission to avoid channel capture (see [12] and the references within). Therefore, this random backoff delay may be inserted between the ACK-pair, leading to a larger inter-ACK time. Secondly, in a WLAN, the ACKs also contend with the data packets (coming from the opposite direction) for the wireless channel. Therefore, data packets may be transmitted between the ACK-pair, and this again increases the inter-ACK time. Our analysis confirms this conjecture by demonstrating that the distributions of Δ_A under WLAN and Ethernet differ dramatically.

We now describe our analysis in more detail, starting with the assumptions and settings. We then present the analysis for Ethernet, 802.11b WLAN, and 802.11g WLAN. In the end, we briefly summarize the insights obtained from this

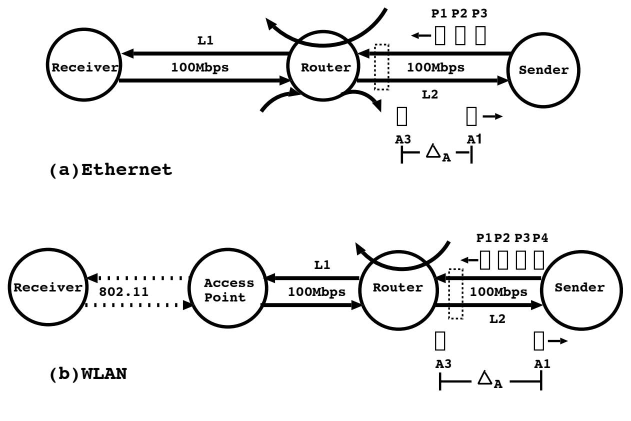

Figure 2: Settings for the analysis: (a) Ethernet (b) WLAN (802.11b or 802.11g). The dashed rectangle between the sender and the router represents the monitoring point. The pair of ACKs, A_1 and A_3, forms an ACK-pair.

analysis.

3.1 Assumptions and settings

The settings for our analysis are shown in Fig. 2. In the figure, an outside server sends data to a receiver in the local network. In the first setting (Fig. 2(a)), the receiver uses Ethernet; in the second setting (Fig. 2(b), the receiver uses 802.11b or 802.11g WLAN. We refer to these two settings as Ethernet and WLAN setting respectively. In both settings, a router resides between the sender and the receiver, and connected to the sender by link L_2 with 100 Mbps bandwidth. The monitoring point is between the sender and the router, tapping into link L_2. In the Ethernet setting, the router and the receiver are connected by link L_1 with 100 Mbps bandwidth (i.e., 100Mbps Ethernet). In the WLAN setting, an access point resides between the router and the receiver. The access point and the router are connected by link L_1 with 100 Mbps bandwidth; and the receiver is connected to the access point using 11 Mbps 802.11b or 54 Mbps 802.11g. In both the Ethernet and WLAN settings, the router is modeled as two $M/D/1$ queues, Q_D and Q_A, in the direction of data packets (i.e., from the sender to the receiver) and ACKs (i.e., from the receiver to the sender) respectively. Let ρ_D and ρ_A denote the utilization of Q_D and Q_A respectively. In the Ethernet setting, cross traffic traverses both queues at the router. In the WLAN setting, cross traffic only traverses queue Q_D; no other traffic is on the path between the router and the receiver. This implies that the access point is only utilized by the receiver. As we shall see, even when the Ethernet link is heavily utilized and the WLAN link is under the above idealized condition, the inter-ACK times of WLAN are generally larger than those of Ethernet.

We assume that the receiver implements delayed ACK policy, since this policy is commonly used in practice, implemented in both Windows and Linux [22, 7]. To accommodate the effects of delayed ACK, we consider four packets P_1, P_2, P_3 and P_4, each of 1500 bytes, sent back-to-back from the sender. Without loss of generality, we assume that packets P_1 and P_3 are acknowledged. Their corresponding ACKs A_1 and A_3 form an ACK-pair. Let Δ_A represent the inter-ACK time of ACKs A_1 and A_3 at the monitoring point. Let Δ denote the inter-arrival time of the data packets corresponding to ACKs A_1 and A_3 (namely, packets P_1 and P_3) at the monitoring point. Then $\Delta = 120 \times 2 = 240\mu s$ since the size of the packets P_i'sis 1500 bytes and the bandwidth of link L_2 is 100 Mbps. Measurement studies show that the average packet size on the Internet is between 300 and 400 bytes [23, 21]. We assume that all packets in the cross traffic are 375 bytes for ease of calculation. Then, the transmission time of a cross traffic packet on a 100 Mbps link is $30\mu s$. For ease of exposition, we define a time unit of length $30\mu s$, and refer to it as a *packet transmission time*.

In both the Ethernet and WLAN settings, we are interested in the distribution of Δ_A, i.e., the inter-ACK time of the ACK-pair A_1 and A_3. When calculating Δ_A in the Ethernet setting, the transmission time of an ACK is ignored since it is negligible.

3.2 Analysis of Ethernet

Let Δ_D be the inter-departure time of packet P_1 and P_3 at queue Q_D (i.e., the queue in the direction of data packets at the router). Discretize Δ_D using the packet transmission time as the time unit, and denote the discretized value as I_D, that is, $I_D = \lfloor \Delta_D/30 \rfloor$. Discretize Δ_A in the similar manner and denote the discretized value as I_A, $I_A = \lfloor \Delta_A/30 \rfloor$.

We now state two lemmas on the distribution of I_D and the conditional distribution of I_A given I_D respectively. Their proofs can be found in Appendix.

LEMMA 1. *Let $Z = I_D - 8$. When $\rho_D = 1$, Z follows a Poisson distribution with the parameter of* 8 *time units.*

LEMMA 2. *Suppose $I_D = x$ time units. When $\rho_A = 1$, the conditional distribution of I_A given I_D follows a Poisson distribution with the parameter of x time units.*

We next state a lemma on the marginal distribution of I_A, which is used to derive a theorem on the inter-ACK time distribution.

LEMMA 3. *When $\rho_D = \rho_A = 1$,*

$$P(I_A \le x) = \sum_{y=8}^{\infty} \frac{8^{y-8} e^{-8}}{(y-8)!} \sum_{i=0}^{x} \frac{y^i e^{-y}}{i!}$$

PROOF. This follows directly from Lemma 1 and Lemma 2. □

THEOREM 1. *In the Ethernet setting, when $0 < \rho_D, \rho_A \le 1$, $P(\Delta_A > 600\mu s) < 0.1754$.*

PROOF. When $\rho_D = \rho_A = 1$, from Lemma 3, by direct calculation, we have $P(I_A > 20) < 0.1754$, which is equivalent to

$$P(\Delta_A > 600\mu s) < 0.1754.$$

The above result also holds when $0 < \rho_D < 1$ and $0 < \rho_A < 1$ due to reasons below. When $0 < \rho_D < 1$, the inter-departure time of data packets at queue Q_D is no more than that when $\rho_D = 1$, since not all cross traffic contribute to the inter-departure time of packet P_1 and P_3. Similarly, when $0 < \rho_A < 1$, the inter-departure time of ACK A_1 and A_3 at the router is no more than that when $\rho_A = 1$. Therefore, when $0 < \rho_D, \rho_A \le 1$, $P(\Delta_A > 600\mu s) < 0.1754$. □

Theorem 1 indicates that the probability that an inter-ACK time exceeds $600\mu s$ is small for the Ethernet setting. Next we consider the sample median distribution of inter-ACK times, and calculate the probabilities that it is above $600\mu s$. Let $\xi^n_{.5}(\Delta_A)$ denote the sample median of $\{\Delta^A_i\}^n_{i=1}$. Let $\Delta^{(1)}_A, \Delta^{(2)}_A, \ldots, \Delta^{(n)}_A$ be the ordered statistic of $\Delta^A_1, \Delta^A_2, \ldots, \Delta^A_n$ in the ascending order. For simplicity, we use $\xi^n_{.5}(\Delta_A) = \Delta^{(\lfloor(n+1)/2\rfloor)}_A$ regardless n being even or odd. We have the following theorem on the median inter-ACK time; Lemma 4 used in the proof is presented in Appendix.

Theorem 2. ***(Median for Ethernet)*** *For the Ethernet setting, given an i.i.d sequence of samples $\Delta^A_1, \Delta^A_2, \ldots, \Delta^A_n$, when $43 \leq n \leq 100$, we have $P(\xi^n_{.5}(\Delta_A) \leq 600\ \mu s) \approx 1$.*

Proof. Let $u = P(\Delta_A \leq 600\mu s)$.

$$P(\xi^n_{.5}(\Delta_A) \leq 600\mu s) = \sum_{i=\lfloor(n+1)/2\rfloor}^{n} \binom{n}{i} u^i (1-u)^{n-i}$$

Let $g(n,u) = \sum_{i=\lfloor(n+1)/2\rfloor}^{n} \binom{n}{i} u^i (1-u)^{n-i}$. By Lemma 4, $g(n,q)$ is an increasing function of q for $0 \leq q \leq 1$. By Theorem 1, we know $u > 1 - 0.1754 = 0.8246$. Therefore, we have $g(n,u) \geq g(n, 0.8246)$. Hence, $P(\xi^n_{.5}(\Delta_A) \leq 600\mu s) \geq g(n, 0.8246)$. By direct calculation, we have $P(\xi^n_{.5}(\Delta_A \leq 600\mu s) \approx 1$ for $43 \leq n \leq 100$. □

The above theorem states that the probability that the median inter-ACK lies below 600 μs is close to 1 when the number of samples is between 43 and 100. This result is utilized in Section 4.2 to derive a sequential hypothesis test.

3.3 Analysis of 802.11b WLAN

Our analysis utilizes the following knowledge of 802.11b. In 11Mbps 802.11b, transmitting a TCP packet with zero payload takes at least $508\mu s$ [19]. Furthermore, under ideal conditions (i.e., perfect wireless channel and no contention), the receiver waits for a random amount of time uniformly distributed in $[0, 620]\mu s$ before transmitting a packet (see [12] and the reference within). As shown in Fig. 2(b), under ideal conditions, only the access point and the receiver contend for the wireless link (to send data packets and ACKs respectively). The transmission time for a data packet (1500 bytes) and an ACK (40 bytes) is uniformly distributed in $[1570, 2190]\mu s$ and $[508, 1128]\mu s$ respectively.

We next state a theorem on the distribution of the inter-ACK time.

Theorem 3. *For a 802.11b WLAN setting, under perfect wireless channel and without contention from other wireless nodes, we have $P(\Delta_A > 600\mu s) > 0.9269$.*

Proof. After receiving packet P_1, the receiver returns ACK A_1 to the sender. At this point, ACK A_1 and packet P_2 contend for the wireless channel. Let ϕ denote the probability that ACK A_1 obtains the channel and hence transmits earlier than packet P_2. We assume that ϕ can take any value in $[0,1]$ since the contention between ACK A_1 and packet P_2 can be affected by many factors, such as when ACK A_1 reaches the MAC layer, when packet P_2 can be transmitted, etc. If A_1 transmits later than P_2, then A_1 contend with P_3 for the wireless channel. In this case, we assume that A_1 and P_3 are equal likely to obtain the channel since they both can be transmitted immediately.

We first derive the probability that Δ_A is no more than $600\mu s$. In order for $\Delta_A \leq 600\mu s$ to hold, no data packet can be transmitted between ACK A_1 and A_3, since the transmission time of a data packet is at least $1570\mu s$. Therefore, only two sequences of data and ACK transmission are possible: $P_2P_3A_1A_3P_4$ or $P_2P_3P_4A_1A_3$. The probability that the first sequence occurs is: $(1-\phi) \times 1/2 \times 1/2 \times \phi = \phi(1-\phi)/4$. The probability that the second sequence occurs is: $(1-\phi) \times 1/2 \times 1/2 = (1-\phi)/4$. To satisfy $\Delta_A \leq 600\mu s$, we also require the transmission time of A_3 to be less than $600\mu s$. The probability of this condition being satisfied is $(600 - 508)/620 = 92/620$. Therefore,

$$\begin{aligned} P(\Delta_A \leq 600\mu s) &= [\phi(1-\phi)/4 + (1-\phi)/4]92/620 \\ &= \frac{1}{4}(1-\phi^2)\frac{92}{620} < 0.0371. \end{aligned}$$

Hence, $P(\Delta_A > 600\mu s) > 1 - 0.0371 > 0.9269$. □

Theorems 1 and 3 show that, inter-ACK times for 100Mbps Ethernet rarely exceeds $600\mu s$, while for a receiver using 802.11b, at least 92% of time, inter-ACK times are above $600\mu s$ under ideal conditions. Under more realistic conditions (e.g., noisy wireless channel and with contention), inter-ACK times may be even higher. This implies that TCP flows from 100Mbps Ethernet and 802.11b WLAN can be easily distinguished.

3.4 Analysis of 802.11g WLAN

Next we describe analysis results for 802.11g WLAN. For 54Mbps 802.11g, we have (1) transmitting a TCP packet with zero payload takes at least $103\mu s$; (2) under ideal conditions, the receiver waits for a random amount of time uniformly distributed in $[0, 126]\mu s$ before transmitting a packet; (3) under ideal conditions, the transmission time for a data packet (1500 bytes) and an ACK (40 bytes) is uniformly distributed in $[320, 446]\mu s$ and $[103, 230]\mu s$ respectively. We have the following theorem for 802.11g.

Theorem 4. *For a 802.11g WLAN setting, under perfect wireless channel and without contention from other wireless nodes, we have $P(\Delta_A > 600\mu s) > 0.6449$.*

Proof. The proof is similar to that for Theorem 3 and can be found in Appendix. □

Combining Theorems 1 and 4, we note that 100Mbps Ethernet and 54Mbps WLAN are quite distinguishable: For Ethernet, less than 18% of the inter-ACK times exceed $600\mu s$, while for 802.11g, at least 64% the inter-ACK times exceed $600\mu s$.

3.5 Summary of Analysis

In the above, we analyzed inter-ACK times when the receiver uses 100Mbps Ethernet, 802.11b WLAN, or 802.11g WLAN and showed that inter-ACK times of WLAN tend to be larger than those in Ethernet even when WLAN is under ideal condition and Ethernet LAN is fully utilized. From the analysis, we can see that the main reasons that Ethernet can be separated from WLAN are the half duplex nature of wireless channels and the random backoff mechanism in 802.11 CSMA/CA. The insights from the analysis are to be used to construct sequential hypothesis tests in Section 4. Moreover, Theorem 1 and Theorem 2 are used explicitly to construct one of the tests.

```
$n = 0, l_E = l_W = 0.$
do {
  Identify an ACK-pair
  $n = n + 1$
  $p_n = P(\Delta_n^A = \delta_n^A \mid E), q_n = P(\Delta_n^A = \delta_n^A \mid W)$
  $l_E = l_E + \log p_n,\ l_W = l_W + \log q_n$

  **if** $l_W - l_E > \log K$
    Report WLAN, $n = 0, l_E = l_W = 0.$

  **else if** $l_W - l_E < -\log K$
    Report Ethernet, $n = 0, l_E = l_W = 0.$

  **else if** $n = N$
    Report undetermined, $n = 0, l_E = l_W = 0.$
}
```

Figure 3: Sequential hypothesis test with training, $K > 1$ is the threshold, N is the maximum number of ACK-pairs used to make a decision.

4. SEQUENTIAL HYPOTHESIS TESTS

In this section, we develop two sequential hypothesis tests based on analytical results in Section 3. When it is possible to construct training sets to obtain the inter-ACK time distributions for Ethernet and WLAN traffic, we develop a sequential hypothesis test using these distributions. When the inter-ACK time distributions are not available (e.g., for organizations with no wireless networks), we develop a sequential hypothesis test based on Theorems 1 and 2 in Section 3. We refer to these two tests as *sequential hypothesis test with training* and *sequential hypothesis test without training* respectively. These two tests use at most N ACK-pairs to determine whether a host uses Ethernet or WLAN connection. After a decision or N ACK-pairs (i.e., no decision has been made), the tests start over. This is to accommodate the scenario that users may switch between Ethernet and WLAN connections. We now describe the two tests in detail.

4.1 Sequential Hypothesis Test with Training

We first describe the main idea of this test. As shown in Section 3, the inter-ACK time distributions for Ethernet and WLAN traffic differ significantly. When these distributions are known, we can calculate the likelihoods that a host uses Ethernet and WLAN given a sequence of observed inter-ACK times. If the likelihood of using WLAN is much higher than that of using Ethernet, we conclude that the host uses WLAN (and vice versa).

Let $\{\delta_i^A\}_{i=1}^n$ represent a sequence of inter-ACK time observations from a host, and $\{\Delta_i^A\}_{i=1}^n$ represent their corresponding random variables. Let E and W represent respectively the events that a host uses Ethernet and WLAN. Let $L_E = P(\Delta_1^A = \delta_1^A, \Delta_2^A = \delta_2^A, \ldots, \Delta_n^A = \delta_n^A \mid E)$ be the likelihood that this observation sequence is from an Ethernet host. Similarly, let $L_W = P(\Delta_1^A = \delta_1^A, \Delta_2^A = \delta_2^A, \ldots, \Delta_n^A = \delta_n^A \mid W)$ be the likelihood that the observation sequence is from a WLAN host.

Let $p_i = P(\Delta_i^A = \delta_i^A \mid E)$ be the probability that the i-th inter-ACK time has value δ_i^A given that it is from an Ethernet host. Similarly, let $q_i = P(\Delta_i^A = \delta_i^A \mid W)$ be the probability that the i-th inter-ACK time has value δ_i^A given that it is from a WLAN host. Both p_i and q_i are known, obtained from the inter-ACK time distributions for Ethernet

```
$m = n = 0.$
do {
  Identify an ACK-pair
  $n = n + 1$
  $m = m + \mathbb{1}(\delta_n^A \geq 600\mu s)$
  $\hat{p} = \frac{m}{n}$

  **if** $\hat{p} = 1$ **and** $n > -\frac{\log K}{\log \theta}$
    Report WLAN. $m = n = 0.$

  **else if** $n < \frac{m(\log \hat{p} - \log \theta + \log(1-\theta) - \log(1-\hat{p})) - \log K}{\log(1-\theta) - \log(1-\hat{p})}$
    Report WLAN. $m = n = 0.$

  **else if** $n \geq 43$ **and** $\hat{p} \geq 0.5$
    Report WLAN. $m = n = 0.$

  **else if** $n = N$
    Report undetermined. $m = n = 0.$
}
```

Figure 4: Sequential hypothesis test without training, where $\mathbb{1}(\cdot)$ is the indicator function, $K > 1$ is the threshold, N is the maximum number of ACK-pairs used to make a decision.

and WLAN traffic respectively. Assuming that the inter-ACK times are independent and identically distributed, we have

$$L_E = P(\Delta_1^A = \delta_1^A, \Delta_2^A = \delta_2^A, \ldots, \Delta_n^A = \delta_n^A \mid E) = \prod_{i=1}^{n} p_i$$

$$L_W = P(\Delta_1^A = \delta_1^A, \Delta_2^A = \delta_2^A, \ldots, \Delta_n^A = \delta_n^A \mid W) = \prod_{i=1}^{n} q_i$$

Our sequential hypothesis test with training is based on the likelihood ratio of being a WLAN and Ethernet host. This test updates L_W and L_E as each ACK-pair comes in. Let $K > 1$ be a threshold. If after the n-th ACK-pair, the ratio of L_W and L_E is over the threshold, i.e., $L_W/L_E > K$, then the host is classified as a WLAN host. If $L_W/L_E < 1/K$, then the host is classified as an Ethernet host. If neither decision is made after N ACK-pairs, the connection type of this host is classified undetermined. In the implementation, for convenience, we use log-likelihood function $l_w = \log(L_W)$ and $l_E = \log(L_E)$ instead of the likelihood function. A detailed description of this test is presented in Fig. 3. We use $K = 10^6$ and $N = 100$ for online detection of rogue AP in our experimental evaluation (Section 6).

4.2 Sequential Hypothesis Test without Training

This test does not require the inter-ACK time distributions for Ethernet and WLAN hosts. Instead, it leverages the analytical result that the probability that an inter-ACK time exceeds $600\mu s$ is low for Ethernet hosts, while it is much larger for WLAN hosts (see Section 3). In the following, we first construct a likelihood ratio test [17], and then derive a sequential hypothesis test from it.

The likelihood ratio test is as follows. Let p be the probability that an inter-ACK time exceeds $600\mu s$, that is, $p = P(\Delta_A > 600\mu s)$. By Theorem 1, we have $p < \theta = 0.1754$ for Ethernet host. Therefore, we test whether $p < \theta$ is true. If $p < \theta$ is rejected by the inter-ACK time observation sequence, we conclude that this host does not use Ethernet and hence uses WLAN. The likelihood ratio test is based on this reasoning. Consider two hypotheses, H_0 and H_a, representing respectively the null hypothesis that a host uses Ethernet and the alternative hypothesis that the host uses WLAN. For a sequence of inter-ACK time observations $\{\delta_i^A\}_{i=1}^n$, let m be the number of observations that exceed $600\mu s$. Let $K > 1$ be a threshold. Then the likelihood ratio test rejects the null hypothesis H_0 when

$$\lambda = \frac{\sup_{0\le p\le \theta} p^m(1-p)^{n-m}}{\sup_{0\le p\le 1} p^m(1-p)^{n-m}} < \frac{1}{K}$$

In the above, the numerator of λ is the maximum probability of having the observed sequence (which has m inter-ACK times exceeding $600\mu s$) computed over parameters in the null hypothesis (i.e., $0 \le p \le \theta$). The denominator of λ is the maximum probability of having the observed sequence over all possible parameters (i.e., $0 \le p \le 1$). If $\lambda < 1/K$, that is, there are parameter points in the alternative hypothesis for which the observed sample is much more likely than for any parameter points in the null hypothesis, the likelihood ratio test concludes that H_0 should be rejected. In other words, if $\lambda < 1/K$, the likelihood ratio test concludes that the host uses WLAN.

We now derive a sequential hypothesis test from the above likelihood ratio test. Let $\hat{p} = m/n$. It is straightforward to show that $\hat{p}$ is the maximum likelihood estimator of p, i.e., $\sup_{0\le p\le 1} p^m(1-p)^{n-m}$ is achieved when $p = \hat{p}$. When $\hat{p} \le \theta$, we have $\sup_{0\le p\le\theta} p^m(1-p)^{n-m} = \sup_{0\le p\le 1} p^m(1-p)^{n-m}$, and hence $\lambda = 1 > 1/K$. In this case, the null hypothesis H_0 is not rejected. Therefore, we only consider the case where $\theta < \hat{p}$, which can be classified into two cases.

Case 1: $\theta < \hat{p} < 1$**.** In this case, to reject the null hypothesis H_0, we need

$$\frac{\hat{p}^m(1-\hat{p})^{n-m}}{\theta^m(1-\theta)^{n-m}} \quad > \quad K$$

which is equivalent to

$$n < \frac{m(\log\hat{p} - \log\theta + \log(1-\theta) - \log(1-\hat{p})) - \log K}{\log(1-\theta) - \log(1-\hat{p})} \qquad (1)$$

Case 2: $\hat{p} = 1$**.** In this case, to reject the null hypothesis H_0, we need

$$\lambda = \frac{1}{\theta^n} > K$$

which is equivalent to

$$n > -\frac{\log K}{\log\theta} \qquad (2)$$

When $K = 10^6$ and $\theta = 0.1754$, from (2), we have $n \ge 8$. This implies that we need at least 8 ACK-pairs to detect a WLAN host for the above setting.

In addition to the two conditions (1) and (2), we also derive a complementary condition to reject the null hypothesis H_0 directly from Theorem 2. Theorem 2 states that, when the number of inter-ACK observations n is between 43 and 100, we have $P(\xi_{.5}^n(\Delta_A) \le 600\mu s) \approx 1$ for Ethernet hosts. Therefore, an additional condition to reject H_0 is when $43 \le n \le 100$ and $\hat{p} > 0.5$ (because this condition implies that at least half of the inter-ACK observations exceed $600\mu s$, that is, $\xi_{.5}^n(\Delta_A) > 600\mu s$, which contradicts Theorem 2).

We combine the above three conditions to construct a sequential hypothesis test as shown in Fig. 4. This test uses at most N ACK-pairs to reach a decision (reporting that the host uses WLAN or undetermined). We use $N = 100$ in our experiments (Section 6). Note that this sequential hypothesis test only reports WLAN hosts, while the sequential hypothesis test with training reports both WLAN and Ethernet hosts.

5. ONLINE DETECTION SCHEME

In this section, we first describe two components in our online rogue AP detection scheme, i.e., identifying ACK-pairs in real time and obtaining Ethernet and WLAN inter-ACK time distributions (required by the sequential hypothesis test with training). We then describe online detection of WLAN hosts and our policy-based online detection of rogue APs.

5.1 Online Identification of TCP ACK-pairs

We refer to two successive ACKs as an ACK-pair if the inter-arrival time of their corresponding data packets at the monitoring point is less than a threshold T. In our experiments, we set T to $240\mu s$ or $400\mu s$. In practice, to identify ACK-pairs, we also need to consider several practical issues, e.g., packet retransmission and delayed ACK (i.e., a receiver releases an ACK after receiving two packets, or if the delayed-ACK timer is triggered after the arrival of a single packet). We exclude all ACKs whose corresponding data packets have been retransmitted or reordered. If delayed ACK is implemented (inferred using techniques in [13]), we exclude ACKs due to expiration of delayed-ACK timers. This is because, if an ACK is triggered by a delayed-ACK timer, it is not released immediately after a data packet. Therefore, the inter-arrival time of this ACK and its previous ACK does not reflect the characteristics of the access link. In addition, we require that the inter-ACK time of an ACK-pair to be below 200ms for the following reason. Consider three ACKs, the second and third ones being triggered by delayed-ACK timer. If the second ACK is lost, the measurement point will only observe a pair of ACKs (the first and third ACK), which is not a valid ACK-pair (since the third ACK is triggered by delayed-ACK timer). Requiring the inter-ACK time of an ACK-pair to be below 200ms can exclude this pair of ACKs because their inter-arrival time is at least 200ms (it takes at least 100ms for a delayed-ACK timer to go off). Furthermore, to ensure that two ACKs are successive, we require that the difference of their IPIDs to be no more than 1. We also restrict that the ACKs are for relatively large data packets (of size at least 1000 bytes), to be consistent with the assumption of our analysis (in Section 3).

A user may purposely violate the above criteria for ACK-pairs (e.g., by never using TCP, using smaller MTUs or tampering the IPID field) so that the measurement point does not capture any valid ACK-pair from this user. However, all the above cases are easy to detect and can raise an alarm that this user may attempt to hide a rogue AP.

5.2 Obtaining Ethernet and WLAN Inter-ACK Time Distributions

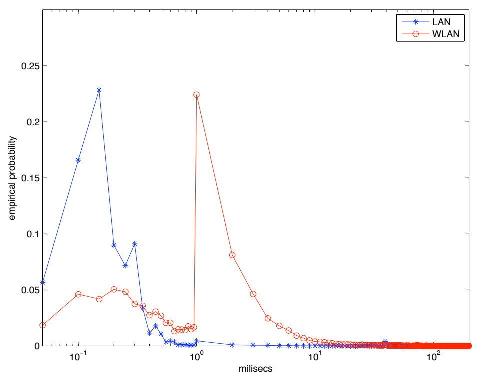

Figure 5: Ethernet and WLAN inter-ACK time distributions obtained from training ($T = 240\mu s$).

To apply the sequential hypothesis test with training, we need to know the inter-ACK time distributions for Ethernet and WLAN hosts beforehand. In general, the inter-ACK time distribution for a certain connection type can be acquired from a training set, which contains measurements for hosts that are *known* to use this type of connection. This requires identifying an IP address block for each interested type of connection. For this specific study, we were able to identify such address blocks for both Ethernet and WLAN based on our knowledge of the campus network. The identified IP block for WLAN consists of a set of IP addresses that are reserved for the campus public WLAN, an 802.11 network providing wireless access to campus users at certain public places such as the libraries, campus eateries, etc. The identified IP block for Ethernet consists of a set of IP addresses for hosts using 100Mbps Ethernet in the Computer Science department. The number of IP addresses in these two blocks is 1177 and 648 respectively.

After identifying an IP address block for a connection type, constructing a training set to obtain the inter-ACK time distribution is straightforward. We collect a group of traces at the measurement point, and then extract TCP flows destined to hosts in the identified IP address blocks to construct training sets for WLAN and Ethernet respectively. The training set for Ethernet is constructed from traces collected between February and April, 2005 (also used in [14]). Since more users are using 802.11g compared to one year ago, we collected a new set of traces on 9/29/2006 to construct the training set for WLAN. From the training sets, we identify ACK-pairs using threshold $T = 240\mu s$ or $400\mu s$. Afterwards, we discretize the inter-ACK times to obtain inter-ACK time distributions: if an inter-ACK time is below 1ms, then the bin size is set to $50\mu s$; otherwise (the inter-ACK time is between 1ms and 200ms), the bin size is set to 1ms. Fig. 5 plots the inter-ACK time distributions for Ethernet and WLAN hosts thus obtained, $T = 240\mu s$. We observe that majority of the inter-ACK times for Ethernet hosts are below $600\mu s$, while majority of the inter-ACK times for WLAN hosts are above $600\mu s$, confirming our analytical results in Section 3.

5.3 Online Detection of WLAN Hosts

We now briefly describe our online monitoring and detection system. Our monitoring equipment is a commodity PC with three Intel Xeon Y CPU 2.80GHz (cache size 512KB), 2Gbytes memory, and SCSI hard disks. This equipment is connected via an optical splitter to the access link connecting the campus network to the commercial network. All packets traversing this link are passed on to the monitoring equipment. A packet capture card (called a DAG card [6]) on the monitoring equipment captures the headers of all packets. The captured packet headers may be filtered (so that only packets from IP addresses that we intend to monitor are left) and then copied to the disk (for offline analysis) or to our online detection engine (for online detection) along with accurate time stamps. We next focus on the online detection engine.

The online detection engine maintains a data-packet reassembly queue for each active TCP flow (i.e., flows that are not terminated and have data transmission during the last minute) in memory. Each data-packet reassembly queue stores information about the data packets that have not yet been acknowledged by the receiver. The information of a data packet will be removed from the queue once its corresponding ACK arrives (since this information is never needed again). For each incoming ACK, the online detection engine finds its corresponding data-packet reassembly queue (using a Hash function for quick lookup) and matches it with the data packets to identify ACK-pairs. Since only unacknowledged data packets are stored in a data-packet reassembly queue, the maximum number of packets in the queue is bounded by the maximum TCP window size. In reality, our experiments show that the number of packets in the queue is much smaller.

Since our sequential hypothesis tests determine a host's connection type using ACK-pairs from all TCP flows destined to this host, our online detection engine also maintains a single data structure for each IP address that we intent to monitor. When observing a valid data packet or ACK-pair for an IP address, the online detection engine updates this data structure. Our sequential hypothesis tests execute whenever a new ACK-pair is identified to detect whether the corresponding host uses WLAN.

5.4 Policy-based Rogue AP Detection

We now describe the last part of our online detection scheme: policy-based rogue AP detection. Once our monitoring system detects a WLAN host, it looks up the host from an authorization list (i.e., a list of hosts authorized to use WLAN) and raises an alarm if this host is not in the list. Therefore, our scheme can easily detect rogues installed by hosts not authorized to use WLAN. We next discuss the case that rouges are installed by hosts authorized to use WLAN (This is not common since rogues are typically installed by users with no wireless connections for convenience or higher productivity. Nonetheless, a malicious user may attempt to do so.).

We discuss the above case in two scenarios. In the first scenario, the IP address blocks for Ethernet and WLAN connections do not overlap. Then a host will have different IP addresses for its Ethernet and WLAN connections. In this scenario, if a host authorized to use both Ethernet and WLAN installs a rogue AP on its Ethernet connection, the host obtains an IP address in the Ethernet block and the associated rogue AP will be easily detected by our scheme. If a host A uses its authorized WLAN connection to connect to the Internet and sets up another wireless card as a rogue AP for host B (as described in Section 6.4), then packets from B will have the IP address of A, which is an authorized WLAN address. Therefore, our scheme does not detect this type of rogue directly. However, since host B connects to the Internet through two wireless hops, the traffic characteristics of B will differ from those through a single wireless hop and those through Ethernet, and hence can be easily differentiated from others. An accurate detection scheme for this type of rogue is left as future work. In the second scenario, the IP address blocks for Ethernet and WLAN connections overlap. Then a host may maintain the same IP address for both Ethernet and WLAN connections. Similar to the first scenario, we can detect rogue APs that provide hosts Internet connection using two wireless hops through traffic analysis. However, a host authorized to use WLAN may also set up a rogue AP on its Ethernet connection for itself to connect to the Internet. Our policy-based scheme does not detect this type of rogue and analyzing traffic characteristics does not help (since this host only use a single wireless hop). However, in this case, we can use our scheme combined with RF-sensing schemes so that only hosts in the authorization list need to be closely monitored by RF sensing, and thus may greatly reduce the number of hosts to be monitored. The above discussions imply that, to achieve tighter security, it is better to use separate IP blocks for Ethernet and WLAN connections.

6. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our rogue AP detection scheme through extensive experiments on a campus network. The measurement point is located at the university gateway router, monitoring the traffic coming into and going out of the campus network. Since it is straightforward to check whether a WLAN host is in the authorization list, we mainly evaluate our scheme to determine a host's connection type in real time. Our detection system monitors all IP addresses in the campus public WLAN and the Computer Science department (totally 3717 addresses) since our test sets are from these two IP spaces (see Section 6.1). We sample the CPU and memory usages at the measurement PC every 30 seconds. The maximum CPU and memory usages are 9.1% and 53.4% respectively (without optimizating our implementation), indicating that the measurement task is within the capability of the measurement PC.

In the following, we investigate the accuracy and promptness of our scheme. For this purpose, we construct two test sets (Section 6.1) and apply our sequential hypothesis tests to these test sets (Sections 6.2 and 6.3). We then demonstrate that our scheme is effective to detect other types of rogues, in particular, wireless networks behind NAT (Section 6.4). Last, we show that our scheme can quickly detect connection switchings (Section 6.5) and is robust to high CPU, disk or network utilization at end hosts (Section 6.6).

6.1 Constructing Test Sets

To validate that our scheme can detect WLAN hosts while does not misclassify Ethernet hosts, we construct two test sets. The first test set consists of the IP address block (of 1177 addresses) reserved for the campus public WLAN (see Section 5.2). The second test set consists of the IP addresses of a subset of Dell desktops that use Ethernet in the Computer Science building. This test set contains 258 desktops, each with documented IP address, MAC address, operating system, and location information for ease of validation. Among these desktops, 35% of them use different versions of Windows operating system (e.g., Windows 2000, Windows ME, Windows XP, etc.); the rest use different variants of Linux and Unix operating systems (e.g., RedHat, Solaris, CentOS, Fedora Core, etc.). These hosts are three hops away from the univerisity gateway router (and the monitoring point).

6.2 Performance of Sequential Hypothesis Test with Training

We now investigate the performance of our sequential hypothesis test with training. The Ethernet and WLAN inter-ACK time distributions required by this test are obtained as described in Section 5.2. This test makes a decision (detect WLAN, Ethernet or undetermined) with at most N ACK-pairs. We use $N = 100$ for all the experiments. A decision of WLAN or Ethernet is referred to as a *detection*.

We evaluate the performance of this sequential hypothesis test in both offline and online manners. In offline evaluation, we first collect measurements and then apply the sequential hypothesis test to the collected trace. In online evaluation, we run the sequential hypothesis test online while capturing the data at the measurement point. The offline evaluation, although does not represent the normal operation mode of our sequential hypothesis tests, allows us to investigate the impact of various parameters. The online evaluation investigates the performance of our scheme in its normal operation mode. We now describe the results of offline and online evaluation respectively.

6.2.1 *Offline Evaluation*

In offline evaluation, we collect measurements on three consecutive days, from 10/18/2006 to 10/20/2006. The trace on each day lasts for 6 to 7 hours. The threshold to identify ACK-pairs, T, is set to $240\mu s$ or $400\mu s$. The threshold to decide a host's connection type, K, is set to $10^4, 10^5$ or 10^6. We next describe the results for the trace collected on 10/20/2006; the results for the other two days are similar.

Table 1 presents the detection results for the campus public WLAN. We observe that the detection results are similar under different values of T and K, indicating that our scheme is insensitive to the choice of parameters. For all values of T and K, the detection results are extremely accurate (the correct detection ratio is between 99.38% to 99.61%). For instance, when $T = 240\mu s$, we observe $65,138$ ACK-pairs from 426 IP addresses in the test set. When $K = 10^6$, 356 IP addresses have at least one detection. The total number of detections is $8,969$ and the correct detection ratio is 99.61%. On average, it takes a few ACK-pairs (corresponding to 250 to 347 data packets, less than 521 Kbytes of data) to make a detection. The relatively large number of data packets for a detection (compared to that of Ethernet) is due to low ACK-pair ratio (i.e., the number of ACK-pairs divided by the total number of packets) in WLAN traffic. This low ACK-pair ratio is because the inter-ACK times in WLAN tend to be large (compared to those in Ethernet), leading to large inter-arrival times between newly triggered data packets due to TCP's self-clocking. When the inter-arrival time of the data packets is larger than the threshold T, the corresponding ACKs are not qualified as an ACK-pair. Even so, the median detection time is no more than 13 seconds. The distribution of the detection time when

Table 1: Experimental results on WLANs using sequential hypothesis test with training (10/20/2006).

	T=240μs			T=400μs		
	$K = 10^4$	$K = 10^5$	$K = 10^6$	$K = 10^4$	$K = 10^5$	$K = 10^6$
Avg. # of ACK-pairs for a detection	5	6	7	5	6	7
Avg. # of data pkts for a detection	250	288	347	204	235	283
Median detection time (sec)	8	10	13	6	8	11
Number of detections	$12,607$	$10,882$	$8,969$	$15,724$	$13,567$	$11,169$
Correct detection ratio	99.43%	99.59%	99.61%	99.38%	99.53%	99.61%
ACK-pair ratio	2%			2%		

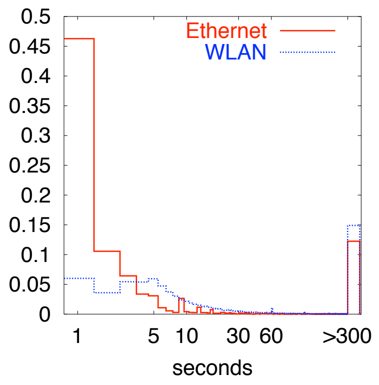

Figure 6: Detection time distribution for the trace collected on 10/20/2006 ($T = 240\mu s$, $K = 10^6, N = 100$).

$K = 10^6$ is shown in Fig. 6. Those long detection times (over 5 minutes) might be caused by users' change of activities (e.g., a user stops using the computer to think or talk and then resume using it). Finally, around 84% of ACK-pairs are generated by web traffic, indicating that our scheme is effective even for short flows.

Table 2 presents the detection results for the Ethernet test set. Again, we observe that our scheme is insensitive to the choice of parameters. The detection is highly accurate. For instance, when $T = 240\mu s$, we discover $55,288$ ACK-pairs from 68 hosts in the test set. When $K = 10^6$, 53 hosts have at least one detection. The total number of detections is $3,363$, with only a single detection as WLAN, indicating 99.94% of correct detection ratio. (The host with the single incorrect detection is detected 17 times as Ethernet and one time as WLAN.) On average, it takes a few ACK-pairs (corresponding to 87 to 124 data packets, no more than 188 Kbytes of data) to make a detection. The median detection time is around 1 second, much shorter than that in WLAN (due to higher ACK-pair ratios in Ethernet). Fig. 6 plots the distribution of the detection time when $K = 10^6$. Again, those long detection times might be caused by users' change of activities. Last, around 89% of ACK-pairs are generated by web traffic.

6.2.2 *Online Evaluation*

In online evaluation, we run our detection scheme online on three consecutive days, from 10/25/2006 to 10/27/2006, lasting for 6 to 7 hours on each day. We set $T = 240\mu s, K = 10^6$, representing a conservative selection of parameters.

Table 3 presents the detection results for both test sets. We observe consistent results as those in offline evaluation. That is, the detection is highly accurate and prompt. The average numbers of ACK-pairs and data packets required for a detection are consistent with those in the offline evaluation. The above demonstrates the efficiency of our online detection algorithm.

6.3 Performance of Sequential Hypothesis Test without Training

We now examine the performance of our sequential hypothesis test without training. Recall that this test does not require training sets. It takes at most N ACK-pairs to make a decision (i.e., detecting WLAN or undetermined). Note that, different from the sequential hypothesis test with training, it does not detect Ethernet hosts.

We apply this sequential hypothesis test to traces collected between 10/18/2006 and 10/20/2006 using $T = 240\mu s$, $K = 10^6$, and $N = 100$. For the Ethernet test set, this sequential hypothesis test detects no WLAN host for all the traces, indicating that it has no false positives. Note that although this test is derived using analytical results in Section 3, in a setting where the receiver is one hop away from the router, our experimental results indicate that this test is accurate in more relaxed settings (the Ethernet hosts in the Computer Science building are three hops away from the gateway router). This is not surprising since our test is based on an extremely conservative analysis (assuming that the single Ethernet link is full utilized). For the WLAN test set, of all the hosts with at least one ACK-pair, this sequential hypothesis test detects 60% to 76% of them as WLAN hosts. Table 4 presents the experimental results for the WLAN test set. In general, this test requires more ACK-pairs and longer time to make a detection than the test with training.

6.4 Detection of Wireless Networks behind NAT

We now demonstrate that our scheme is equally applicable to detect other types of rouges, in particular, wireless networks behind a NAT box. Since all traffic going through a NAT box have the same MAC address (i.e., the MAC address of the NAT box), schemes using MAC address fail to detect this type of rogues while our scheme successfully detects this type of rogues, as we shall see. We look at NAT boxes in two settings, one configured by ourselves and the other used in the Computer Science department.

6.4.1 *Self-configured NAT*

We configure a Linux box A as a NAT box. Host A has two network interfaces, an Ethernet card and a "ZCOMAX AirRunner/XI-300" 802.11b wireless card. The Ethernet

Table 2: Experimental results on Ethernet using sequential hypothesis test with training (10/20/2006)**.**

	T=240μs			T=400μs		
	$K = 10^4$	$K = 10^5$	$K = 10^6$	$K = 10^4$	$K = 10^5$	$K = 10^6$
Avg. # of ACK-pairs for a detection	11	13	16	13	16	19
Avg. # of data pkts for a detection	87	106	124	73	89	106
Median detection time (sec)	0.6	1.0	1.2	0.3	0.6	0.9
Number of detections	4,896	3,990	3,363	5,860	4,747	4,002
Correct detection ratio	99.88%	100.00%	99.97%	99.61%	99.79%	99.78%
ACK-pair ratio	13%			17%		

Table 3: Online detection results using sequential hypothesis test with training (10/25/2006 - 10/27/2006)**.**

	10/25/2006		10/26/2006		10/27/2006	
	WLAN	Ethernet	WLAN	Ethernet	WLAN	Ethernet
Avg. # of ACK-pairs for a detection	7	16	8	21	7	16
Avg. # of data pkts for a detection	310	145	351	153	336	135
Median detection time (sec)	9.7	1.2	15.0	0.1	11.4	1.2
Number of detections	23,266	5,798	15,977	15,654	10,628	2,948
Correct detection ratio	99.58%	99.93%	98.44%	99.92%	99.72%	99.76%
ACK-pair ratio	2%	11%	2%	13%	2%	12%

interface connects directly to the Internet. The wireless card is configured to the master mode using Host AP [5] so that it acts as an AP. We then set up two laptops B and C to access the Internet through the wireless card of A. When host B or C accesses the Internet, its packets reach host A. Host A then translates the addresses of the packets and forwards the packets to the Internet through its Ethernet card.

We conduct an experiment on 10/26/2006. The experiment lasts for about two minutes. We observed 163 ACK-pairs. Among them, 92% of the ACK-pairs are from web traffic via port 80. The remaining ACK-pairs are from port 1935, which is used by Macromedia Flash Communication Server MX for the RTMP (Real-Time Messaging Protocol). The sequential hypothesis test with training makes 37 online detections, all as WLAN host. On average, one detection is made in every 4 ACK-pairs. The above demonstrates that our test can effectively detect wireless networks behind NAT boxes.

6.4.2 NATs in the Computer Science Department

Two NAT boxes in the Computer Science department provide a free local network to users in the department. A host may use either Ethernet or WLAN to connect to a NAT box. All traffic through a NAT box will have the IP address of the NAT box. We monitor the IP addresses of these two NAT boxes. Our offline detection (from 10/18/2006 to 10/20/2006) and online detection (from 10/25/2006 to 10/27/2006) both indicate a mixture of WLAN and Ethernet connections. The ACK-pair ratios are higher than that of WLAN and lower than that of Ethernet hosts. The above results are consistent with the setting that these two NAT boxes provide both WLAN and Ethernet connections.

6.5 Detection of Connection Switchings

Next we explore the scenario where an end host may switch between connection types and examine whether our detection scheme can accurately report the connection switchings. We use an IBM laptop with both 100Mbps Ethernet and 54Mbps 802.11g WLAN connections. This laptop uses a web crawler to download the first 200 web files from cnn.com (8.3 Mbytes of data) using Ethernet, and then switch to WLAN to download the first 200 web files from nytimes.com (6.5 Mbytes of data). This process is repeated for three times. We run the sequential hypothesis test with training using $T = 240\mu s, K = 10^6$ and $N = 100$. Our test makes 284 detections, 283 correct and one incorrect. The correct detection ratio is 99.65%. This demonstrates that our scheme is effective in detecting connection switchings. If a host switches between using Ethernet and WLAN provided by a rogue AP, our scheme can effectively detect its rogue AP.

6.6 Detection under High CPU, Disk or Network Utilization

We now investigate whether the performance of our scheme will be affected when an end host has very high CPU, disk or network utilizations. For this purpose, we stress either the CPU, disk or network connection of an end host, while downloading the first 200 web files from cnn.com using a web crawler at the host. For each scenario, we conduct experiments for both Ethernet and WLAN connections and detect the connection type using sequential hypothesis test with training. All experiments are conducted on an IBM laptop.

We stress the CPU (utilization reaching 100%) by running an infinite loop. For the Ethernet connection, we observe 1077 ACK-pairs and 53 detections. For the WLAN connection, we observe 921 ACK-pairs and 123 detections. All the detections are correct. We stress the hard disk by running a virus scanning program that scans the disk. For the Ethernet connection, we observe 1158 ACK-pairs and 57 detections. For the WLAN connection, we observe 872 ACK-pairs and 84 detections. Again, all the detections are correct.

To stress the network connection, we conduct two sets of

Table 4: Experimental results on WLANs using sequential hypothesis test without training.

Date	10/18/2006	10/19/2006	10/20/2006
Detection ratio	68%	76%	60%
Avg. # of ACK-pairs for a detection	22	21	19
Avg. # of data pkts for a detection	997	858	903
Median detection time (sec)	105	59	52
Number of detections	3,259	6,539	2,722

experiments, one stressing the downlink direction by downloading a large file from the local network; one stressing the uplink direction by uploading a large file to the local network. Note that this local downloading and uploading traffic is not captured at the monitoring point and hence does not generate ACK-pairs. For the downloading case, we observe 848 ACK-pairs and 42 detections for the Ethernet connection; 660 ACK-pairs and 72 detections for the WLAN connection. For the uploading case, we observe 438 ACK-pairs and 21 detections for the Ethernet connection; 487 ACK-pairs and 46 detections for the WLAN connection. All the detections are correct. We observe that while doing downloading or uploading, the number of ACK-pairs is significantly smaller than that when stressing CPU or disk. This is due to cross traffic generated by the downloading or uploading activities. We also observe that the number of ACK-pairs is less in the uploading case than that in the downloading case. This is because the uploading data packets may be inserted between ACKs and lead to less ACK-pairs.

In summary, the above results indicate that our detection scheme remains effective for hosts with high CPU, hard disk or network utilizations.

7. CONCLUSION AND FUTURE WORK

In this paper, we have proposed an online scheme to detect rogue access points, based on real time passive measurements collected at a gateway router. This scheme utilizes the CSMA/CA MAC protocol of IEEE 802.11 and the half duplex nature of wireless channels to differentiate Ethernet and WLAN TCP traffic. Central to our approach are two sequential hypothesis tests that determine a host's connection type in real time using TCP ACK-pair techniques that we developed earlier [14]. Extensive experiments in various scenarios and over hosts with various operating systems demonstrate the superb performance of our approach: the sequential hypothesis test with training is prompt (the detection time is mostly within 10 seconds) and extremely accurate (very low false positive and false negative ratios are obtained). The sequential hypothesis test without training detects 60%-76% of the WLAN hosts without false positives. Furthermore, our scheme can detect connection switchings and wireless networks behind a NAT box. Last, our scheme remains effective for hosts with high CPU, hard disk or network utilizations.

As future work, we are exploring in two directions: (1) optimize the implementation of the online detection algorithm; (2) implement the algorithm in hardware.

8. REFERENCES

[1] *AirDefense, Wireless LAN Security.* http://airdefense.net.

[2] *AirMagnet.* http://www.airmagnet.com.

[3] *AirWave, AirWave Management Platform.* http://airwave.com.

[4] *Cisco Wireless LAN Solution Engine (WLSE).* http://www.cisco.com/en/US/products/sw/cscowork /ps3915/.

[5] *Host AP.* http://hostap.epitest.fi.

[6] http://www.endace.com.

[7] Microsoft Windows 2000 TCP/IP implementation details, http://www.microsoft.com/technet/itsolutions /network/deploy/depovg/tcpip2k .mspx.

[8] *NetStumbler.* http://www.netstumbler.com.

[9] *White Paper, Rogue Access Point Detection: Automatically Detect and Manage Wireless Threats to your Network.* http://www.proxim.com.

[10] A. Adya, V. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in ieee 802.11 infrastructure networks. In *ACM Mobicom*, September 2004.

[11] Anonymized. Technical report, 2004.

[12] Anonymized. March 2005.

[13] Anonymized. Technical report, 2005.

[14] Anonymized. 2006.

[15] P. Bahl, R. Chandra, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Enhancing the security of corporate Wi-Fi networks using DAIR. In *ACM MobiSys*, 2006.

[16] R. Beyah, S. Kangude, G. Yu, B. Strickland, and J. Copeland. Rogue access point detection using temporal traffic characteristics. In *GLOBECOM*, Dec 2004.

[17] G. Casella and R. L. Berger. *Statistical Inference.* Duxbury Thomson Learning, 2002.

[18] L. Cheng and I. Marsic. Fuzzy reasoning for wireless awareness. *International Journal of Wireless Information Networks*, 8(1), 2001.

[19] S. Garg, M. Kappes, and A. S. Krishnakumar. On the effect of contention-window sizes in IEEE 802.11b networks. Technical Report ALR-2002-024, Avaya Labs Research, 2002.

[20] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan. Fast Portscan Detection Using Sequential Hypothesis Testing. In *IEEE Symposium on Security and Privacy 2004*, Oakland, CA, May 2004.

[21] Packet trace analysis. http://ipmon.sprintlabs.com/packstat/packetoverview.php.

[22] P. Sarolahti and A. Kuznetsov. Congestion control in linux TCP. In *Proc. of USENIX'02*, June 2002.

[23] K. Thompson, G. Miller, and R. Wilder. Wide-area

Internet traffic patterns and characteristics. *IEEE Network*, 11(6):10–23, Nov./Dec. 1997.

[24] A. Wald. *Sequential Analysis*. J. Wiley & Sons, 1947.

APPENDIX

Proof of Lemma 1:

PROOF. When $\rho_D = 1$, the probability of queue Q_D being empty is 0. Hence, all the cross traffic arriving between packet P_1 and P_3 contribute to the inter-departure time of P_1 and P_3 at queue Q_D. By assumption, the transmission time of each cross traffic packet is 1 time unit at queue Q_D. Therefore, each cross traffic packet arriving between P_1 and P_3 increases the value of I_D by 1 time unit. One component of I_D is the transmission time of packet P_1 and P_2, which is $2 \times 120\mu s$ or 8 time units. The other component of I_D is the transmission time of cross traffic packets between P_1 and P_3, denoted as Z. Then $Z = I_D - 8$. Since the transmission time of packet P_1 and P_2 is 8 time units, on average, the interval between P_1 and P_3 is 8 time units. By the assumption that queue Q_D is a $M/D/1$ queue, $Z = I_D - 8$ follows a Poisson distribution with the parameter of 8 time units. □

Proof of Lemma 2:

PROOF. Since $\rho_A = 1$, the probability of queue Q_A (in the direction of ACKs at the router) being empty is 0. Hence all of the cross traffic arriving between ACK A_1 and A_3 contribute to the inter-departure time of ACK A_1 and A_3 at the router. Since we ignore the transmission time of ACK A_1 and no other traffic is between the router and the receiver, the spacing between ACK A_1 and A_3 is the same as the inter-departure time of packet P_1 and P_3 at queue Q_D, i.e., I_D. Therefore, the conditional distribution of I_A given $I_D = x$ follows a Poisson distribution with the parameter of x time units. □

Proof of Lemma 4:

Let $g(n,q) = \sum_{i=\lfloor (n+1)/2 \rfloor}^{n} \binom{n}{i} q^i (1-q)^{n-i}$, where $0 \le q \le 1$. The following Lemma describes the monotonicity of the function $g(n,q)$ with respect to q.

LEMMA 4. *$g(n,\ q) = \sum_{i=\lfloor (n+1)/2 \rfloor}^{n} \binom{n}{i} q^i (1-q)^{n-i}$ is an increasing function of q, where $0 \le q \le 1$.*

PROOF.

$$\begin{aligned}
\frac{\partial g(n,q)}{\partial q} &= \sum_{i=\lfloor (n+1)/2 \rfloor}^{n} \frac{n!}{i!(n-i)!} i q^{i-1} (1-q)^{n-i} \\
&\quad - \sum_{i=\lfloor (n+1)/2 \rfloor}^{n-1} \frac{n!}{i!(n-i)!} (n-i) q^{i} (1-q)^{n-i-1} \\
&= \sum_{i=\lfloor (n+1)/2 \rfloor}^{n} \frac{n!}{(i-1)!(n-i)!} q^{i-1} (1-q)^{n-i} \\
&\quad - \sum_{i=\lfloor (n+1)/2 \rfloor}^{n-1} \frac{n!}{i!(n-i-1)!} q^{i} (1-q)^{n-i-1} \\
&= \sum_{j=\lfloor (n+1)/2 \rfloor - 1}^{n-1} \frac{n!}{j!(n-j-1)!} q^{j} (1-q)^{n-j-1} \\
&\quad - \sum_{i=\lfloor (n+1)/2 \rfloor}^{n-1} \frac{n!}{i!(n-i-1)!} q^{i} (1-q)^{n-i-1} \\
&= \frac{n! q^{\lfloor (n+1)/2 \rfloor - 1} (1-q)^{n - \lfloor (n+1)/2 \rfloor}}{(\lfloor (n+1)/2 \rfloor - 1)!(n - \lfloor (n+1)/2 \rfloor)!} \\
&\ge 0
\end{aligned}$$

Hence $g(n,q)$ is a increasing function with respect to q for $0 \le q \le 1$. □

Proof of Theorem 4:

PROOF. Similar to the proof of Theorem 3, Let ϕ denote the probability that ACK A_1 obtains the channel and hence transmits earlier than packet P_2.

We first derive the probability that Δ_A is no more than $600\mu s$. In order for $\Delta_A \leq 600\mu s$ to hold, there can be at most one data packet transmitted between ACK A_1 and A_3, since the minimum transmission time of two data packets and one ACK packet exceeds $600\mu s$. This leads to four possible sequences of data and ACK transmission: $P_2P_3A_1A_3P_4$, $P_2P_3P_4A_1A_3$, $P_2A_1P_3A_3P_4$, and $P_2P_3A_1P_4A_3$. Their corresponding probabilities are $(1-\phi)\times 1/2 \times 1/2 \times \phi = \phi(1-\phi)/4$, $(1-\phi) \times 1/2 \times 1/2 = (1-\phi)/4$, $(1-\phi) \times 1/2 \times 1 \times \phi = \phi(1-\phi)/2$, and $(1-\phi) \times 1/2 \times 1/2 \times 1/2 = (1-\phi)/8$ respectively. For the first two sequences, we have $\Delta_A \leq 600\mu s$. For the third sequence, to satisfy $\Delta_A \leq 600\mu s$, we need the total transmission time of P_3 and A_3 to be below $600\mu s$. Similarly, for the fourth sequence, to satisfy $\Delta_A \leq 600\mu s$, we need the total transmission time of P_4 and A_3 to be below $600\mu s$. Let X denote the transmission of a data packet. Let Y denote the transmission of an ACK. Let $\alpha = P(X + Y \leq 600\mu s)$. Then

$$\alpha = 1 - 76 \times 76/((446 - 320) \times (230 - 103)) = 0.6390.$$

Let $\Delta^D_{i,i+1}$ represent the inter-arrival time of data packet P_i and P_{i+1} at the access point. Then

$$\begin{aligned} & P(\Delta_A \leq 600\mu s \mid \Delta^D_{i,i+1} < 320\mu s) \\ = \; & \phi(1-\phi)/4 + (1-\phi)/4 + \alpha\phi(1-\phi)/2 + \alpha(1-\phi)/8 \\ = \; & 1/64008(-36454\phi^2 + 15339\phi + 21115) \leq 0.3551. \end{aligned}$$

Therefore, $P(\Delta_A > 600\mu s \mid \Delta^D_{i,i+1} \leq 320\mu s) > 1 - 0.3551 = 0.6449$.

When $\Delta^D_{i,i+1} > 320\mu s$, at the time packet P_2 is ready to be transmitted, packet P_1 has already been transmitted. In this case, A_1 has better chance to grab the wireless channel. Therefore Δ_A is more likely to be over $600\mu s$ in this case than the case $\Delta^D_{i,i+1} \leq 320\mu s$. Hence, we have $P(\Delta_A > 600\mu s \mid \Delta^D_{i,i+1} > 320\mu s) \geq P(\Delta_A > 600\mu s \mid \Delta^D_{i,i+1} \leq 320\mu s) > 0.6449$.

In summary, we have $P(\Delta_A > 600\mu s) > 0.6449$. $\square$