Identifying Unique Devices through Wireless Fingerprinting

Desmond C. C. Loh[1] Chia Yuan Cho[2] Chung Pheng Tan[2] Ri Seng Lee[2]

[1]National University of Singapore
21 Lower Kent Ridge Road
Singapore 119077
desmondloh@ncs.com.sg

[2]DSO National Laboratories
20 Science Park Drive
Singapore 118230
{cchiayua, tchungph, lriseng}@dso.org.sg

ABSTRACT

We propose a new fingerprinting technique that differentiates between unique devices over a Wireless Local Area Network (WLAN) simply through the timing analysis of 802.11 probe request frames. Our technique can be applied to spoof detection, network reconnaissance, and implementation of access control against masquerading attacks. Experimental results indicate that our technique is consistent and accurate in differentiating between unique devices. In contrast with existing wireless fingerprinting techniques, our technique is passive, non-invasive and does not require the co-operation of fingerprintee hosts.

Categories and Subject Descriptors

C.2.0 [**Computer – Communication Networks**]: General – *security and protection.*

C.2.1 [**Computer – Communication Networks**]: Network Architecture and Design – *wireless communication.*

General Terms

Measurement, Experimentation, Security.

Keywords

Wireless fingerprinting, spoof detection.

1. INTRODUCTION

Wireless LANs are much more difficult to secure compared to wired LANs, mainly due to that fact that wireless signals and network access may extend well beyond the physical boundaries and control of an organization. This inherent characteristic presents numerous security challenges – information may be sniffed promiscuously off the air; network services may be accessed without authorization; identity attacks, such as rogue access points, MAC address spoofing, ARP poisoning and session hijacking attacks are all made easier as a result.

The initial 802.11 security scheme that attempted to deal with these problems was the Wired Equivalent Privacy (WEP). In spite of having mechanisms to provide authentication, confidentiality and data integrity, WEP was found to be insecure and trivially cracked after an attacker has collected enough frames with the same Initialization Vector [23]. By actively speeding up the collection of frames, the latest WEP attack has been able to achieve a breaking of WEP in under a minute [24]. WEP is increasingly being replaced by the Wi-Fi Protected Access (WPA) scheme, which has hardened WLANs from the security vulnerabilities of WEP. However, to retain backward compatibility, WPA has not completely resolved some security issues – while 802.11 data frames are encrypted and resistant to spoofing and replay attacks, control and management frames remain unencrypted and depend mainly on MAC address for identity resolution. Because control and management frames can be spoofed and forged even with WPA enabled, wireless LANs remain susceptible to identity attacks and denial of service attacks.

In recent work, fingerprinting techniques have emerged as possible alternatives to counter the problem of identity in wireless LANs. Essentially, fingerprinting is a process by which a machine, driver or the software the machine is running can be uniquely identified due to its externally observable characteristics. For example, [3, 4, 25] made use of differences in physical properties of radio signals to identify unique radio-frequency based devices. Although the technique showed remarkable results, it required expensive equipment and cannot be easily deployed across WLANs. Other fingerprinting techniques that require just standard computing equipment have been proposed in [5, 6, 7]. We provide a more in-depth discussion of these techniques in Section 2.

Contributions

Our main contribution in this paper is to propose a new wireless fingerprinting technique that differentiates between unique devices through timing analysis of 802.11 probe request frames. More specifically, a unique device is defined as a unique combination of the following tuple: {Machine, Wireless NIC Driver, Operating System (OS)}, and our fingerprinting technique distinguishes between different combinations in the tuple. In contrast to existing techniques, our approach is based on the timing analysis of 802.11 probe request frames, which is passive and noninvasive.

WiSec'08, March 31–April 2, 2008, Alexandria, Virginia, USA.
Copyright 2008 ACM 978-1-59593-814-5/08/03...$5.00.

The remaining of this paper is organized as follows. Section 2 provides background and related work associated with network fingerprinting techniques. Section 3 details the design and implementation of our fingerprinting technique. In Section 4, we present the experimental results and analysis of the proposed solution on a controlled network, a public hotspot and data traces collected during a major conference. Section 5 briefly outlines the applications of our device fingerprinting, with its limitations discussed in Section 6. Section 7 concludes the paper, with directions for future work.

2. RELATED WORK

2.1 Clock Skew Deviations as Fingerprints

Minute deviations between the clock oscillators of different machines could result in clock skews [8] observable from the network, which was found to be suitable as a basis for fingerprinting wired physical machines as described in [7]. Clock skews have also been found to be relatively stable and constant [9].

Given the same time interval, the clock skew is observed through the measurement and comparison of relative notions of time between two different machines. The fingerprinting approach as proposed in [7] was based on measuring the approximate clock skew between a remote (fingerprintee) machine and a local (fingerprinter) machine, using packets that provide timestamps from the fingerprintee. In the TCP/IP protocol suite, such packets are found in the timestamp option field of the TCP header; timestamps may alternatively be obtained by sending ICMP timestamp requests to the fingerprintee. Thus for two consecutive packets received at the fingerprinter, the clock skew can be approximately measured by comparing the inter-packet interval as found in the packet timestamps, with the inter-packet arrival interval based on the local timestamps at the fingerprinter. The clock skew then becomes the fingerprint that uniquely identifies the remote fingerprintee machine.

The main advantage of this technique is that it relies on the clock skews between machines, which are relatively stable and do not deviate much over time. However, this technique was designed for wired networks and it actually requires some co-operation from the fingerprintee machine – both the TCP timestamp option and response to ICMP timestamp requests can be disabled and even on Windows machines, TCP timestamps are not enabled by default.

We propose a fingerprinting technique with the same basic concept as the clock skew fingerprinting approach. In contrast, our technique is meant to work on wireless LANs. Further, our technique does not require co-operation from the fingerprintee because our approach *infers* the fingerpintee's timestamp information without any dependency on timestamp values explicitly given by the fingerprintee.

2.2 Identifying MAC Spoofing by Detecting Sequence Number Anomalies

Techniques to identify wireless client stations with masqueraded MAC addresses based on 802.11 sequence numbers were presented in [10, 28]. The basis for these techniques is that the 12-bit sequence number field in IEEE 802.11 frames increments by one for each frame generated. A wrap around will occur after 4095, the maximum frame number. Hence, by keeping track of MAC addresses to its corresponding sequence number at any point of time, it becomes possible to approximately detect an illegitimate client station with a spoofed MAC. But frames that are lost or retransmitted will affect the tracking of 802.11 frame sequence numbers.

Although the approach works well for the temporal identification of unique wireless client stations, it requires the constant tracking of sequence number growth for all clients in the network. It is also susceptible to sequence number spoofing attacks, and given the small 12-bit sequence number space (for IEEE 802.11), the probability of sequence number collision increases with the number of wireless client stations in the network.

2.3 Location Fingerprinting through RSSI Levels and Physical Layer Link Signatures

Indirect methods to detect spoofed MAC addresses were introduced in [22, 26, 27] by fingerprinting the physical location of wireless clients. The technique in [22] fingerprints the location of each client by measuring the received signal strength (RSSI) of client-transmitted packets with respect to *n* access points within its range. The resultant RSSI n-tuple constitutes the "*signalprint*" of each client at that particular location. To deal with non-stationary clients, the *signalprint* of each wireless client is constantly tracked and updated. Spoofing is detected when significantly different *signalprints* are simultaneously detected for the same MAC address, indicating the existence of distinct wireless clients transmitting from different locations.

The physical layer approach to location fingerprinting was proposed in [26, 27], because the transmission medium between any two points is distinctive in space, time and frequency characteristics in a way that is location-specific [26]. Thus in applications where wireless nodes rarely alter their positions, it is possible to identify them through distinctive channel signatures. Taking this idea further, this physical characteristic can be further exploited to support secret key dissemination through probabilistic encryption [26].

The main limitation of this spoof detection technique is its association of identity to physical location. Methods that employ RSSI have limitations in their ability to resolve entities, as noted by the authors in [22]. On the other hand, methods that employ the characteristics of the RF channel are limited in their resolution by the wavelength of the wireless technology, which can be on the order of tens of centimeters for IEEE 802.11b/g. In addition, similar to the previous technique in section 2.2 based on sequence number tracking [10], the fingerprints are temporal in nature and not fundamentally linked to the identity of clients. Thus if a legitimate client changes its location while it is switched off, its legitimacy will be difficult to truly establish thereafter.

2.4 Wireless AP Fingerprinting

In [5], an active wireless fingerprinting technique that identifies unique access points (APs) instead of client stations is described. The author claimed the technique works because "physical differences in NICs board layout (paths dimensions, soldering, etc.) might lead to time variability when electrical signals are transmitted."

The test setup was such that a wireless client station would be regularly transmitting authentication request frames to the APs. Upon receiving each authentication frame, an AP replied with an acknowledgement, followed immediately by an authentication response frame. The fingerprinting approach was to conduct precise measurements of the time intervals between generation of the acknowledgement and the authentication response frame. Differences in timing intervals between APs were used to distinguish between unique APs. The Support Vector Machine (SVM) was used as the classification algorithm.

On average, the success rate in recognizing unique APs was reported to be approximately 86%. However, the technique was limited to the fingerprinting of APs but not wireless client stations, and is hence more useful for rogue AP detection. In addition, since elicits responses from APs, it is also an active, noisy approach which can be easily detected.

2.5 Wireless NIC Driver/Firmware Identification

Unlike other previously described work, the technique introduced in [6] identified the drivers and firmware used by wireless NIC instead of physical machines. Similar to the concept of OS fingerprinting, the purpose was to identify the wireless NIC driver used by a remote machine. Since vulnerabilities are often firmware and driver specific, this information may be used by a hacker to launch a directed attack based on that vulnerability.

The fingerprinting technique relied on subtle differences between driver developers on the implementation of the active scanning process – when wireless client stations periodically send out probe request frames to scan for APs. During the process of active scanning, the client cycles through each channel in the 2.4 GHz ISM band to broadcast probe request frames. After each frame is broadcasted in a channel, it waits for a period of time defined as the *MinChannelTime* if the channel is idle; otherwise it waits for *MaxChannelTime* before scanning the next channel. Because the standard does not specific the exact implementation details, the differences in implementations of the active scanning process was the basis for device driver fingerprinting in [6].

The fingerprinting technique was essentially a time-series analysis based on passively monitoring the network for 802.11 probe request frames and recording the time intervals between frames. By building the distributions of the time interval bins characterized by each known wireless NIC driver, a database of probabilistic signatures was built for each driver. The NIC driver from an unknown source could then be discovered using the Bayesian Classifier.

This technique was shown to be completely passive and did not require cooperation from the fingerprintee hosts. Since our approach is also based on the timing analysis of probe request frames, the advantages of this approach are shared by our technique. However, we have discovered consistent minute differences in timing intervals of probe request frames emitted from different machines, even when they used identical NIC drivers. Thus we have generalized this approach to distinguish between unique combinations of the tuple {Machine, Wireless Network Interface Card (NIC) Driver, Operating System} instead of just between unique drivers.

3. WIRELESS DEVICE FINGERPRINTING

Our fingerprinting technique aims to resolve the problem of identity resolution between 802.11 wireless client stations, by distinguishing between unique combinations of the tuple {Machine, Wireless Network Interface Card (NIC) Driver, Operating System}. We define the term device as this tuple for the remaining of this paper.

Our fingerprinting technique is a process with three phases:

1. Traffic Capturing Phase – Passive collection of wireless traces.

2. Fingerprint Generation Phase – Processing raw data to extract meaningful information.

3. Analysis Phase – Employing statistical significance testing to distinguish between unique devices.

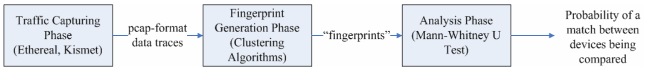
Traffic Capturing Phase (Ethereal, Kismet)
pcap-format data traces
Fingerprint Generation Phase (Clustering Algorithms)
"fingerprints"
Analysis Phase (Mann-Whitney U Test)
Probability of a match between devices being compared

Figure 1: High Level Overview of Fingerprinting Technique

3.1 Traffic Capturing Phase

In essence, the Traffic Capturing Phase involves the passive collection of probe request frames emitted from wireless client stations for timing analysis in the Fingerprint Generation Phase. The task of collecting probe request frames may be performed using tools such as Kismet.

Probe requests are used for active scanning in 802.11 to allow wireless client stations to detect the presence of APs within the vicinity, so that they can subsequently associate to a suitable AP. The process of active scanning, and thus the transmission of probe requests, starts automatically upon initialization of wireless NICs. Further, probe requests are still transmitted even after a client is associated with an AP, albeit at a different rate as compared to its unassociated state.

3.1.1 Intuition behind Timing Analysis on Probe Request Frames

The key idea of our technique is to distinguish between unique devices by timing analysis of probe request frames. It is critical to establish that for timing analysis of probe request frames to be meaningful, the probe request intervals for any particular device have to be cyclical or at least be predictable over the initial traffic capturing phase, during an adversary-free period of operation.

Our initial experiments appeared to confirm what has already been found in prior work [6], that periodic probe request intervals appear to be different across wireless NIC drivers. Figures 2 and 3 show the time delta between successive probe request frames received at the fingerprinter from two different types of wireless NIC drivers from a single channel.

The existence of repeated pulses of consistent amplitudes with different values for each wireless NIC confirms the results from the device driver fingerprinting technique in [6]. Probing deeper, what we have found to be even more interesting was that these *timing intervals changed in amplitude* when we repeated the same set of experiments using the *same NIC driver*, substituted with *different machines*, or when we *changed the operating system.* This means that these other factors also affect probe request intervals, suggesting that the NIC driver-specific fingerprints discovered in [6] may be OS specific and may have to be made more probabilistic to tolerate errors when the same NIC driver is used with different machines and/or different operating systems. What we found was the most challenging case was when *different machines* used the *same NIC driver* and the *same OS*, such that differences in timing intervals may become small and less apparent.

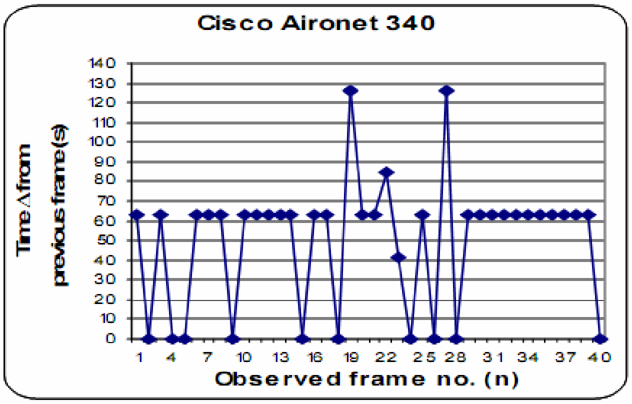

Figure 2: Active Scanning by Cisco Aironet 340

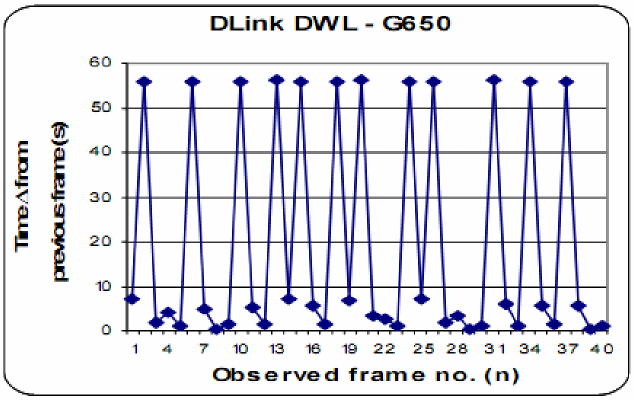

Figure 3: Active Scanning by DLink DWL-G650

Since an attempt to explain all the above effects based on hard evidence requires massive reverse engineering efforts, we adopt an inductive reasoning approach to explain our observations based on known facts – (1) The scanning interval in each channel is driver-specific as found in [6] and described in Section 2.5; (2) Clock skews exists between machines as found in [7] and described in Section 2.1; (3) Implementations of timing mechanisms are known to be operating system dependent; (4) Variability is also introduced through noisy channel effects. Now, let τ be the observed interval between two probe request frames. We can see that τ is ultimately the summation of following individual components:

$$\tau = \tau_{dev} + \delta_m + \delta_{os} + \delta_n,$$

where τ_{dev} is the driver-specific scanning interval; δ_m is variability due to clock skew between machines; δ_{os} is the variability due to the operating system; and δ_n is the variability due to noise.

When we compare between the probe request intervals of two machines using the *same driver* and the *same OS*, we are left with mainly $\delta_m + \delta_n$, which is miniscule in nature. Thus, in order to develop a feasible fingerprinting approach to distil out these differences, we must deal with two main problems: (1) The clock skew between machines must be measurable using our approach, i.e. it must not fall beyond the resolution limits of software-based measurement; (2) Our approach must expect data instances where noise dominates the probe request interval. To deal with this, it should separate the noisy data from relatively noiseless data, and perform robust comparisons in the presence of noise.

Motivated by the above reasons, we have designed the Course-Grained Data Selection in Section 3.2. to ensure that we extract the meaningful data where the clock skew is measurable. We accept that in some cases, the effects of noise will dominate the probe request interval. Our approach to deal with noise is to recognize and distil noisy data from relatively noiseless data by clustering, which we describe under Fine-Grained Data Selection (Section 3.2.2). The resultant set of clusters obtained for each device is what we call its device fingerprint. Finally, in order to be able to robustly compare between sets of device fingerprints, we adopt statistical tests of significance in our Analysis Phase. We will show through our experimental results in Section 4 that our technique can be used to detect changes in either the machine, NIC driver, or the OS.

3.2 Fingerprint Generation Phase

After the Traffic Capturing Phase, we would have acquired traces of probe request frames being transmitted by the device that we wish to fingerprint during its active scanning process. The next phase of Fingerprint Generation requires the selection of key points from the data that are valuable for distinguishing between devices.

3.2.1 Coarse-Grained Data Selection

The active scanning process may be seen to be characterized as cycles where each cycle involve: (1) a rapid burst of zero or more probe requests with tiny timing intervals in the range of milliseconds, followed by (2) a probe request after a large timing interval in the range of tens of seconds, which are seen as peaks in Figure 2 and Figure 3. We note that skews in timing between distinct machines are miniscule in nature – in [7], clock skews between distinct machines were found to be typically in the range of tens to hundreds of *µs/s*, or ppm (parts per million). This means that an attempt to distill differences in timing skews from measuring timing intervals in the range of milliseconds will not be meaningful, as the timing skews falls far below the resolution limits of software-based measurement. On the other hand, by focusing on the probe requests with large timing intervals in the range of tens of seconds (the large amplitudes in the figures), it becomes possible to distinguish between devices, though further processing would be required (as we will show in the next section) to remove the noise.

For ease of discussion, we term these large delays between cycles as the "inter-burst latencies". Figure 4 illustrates where inter-burst latencies are found when probe request frames are time-stamped against the first observed frame.

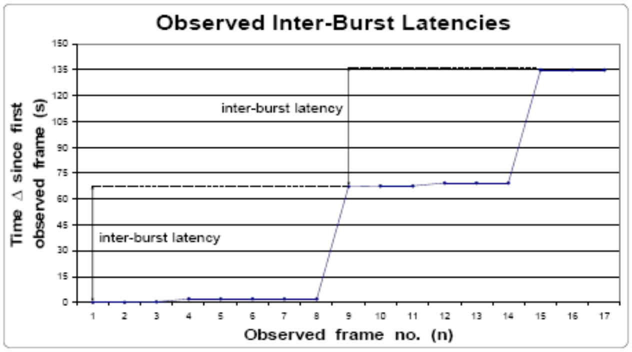

Figure 4: Inter-burst latency

Our first step to generate device fingerprints is hence to compile a series of inter-burst latencies observed from the devices, discarding the remaining points. Upon closer inspection of the resultant data, we also observed that the inter-burst latencies were actually packed into distinct groups rather than clustered around a centroid. Figure 5 and Figure 6 show the clusters of inter-burst latencies as captured from two different wireless NIC drivers.

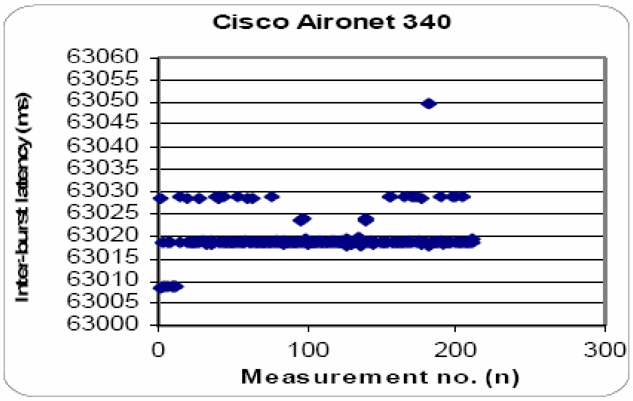

Figure 5: Inter-burst latencies of Cisco Aironet 340

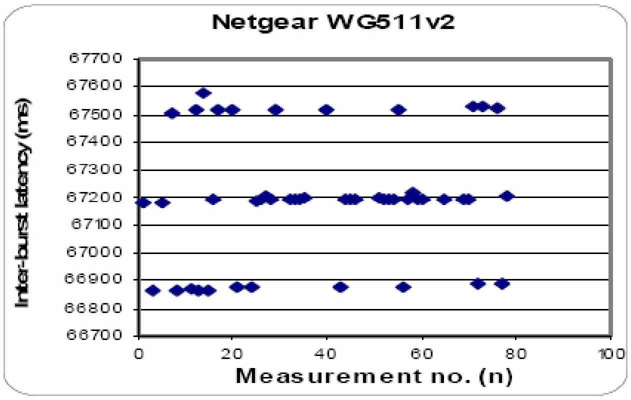

Figure 6: Inter-burst latencies of Netgear WG511v2

We believe the inter-burst latencies occupied distinct clusters due to rate adaptation algorithms implemented on the wireless NIC drivers [11]. Distinct clusters were formed due to the fact that the rate adaptation algorithms switch between transmission rates depending on wireless channel conditions. We use clustering techniques to automatically partition the inter-burst latencies into distinct clusters, a process that we call Fine-Grained Data Selection.

3.2.2 Fine-Grained Data Selection through Clustering

We employ a modified version of the Maximum Variance Clustering, where the reader is referred to [12, 13] for the details. The Maximum Variance Clustering approach requires the maximum variance (σ^2_{max}) within a cluster to be known a *priori* and specified as input, in contrast to *k*-means clustering which requires the number of clusters *k* to be known a *priori*. Since it is more practically feasible to determine the maximum variance of each cluster rather than the number of clusters, we have chosen the Maximum Variance Clustering as our clustering approach.

Our approach towards selecting an appropriate σ^2_{max} is as follows. As discussed in [12], cluster homogeneity can be read from the sum-of-squared-errors criterion, Je. Since changing the value of σ^2_{max} affects the cluster homogeneity, if we are able to compute constant values of Je for a substantial range of σ^2_{max} as σ^2_{max} is varied over a range, then there is a high possibility of having real cluster structure within this range of σ^2_{max}.

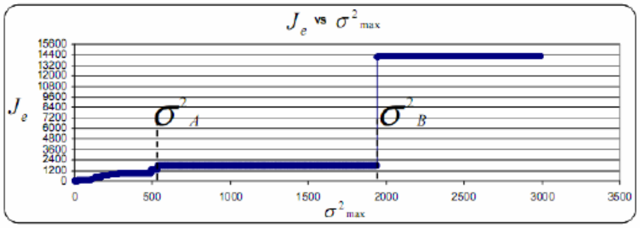

Figure 7: Je as a function of σ^2_{max}

As illustrated in the diagram above, Je plateaus are formed as σ^2_{max} varies, indicative of significant clusters forming at those corresponding σ^2_{max} values. For each plateau, let σ^2_A be the lower-bound variance that forms the plateau and σ^2_B be the upper-bound variance, so that $\sigma^2_A < \sigma^2_B$. The strength of each Je plateau can then be defined as the ratio:

$$S\left(\sigma^2{}_A, \sigma^2{}_B\right) = \frac{\sigma^2{}_B}{\sigma^2{}_A} .$$

As noted in [12], the strength of each Je plateau, $S(\sigma^2_A, \sigma^2_B)$, is indicative of the extent to which the clusters formed in the range $\sigma^2_A \leq \sigma^2_{max} \leq \sigma^2_B$ are represented by real structures in the data. Through experiments with artificial and real data sets in [12], the rule of thumb suggested was that real cluster structures start to be observed when the plateau approaches 2. Intuitively, the explanation for this is that two real clusters will be lumped together if the variance constraint σ^2_{max} is higher than the individual variance of each cluster.

Based on the above observations, our strategy is to find σ^2_A at the maxima of $S(\sigma^2_A, \sigma^2_B)$ within the range $S(\sigma^2_A, \sigma^2_B) \leq 2$, in the search space $\sigma^2_A, \sigma^2_B \in \{1.. \sigma^2_U\}$, where σ^2_U is the upper limit in the search space for σ^2_{max}. The variance thus obtained, σ^2_A, is then used as the Maximum Variance σ^2_{max} in the Maximum Variance Clustering algorithm.

We observed from Figures 5 & 6 that the data points of perceived clusters were generally located within 10ms from one another, consistent with the results in [14]. Thus we used $\sigma^2_U = 50$ as the upper limit in the search space for σ^2_{max} since this is the variance derived from a cluster with two points 10 ms apart.

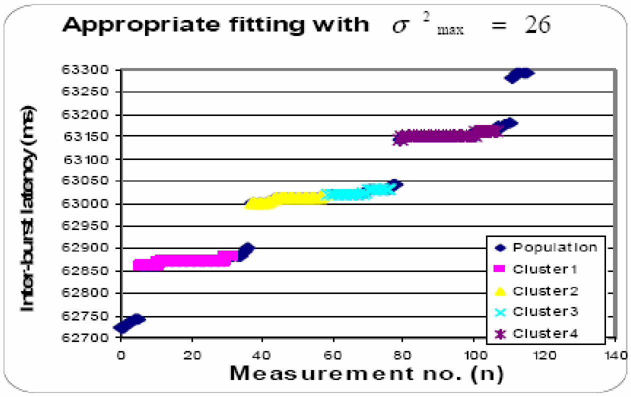

Figure 8: Appropriate Fitting with suitable σ^2_{max}

Figure 8 shows the results of applying the clustering algorithm on a series of inter-burst latencies from a wireless device equipped with Asus WL-167G in a relatively noise free environment, where the data points have been sorted for presentation. At present, our algorithm only retains clusters of size larger than or equal to 8 – clusters smaller than that are discarded.

To summarise, these clusters of inter-burst latencies are the processed timing measurements of the probe requests generated by a wireless client device during active scanning. Hence, the fingerprints of unique devices are represented by the clusters of inter-burst latencies. We now proceed to the Analysis Phase where comparisons between sets of fingerprints are used to determine if they were actually generated from different devices with at certain probability.

3.3 Analysis Phase

The Analysis Phase is the final step where the task of fingerprint comparison is performed to distinguish between unique devices. Using the data point clusters obtained after the Fingerprint Generation Phase, we employ statistical hypothesis testing to determine if supposedly different traffic traces captured (based on inspection of the MAC addresses) are actually emitted from different devices.

We note that the key consideration in statistical hypothesis testing is to choose the right statistical test based on the nature of the data collected [21]. We propose the use of the Mann-Whitney U-test [16] for its statistical strength in determining if two samples are derived from the same distribution. In which case, two samples of inter-burst latencies can be tested if they are from the same distribution. Furthermore, the Mann-Whitney U-test is a non-parametric test, hence does not rely on any assumptions that the data are drawn from a given distribution. This is significant as it would be inappropriate to assume that all wireless devices generate probe request frames based on a common underlying distribution.

Before running the U-test, we need to extract the best suited clusters from the two traces to compare. At present our approach is to compare the cluster means between the two sets of device fingerprints and select the nearest two clusters for statistical testing. It is however also possible to use multiple clusters from each fingerprint set to increase accuracy in comparison - we discuss the possibility of this in the last section on future work.

4. EXPERIMENTAL RESULTS

The objectives of our experiments were to determine whether our fingerprinting technique was able to:

1. Accurately differentiate between unique devices;
2. Produce consistent results for each particular device over time, on separate occasions, and under varying network loads; and
3. Distinguish between different machines in the most challenging case where the machines have the same specifications (model, RAM, processor, etc), and are equipped with the same OS and wireless NIC drivers.

We conducted experiments across different environments in both controlled and real operating environments to validate our technique and for all the tests performed, where the *Null Hypothesis* H_0 for our Mann-Whitney U-test was that the samples were generated by the *same wireless device*. Thus depending on whether the same or different devices were used in the actual condition, the test outcomes could be Correct or Wrong as defined in Table 1.

Table 1: Table of Test Outcomes based on Actual Conditions

		Actual Condition	
		Different Devices	**Same Device**
Test Result	**Positive** (Reject H0)	Correct (True Positive)	Wrong (False Positive)
	Negative (Fail to reject H0)	Wrong (False Negative)	Correct (True Negative)

4.1 Controlled Environments

The first set of experiments was conducted in a controlled environment with little noise and interference, with only a single AP and four laptops (one laptop as the fingerprinter and the other three acts as the fingerprintees) in the vicinity. The fingerprintees were three Fujitsu laptops (P1, P2 and P3) with *identical specifications* (Intel Centrino 1.6 GHz, 512 MB RAM), the *same OS* - Windows XP Service Pack 2, and the *same wireless NICs and drivers*. Thus the only variable within the tuple constituting the device was the machine. The experiment involved collecting traffic from each of them simultaneously for eight hours. This process was repeated for *three different wireless NICs* and their respective drivers (Asus WL-167G, DLink DWL-G650 and the built-in Intel Wireless NIC).

4.1.1 Comparisons between Similar Devices

Our first experiment aimed to determine if we could successfully differentiate between the three devices (P1, P2 and P3) despite all their similarities. For the Asus and DLink wireless NIC drivers, we did six pair-wise U-tests (on the hourly generated fingerprints) between P1 vs P2, P1 vs P3 and P2 vs P3. And for the Intel wireless NIC driver, three pair-wise U-tests were conducted. In total 15 U-tests were conducted for each pair of comparison.

Table 2: Results of fingerprinting 3 wireless devices with identical specifications, OS and wireless NIC Driver

	Correct (True Positive)	Wrong (False Negative)
P1 vs P2	86.67%	13.33%
P1 vs P3	100.00%	0%
P2 vs P3	60.00%	40.00%
Average	82.22%	17.78%

On average, our fingerprinting technique was 82% accurate at differentiating between these three similar devices, at a confidence level of 95%. We noticed that between P2 and P3, our fingerprinting technique did not fare as well. A probable reason for this was that these two machines happened to be too similar, sharing little physical hardware / clock disparities. We discuss possible refinements to improve our technique on such machines in the last section on future work.

4.1.2 Fingerprinting the Same Device

We next investigate the effect of performing pair-wise comparisons between traces from the same device using the exact test settings. For these tests we should desire a conclusion that fails to reject the Null Hypothesis ("Correct"). The comparisons were done only for traces captured from one of the wireless NIC (Asus WL-167G) and a total of 15 U-tests were performed for each machine.

Table 3: Results of fingerprinting the same device

	Correct (True Negative)	Wrong (False Positive)
P1 vs P2	100.00%	0%
P1 vs P3	86.67%	13.33%
P2 vs P3	100.00%	0%
Average	95.56%	4.44%

Although in statistical hypothesis testing we could not draw the direct conclusion that the devices being tested were the same when we the Null Hypothesis was accepted, the results indicated at least that they were not uniquely different devices (True Negatives). On average, our fingerprinting technique was correct 95% of the time.

4.1.3 Investigating the Effects of Distance

One of the most practical factors that we needed to test was the effect of distance between the fingerprintee and fingerprinter on our fingerprinting technique. We collected another five hours of traces from P1 and P2 (both equipped with DLink DWL-G650) after locating them approximately 50m and 25m further away from the fingerprinter. In addition, in this particular experiment there were walls in between the fingerprinter and {P1, P2}. We then ran a total of 100 pair-wise U-test of this data against the other hourly data sets previously collected from P1 and P2.

We first present the overall results of comparing the newly located P1 against P2 and P3, and the newly located P2 against P1 and P3. For these tests we should desire a conclusion that rejects the Null Hypothesis to acknowledge that the devices were indeed different.

Table 4: Effect of distance on fingerprinting different devices

	Correct (True Positive)	Wrong (False Negative)
$P1_{new}$ vs P2	80.00%	20.00%
$P1_{new}$ vs P3	80.00%	20.00%
$P2_{new}$ vs P1	92.00%	8.00%
$P2_{new}$ vs P3	88.00%	12.00%
Average	85.00%	15.00%

Our results indicated a high average of 85% accuracy in differentiating the devices despite moving to them to new locations. It should be no surprise that the detection rate has decreased in some cases, due to the effects of signal attenuation, multi-path propagation and fading with increasing distance. However, the results indicate that our technique remains practical for the distances seen in real wireless LANs.

We next examine the results of comparing between the fingerprints from the newly located devices with the fingerprints generated from their previous positions.

Table 5: Results of fingerprinting the same device at different distances

	Correct (True Negative)	Wrong (False Positive)
P1	68.00%	32.00%
P2	72.00%	28.00%
Average	70.00%	30.00%

Although we have maintained our ability to perceive the same device as the distance increased, the False Positive Rate has increased as well. Nonetheless, the overall result achieved 70% accuracy rate in correctly identifying the same device.

4.2 Public Hotspot

Our next experiment was conducted in a public hotspot in real operational use. Our traffic capturing phase was conducted on four separate days of about three hours each. However, we performed the analysis only on devices that stayed in the hotspot for at least a continuous hour.

The scenario for this set of experiments was drastically different from the previous controlled environments. We detected more than one AP in the vicinity and there were at least five wireless client stations at any one point of time. This environment provided a real-world scenario to determine if our technique would be able to perform well in practice.

From our traces, we observed about 72 unique MAC addresses that stayed for at least a continuous hour within the WLAN. However, in some cases there were still insufficient data to form clusters of at least eight points, which our algorithm discarded as we have noted at the end of Section 3.2.2. After filtering the traces that did not provide sufficient data for our fingerprinting technique, we had 45 unique devices to perform comparisons against each other.

Table 6: Fingerprinting technique applied in public hotspot

	Correct (True Positive)	Wrong (False Negative)
Average	81.85%	18.15%

The results indicated that our fingerprinting technique achieved an average of 81% accuracy at differentiating the different devices at a confidence level of 95%. We believe the variety of wireless NIC drivers and OS being used by different clients contributed partly to the accuracy of our technique.

4.3 Large Scale Conference

Our last test scenario involved the use of large scale traces [17] obtained during the Special Interest Group on Data Communications (SIGCOMM 2004) conference. As our previous experiments were performed on much smaller scales, the results were less conclusive for testing the effectiveness of our fingerprinting technique. The use of this large scale data set provided a much stronger conclusion.

In total, we observe about 211 unique MAC addresses that stayed connected for at least a consecutive hour. After filtering the traces that did not provide sufficient data for our fingerprinting technique, we had 174 unique devices to perform comparisons against each other.

Table 7: Fingerprinting technique applied on data from SIGCOMM 2004

	Correct (True Positive)	Wrong (False Negative)
Average	71.30%	28.70%

Our fingerprinting technique was able to differentiate the devices about 71% of the time. Again, we witnessed the rates decreasing slightly as the surrounding environment became saturated with wireless users. The effects of distance and walls also possibly contributed to the decrease in accuracy.

5. APPLICATIONS

We now consider some important applications that may be realized by our technique. Note that we do not elaborate extensively on applications as we consider our most important contribution to be the wireless device fingerprinting technique that we have laid in the previous sections. We believe that our technique can be tuned to suit any application where identity resolution in WLAN is a problem.

5.1 Spoof Detection

As mentioned in the Introduction section, at present WEP can be cracked and even with WPA, management and control frames remain susceptible to MAC spoofing / replay / masquerading attacks due to lack of authentication and encryption for such frames.

We see potential in applying our fingerprinting technique to spoof detection, complementing the weaknesses of the existing WEP/WPA schemes. This requires, for example, the maintenance of a white-list of allowed devices, each associated with its MAC addresses and wireless fingerprint profile. An alien device that attempts to masquerade using a valid MAC address will be picked up through device fingerprinting. In stark contrast with identification based on MAC address, our technique would prove to be much more challenging to evade.

We note that future improvements in accuracy will be desirable for our technique to be more effective for spoof detection. We believe there are many possible ways to improve our technique that we have not yet explored, for example, by comparing all obtained clusters from two device fingerprints, and not just the two nearest clusters.

5.2 Network Forensics

At present, network logs are recorded such that the notion of virtual identity is tied to IP addresses, MAC addresses and user accounts. A problem that easily arises in wireless LANs occurs especially when virtual identities are entirely spoofed during an attack. Thus intrusion detection records and network traces may in fact point to the wrong culprit, instead of identifying the right attacker.

Again, this problem can be resolved if wireless device fingerprints can be used as an independent mechanism to identify unique devices.

5.3 Network Profiling and Reconnaissance

Our technique can also be useful for the purpose of network profiling and reconnaissance. For example, a honeynet may be set up to simulate the presence of multiple virtual hosts on a single physical machine. The intention is normally to lure attackers into honeypots to keep them away from real production systems, as well as to study the attack techniques employed by the attackers.

Our device fingerprinting technique can be applied to indicate (with some probability) whether a set of candidate MAC or IP addresses observed in the channel actually come from the same physical device.

6. LIMITATIONS

As with all other techniques, we acknowledge that our technique is faced with certain limitations and it is possible to develop countermeasures to evade our technique.

The first limitation is fundamental to statistical hypothesis testing. When the Null Hypothesis - that the samples came from the same device - is not rejected, we cannot conclude that two devices are actually the same. Statistically, we can only establish that there is not enough evidence to affirm that the devices are different, but the rejection of the alternate hypothesis could be due to other factors, e.g. due to limited samples or simply because two unique devices simply happened to be extremely similar.

The second issue is the amount of time and data required to fingerprint each device. As our device fingerprinting technique focuses mainly on analyzing the inter-burst latencies based on clustering, at present it takes about at least an hour before we can gather enough data to perform the fingerprinting. Some wireless NICs such as those from Intel perform the active scanning procedure at a much slower rate and require at least two hours before enough data can be harvested.

Perhaps the most challenging issue we face is the lossy nature of wireless communications. Due to shadowing, interferences and channel fading effects, packet losses and delays inevitably occur, thereby decreasing the efficacy of our timing-based fingerprinting technique [18]. In particular, in a congested environment, 802.11 medium access control kicks in, causing back-offs with a certain amount of random time involved. This makes it hard to characterize. By using measurements of the ambient background traffic, it is possible to normalize the device fingerprinting to cope with medium access control contention, as was done in [28].

While designing our technique, we have been constantly scrutinizing our implementation, looking for possible countermeasures that may potentially evade or fool our fingerprinting technique. One such possible method would be to modify parts of the driver code to inject randomness into the inter-burst latency. Alternatively, an Attacker may also attempt to confuse our technique by mixing the probe request frames from his masqueraded device with the frames from a concurrently active legitimate device. This attack may cause a denial of service on the legitimate device being spoofed as this alters the measured inter-burst latency intervals and the fingerprints for that device.

Last but not least, the effects of temperature may alter clock oscillations, which may affect the accuracy of fingerprint. This effect was studied in [20]. In fact, this concept was applied to fingerprint Tor servers in [21]. By similar logic, an attacker may alter the fingerprint of his own machine by altering its temperature; he could also remotely alter the fingerprint of his victim by deliberately increasing the CPU load (and thus the temperature) of his victim, thereby potentially deceiving our fingerprinting technique.

To mitigate against the above attacks, we note that the overall robustness would be increased by using our technique in combination with other techniques based on different approaches. For example, when using our technique in combination with frame sequence number tracking (section 2.2, [10]) and RSSI-based location tracking (section 2.3, [22]), an Attacker would have to also use the right frame sequence numbers from the right physical location in order for the attack to be successful.

7. CONCLUSION

We have proposed a new fingerprinting technique that differentiates between unique devices over a wireless Local Area Network simply through the timing analysis of 802.11 probe request frames. Our technique can be applied to spoof detection, network reconnaissance, and implementation of access control against masquerading attacks. In contrast with existing wireless fingerprinting techniques, our technique is passive, non-invasive and does not require the co-operation of fingerprintee hosts.

Experimental results further justify the effectiveness of our technique. Depending upon the environment, we achieved an average accuracy rate of about 70% to 80% in differentiating between unique devices. Even for machines with the same specifications that were equipped with the same OS and similar wireless NIC drivers, we could achieve an accuracy rate of 82% in distinguishing between them. As part of the future work, we see the following areas as possible improvements:

1. Multi-Cluster Comparison – Instead of selecting the cluster with the least intra-cluster variance as the fingerprint for inter-device comparison, we could instead perform a comparison using multiple clusters. The p-value of the U-test is a non-parametric measure of the overlap between two distributions. Thus, it might be useful to come up with an overall metric based on the p-values of individual cluster comparisons.

2. Inclusion of More Test Metrics – Our current implementation relies solely on the inter-burst latency as criteria for differentiating the devices. More metrics could be included to increase the robustness of the technique.

8. ACKNOWLEDGMENTS

We would like to thank Dr. Anthony K. H. Tung for his advice and invaluable knowledge on data mining techniques; Tzu Hon Ng for performing further experiments and refinements of the approach. We also thank our shepherd, Dr. Wade Trappe and the anonymous reviewers for many insightful and helpful comments.

9. REFERENCES

[1] AirDefense. Website, 2007, http://www.airdefense.net/index.php.

[2] Snort Intrusion Detection and Prevention System, 2007, http://www.snort.org.

[3] Ryan M. Gerdes, Thomas E. Daniels, Mani Mina, Steve F. Russell. Device Identification via Analog Signal Fingerprinting: A Matched Filter Approach. In Proceedings of the Network and Distributed System Security Symposium, NDSS 2006, San Diego, California, USA

[4] Jeyanthi Hall, Michel Barbeau, Evangelos Kranakis. Radio Frequency Fingerprinting for Intrusion Detection in Wireless Networks. In IEEE Transactions on Defendable and Secure Computing, July 2005.

[5] Bartlomiej Sieka. Active Fingerprinting of 802.11 Devices by Timing. In IEEE Consumer Communications and Networking Conference (CCNC 2006), Las Vegas, NV, USA

[6] Jason Franklin, Damon McCoy, Parisa Tabriz, Vicentiu Neagoe, Jamie Van Randwyk, Douglas Sicker. Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting. In Proceedings of the 15th USENIX Security Symposium, Vancouver, Canada, 2006.

[7] Tadayoshi Kohno, Andre Broido, and K.C. Claffy. Remote Physical Device Fingerprinting. In Proceedings of the 2005 IEEE Symposium on Security and Privacy (SP 2005), Washington, DC, USA

[8] Vern Paxson. On Calibrating Measurements of Packet Transit Times. In Proceedings of SIGMETRICS '98, June 1998. June 1998.

[9] Sue B. Moont, Paul Skelly, Don Towsley. Estimation and Removal of Clock Skew from Network Delay Measurements. In Proceedings of IEEE INFOCOM '99, New York, NY

[10] Joshua Wright. (2003). Detecting Wireless LAN MAC Address Spoofing. http://home.jwu.edu/jwright/

[11] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. IEEE 802.11 Rate Adaptation: A Practical

Approach. In Proceedings of the 7th ACM International Symposium on Modelling, Analysis and Simulation of Wireless and Mobile Systems, 2004

[12] Cor J. Veenman, Marcel J.T. Reinders and Eric Backer. A Maximum Variance Cluster Algorithm. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24, No.9, 2002, pp. 1273 –1280

[13] Cor J. Veenman and Marcel J.T. Reinders. The Nearest Subclass Classifier: A Compromise between the Nearest Mean and Nearest Neighbour Classifier. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.27, No.9, 2005, pp. 1417 – 1429

[14] Cherita Corbett, Raheem Beyah and John Copeland. A Passive Approach to Wireless NIC Identification. In Proceedings of IEEE International Conference on Communications (ICC), 2006.

[15] S. S. Shapiro and M. B. Wilk. An Analysis of Variance Test for Normality (Complete Samples). In Biometrika, Vol.52, No.3/4, 1965, pp. 591 – 611.

[16] H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. In The Annals of Mathematical Statistics, Vol.18, No.1, 1947, pp. 50 – 60.

[17] Maya Rodrig, Charles Reis, Ratul Mahajan, David Wetherall, John Zahorjan and Ed Lazowska. {CRAWDAD} trace uw/sigcomm2004/wireless/sahara (v. 2006-10-17)

[18] Felix Hernandez-Campos and Maria Papadopouli. Assessing the Real Impact of 802.11 WLANs: A Large-Scale Comparison of Wired and Wireless Traffic. In Proceedings of the 14th IEEE Workshop on Local and Metropolitan Area Networks, Crete, Greece, 2005

[19] David W. Allan, Chairman, James A. Barnes, Franco Cordara, Michael Garvey, William Hanson, Jack Kusters, Robert Smythe and Fred L. Walls. Precision Oscillators: Dependence of Frequency on Temperature, Humidity and Pressure. In Proceedings of the IEEE Frequency Control Symposium, 1992

[20] Steven J. Murdoch. Hot or Not: Revealing Hidden Services by their Clock Skew. In ACM Conference on Computer and Communications Security, 2006.

[21] G. David Garson, “Significance Testing", from Statnotes: Topics in Multivariate Analysis. Retrieved 09/20/2007 from http://www2.chass.ncsu.edu/garson/pa765/statnote.htm

[22] Daniel B. Faria, David R. Cheriton, Detecting Identity-Based Attacks in Wireless Networks Using Signalprints. In Proceedings of the 5th ACM workshop on Wireless security, WiSec 2006.

[23] Nikita Borisov, Ian Goldberg, and David Wagner, Intercepting mobile communications: the insecurity of 802.11. In Proceedings of ACM MobiCom 2001, pp. 180-189.

[24] Erik Tews, Ralf-Philipp Weinmann and Andrei Pyshkin. Breaking 104 bit WEP in less than 60 seconds. In Cryptology ePrint Archive, Report 2007/120, 2007.

[25] Kasper Bonne Rasmussen and Srdjan Capkun. Implications of radio fingerprinting on the security of sensor networks. In Proceedings of IEEE SecureComm, 2007.

[26] L. Xiao, L. J. Greenstein, N. B. Mandayam, W. Trappe, "Using the Physical Layer for Wireless Authentication under Time-variant Channels", to appear in IEEE Transactions on Wireless Communications, Feb 2007.

[27] Neal Patwari, Sneha Kumar Kasera. Robust location distinction using temporal link signatures. In Proceedings of the 13th annual ACM international conference on Mobile computing and networking (MOBICOM 2007), pp 111-122.

[28] Qing Li and Wade Trappe. Detecting Spoofing and Anomalous Traffic in Wireless Networks via Forge-Resistant Relationships. In IEEE Transactions on Information Forensics and Security, Volume 2, Issue 4, Dec. 2007, pp 793 - 808.