Good Neighbor: Secure Pairing of Nearby Wireless Devices by Multiple Antennas*

Liang Cai Kai Zeng Hao Chen Prasant Mohapatra
University of California, Davis
Computer Science Department
2063 Kemper Hall, 1 Shields Avenue, Davis, CA 95616
lngcai@ucdavis.edu {kzeng,hchen,prasant}@cs.ucdavis.edu

Abstract

The proliferation of personal wireless devices requires secure connection between them. While it is easy to securely pair electronic devices by wires, it is very challenging to pair them wirelessly when they have no prior association. We propose Good Neighbor, a novel scheme that securely pairs nearby wireless devices by exploiting multiple antennas built in them. Our scheme requires neither shared secrets nor out-of-band channels (e.g., audio, visual, keyboard, etc.) between the pairing devices. It only requires that the receiver has multiple antennas and that the sender can be placed nearby the receiver. Our scheme is based on the propagation characteristic of the wireless signal that the power of the received signal is inversely proportional to some exponent of the distance between the sender and receiver. When a nearby sender moves very close to one antenna on the receiver, the receiver can observe a large difference between the signal strength measured on its two antennas, whereas a faraway sender would be unable to induce such a large difference. We validate our scheme through theoretical analysis and experimental measurements. We discuss the factors that may affect our scheme — including antenna gain, received signal strength (RSS) saturation, dynamic rate adaptation, and multipath effects — and how to mitigate them. Finally, we demonstrate the practicality of our scheme by implementing and evaluating a prototype.

1 Introduction

The proliferation of wireless devices requires secure connection between these devices. However, how to set up a secure connection between two previously unassociated devices remains an important yet challenging problem. Most current schemes rely on a common secret to bootstrap the secure connection. However, creating a strong secret and delivering it to both devices often poses usability challenges. First, users are known not to be competent at creating strong secrets. Second, users have to go through different, and often laborious and unintuitive, procedures to enter the secret on different devices. This mechanism becomes even more problematic for devices with no keyboard, such as wireless headphones. In this case, manufacturers often hardcode the secrets in the devices and print them in the manuals. For usability reasons, manufacturers tend to choose easy-to-remember secrets, such as `0000` for many bluetooth headphones, which completely defeats the purpose of shared secrets.

Moreover, we often need to set up ad hoc, temporary connections between nearby devices. For example, two business people wish to exchange contact information via their cell phones, and a group of tourists wish to exchange photos in their wireless-capable cameras. We can establish such ad hoc connections between unassociated devices easily via wires, but to do so wirelessly is very challenging. This problem is called *Secure Device Paring*.

The *Wi-Fi Protected Setup* (WPS) [2] standard from *Wi-Fi Alliance* specifies four device pairing methods. (1) PBC method: the user pushes a hardware or software button on both devices; (2) PIN method: the user reads a PIN from one device and enters it at the other; (3) NFC (Near Field Communication) method: the user brings the devices close enough to allow near field communication between them (such as RFID tags); (4) USB method: the user transfers data between the devices using either a USB flash drive or a USB cable. Approaches proposed by researchers for secure device pairing, including the above four methods, fall into two categories: (1) based on out-of-band channels, and (2) based on proximity.

*This work was partially supported by the National Science Foundation through grants CNS 0644450, 1018964, and 0709264, and the Army Research Office through MURI grant W911NF-07-1-0318.

Kobsa, et al. [10] compared device pairing schemes based on out-of-band channels, such as acoustic [16, 6], visual [14, 21], and motion [13, 8]. These schemes require either sensors — such as cameras, microphones, or accelerometers – or peripherals, such as displays or keyboards. As wireless capability is expanding to a wide variety of devices (such as cameras, scanners, or even digital picture frames) that do not have these sensors or peripherals, the scope of applicability of these schemes is limited.

Alternatively, device pairing can be based on proximity. In many circumstances, the adversary cannot come close to the user's devices (or cannot do so without being detected). Frank Stajano described many scenarios where the user wishes to pair any devices within proximity [24]. Using a USB cable is a form of proximity-based pairing; however, since it requires a cable and USB interfaces on both devices, its applicability is limited. The NFC method in WPS is also a proximity-based method; however, it is vulnerable to attacks using powerful transmitting and receiving antennas. Distance bounding protocols [3, 4, 19] are resilient to these attacks; however, since they require highly precise clocks (of nanosecond precision) because electromagnetic waves propagate over 30cm in 1 nanosecond, they are unsuitable for many consumer wireless devices [20].

We propose a simple yet reliable proximity-based device pairing scheme by taking advantage of multiple antennas available on many modern wireless devices. Our scheme only requires that one of the pairing devices has at least two antennas (We call the device with multiple antennas the *receiver*, and the other one the *sender*. If both devices have multiple antennas, either one can serve as the receiver). Our scheme requires neither shared secrets nor out-of-band channels between the devices. Our key insight is that the difference in the received signal strengths (RSS) on different antennas on the receiver can indicate if the sender is nearby. RSS is inversely proportional to some exponent of the distance between the sender and the receiver. When the user places the sender very close to one antenna on the receiver, the receiving signal strength on this antenna would be far greater than that on the other antennas on the receiver. By contrast, when the sender is far from the receiver, it is of similar distance from all the antennas on the receiver and therefore would be unable to cause a large difference in the RSS values. Although a faraway attacker can attack the NFC method by increasing its transmitting power, such an attack has no effect on our method because transmitting power does not affect the diffierence of RSS values between different antennas on the receiver.

Since our scheme requires neither sensors nor peripherals, it can be applied to simple wireless devices like Eye-Fi [1] cards. Our scheme requires that one of the pairing devices has at least two antennas. Even though not all the wireless devices have multiple antennas yet, we expect multiple antennas to become widely available soon as wireless devices embrace the *multiple-input multiple-output* (MIMO) technology proposed in IEEE 802.11n to increase their maximum raw data rates.

Although multiple RSS values have been explored for location inference [11, 25], we are the first, to the best of our knowledge, to apply it to secure device pairing. Since device pairing requires a much more precise estimation of proximity than the previous schemes, as the attacker may come within a reasonable distance from the user's wireless device (e.g., at an airport lounge), we need to overcome a series of challenges (Section 4) in designing our scheme.

2 Proximity detection based on differential RSS

Compared to the difficulty in pairing wireless devices, pairing wireline devices is often straightforward: by simply plugging one device into the other. This approach is secure against all adversaries that are physically distant from the pairing devices. Due to the simplicity of this scheme, one naturally wishes to find a similar mechanism to securely pair wireless devices that are in immediate proximity. As is well known in wireless communication, the *received signal strength* (RSS) depends on the distance between the sender and receiver. A naive idea would be to infer proximity based on the RSS value alone. However, a faraway attacker could defeat this naive scheme by sending powerful signal to induce large RSS values on the receiver

We can defeat the above attack if the receiver has two antennas that can measure RSS independently. Most recent laptops, including all the laptops with 802.11n MIMO modules, have two or more antennas to take advantage of antenna diversity. Although not all mobile devices currently have multiple antennas, we expect multiple antennas to appear on these devices soon as they embrace the MIMO technology to improve their data rate. Note that our scheme requires only one of the two pairng device to have multiple antennas (we call this device the *receiver*, and the other device the *sender*).

Our key observation is that the ratio between the RSS values measured on the multiple antennas on the receiver is independent of the sending power. However, the ratio depends on the difference between the distances between the sender and the two receiving antennas. While a nearby sender can make this difference large, a faraway sender cannot.

2.1 Theories of RSS

2.1.1 Free space propagation model

In the absence of any reflections or multipath, we can model radio wave propagation using the free space propagation model in Equation (1) [18]. The power of the signal at the receiving antenna is:

$$P_r = P_s G_s G_r (\frac{\lambda}{4\pi d})^2. \quad (1)$$

where G_r and G_s are the gains of the receiving and sending antenna, respectively, P_s is the power at the surface of the sending antenna, and d is the distance between the two antennas.

When we represent P_r in dBm:

$$P_r[dBm] = P_0 - 20log(\frac{d}{d_0}). \quad (2)$$

where P_0 is the power of the signal in dBm at distance d_0 away from the sender.

2.1.2 Log-normal shadowing model

A more widely used signal propagation model is log-normal shadowing [18].

$$P_r[dBm] = P_0 - 10\alpha log(\frac{d}{d_0}) + X_\sigma. \quad (3)$$

where P_0 is the receiving power at distance d_0, α is the path loss exponent, and X_σ is a Gaussian noise (random variable) with zero mean and standard deviation σ. The path loss exponent α depends on the specific propagation environment, i.e., type of construction material, architecture, and location within a building. The values of α range from 1.2 (Waveguide effect) to 8 [15]. In free space, α is 2.

2.2 RSS ratio

Inferring distance by RSS alone is difficult because of the uncertainty of α and X_σ, especially in a dynamic environment. However, for device pairing, the influence from the environment is small because the sender and the receiver are close. We evaluate our hypothesis with a series of experiments in Section 4.

The setting of our RSS based proximity inference scheme is shown in Figure 1. The receiver R has two antennas, A_1 and A_2, separated by a reasonable distance L. When R receives a packet, R reads the RSS values (RSS_1 and RSS_2) independently on A_1 and A_2, respectively. Since RSS is a value in dBm[1], we term the difference between the two RSS values, $r = RSS_1 - RSS_2$, as *RSS ratio*.

[1] *RSS* is the ratio of the power of the received signal (P in mW) to $1mW$ in decibels, i.e., $RSS = 10\log_{10}(P)$.

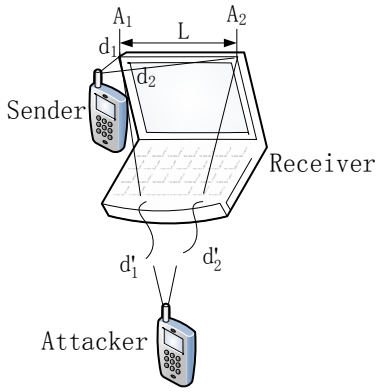

Figure 1. Our scheme requires the receiver to be a wireless device with at least two antennas. We use a laptop equipped with 802.11n MIMO antenna in our prototype system.

We assume that the RSS_1 and RSS_2 values follow the Log-normal shadowing model. For clarity, we let RSS_1 and RSS_2 denote the average of sufficient number of RSS measurements so that X_σ can be removed. Therefore, the RSS ratio r observed at the receiver is $10\alpha log(\frac{d_2}{d_1})$. When the sender is placed close to A_1, d_1 is very small while $d_2/d_1 \approx l/d_1$ is large, so r becomes a large positive value. Similarly, when the sender is moved to A_2, d_2 decreases and $d_2/d_1 \approx d_2/l$ becomes very small, so r becomes a large negative value.

When an attacker not in the proximity of the receiver sends packets, the RSS ratio r observed at the receiver is $10\alpha log(\frac{d'_2}{d'_1})$. The largest value of $|r|$ that the attacker can incur is $(d'_1 + l)/d'_1$ where the attacker's antenna is on the same line of A_1A_2 and is closer to A_1(or A_2). When d'_1 is sufficiently larger than L, $|r|$ is a small number. In other words, a faraway attacker is unable to yield a large RSS ratio no matter where the attacker is. Based on this observation, the receiver can choose appropriate thresholds r_H (when the sender is close to A_1) and r_L (when the sender is close to A_2) to distinguish a faraway attacker from a legitimate nearby sender.

3 Design

3.1 Goal and threat model

Our goal is to build a practical, reliable scheme for securely pairing nearby devices that have no prior association. In this paper, we only consider one-way authentication, i.e., only the receiver authenticates the sender but not the other way around. In many scenarios, only one-way authentica-

tion is necessary. For example, when a user wants to transfer her personal files from the receiver (e.g. laptop) to the sender (e.g. PDA), she only requires the receiver to authenticate the sender. It is straightforward to extend this one-way authentication into mutual authentication if the sender also has multiple antennas.

Our scheme requires only that the legitimate sender be physically close to the receiver. We wish to ensure that no faraway malicious sender can be paired with the receiver successfully. Our scheme can resist powerful attackers. For example, the attacker may have arbitrarily high transmission power and can adjust the transmission power arbitrarily; he may sniff all the traffic between the two pairing wireless devices; he may have exact copies of the two pairing devices and use the copies to attack our scheme; he may know the exact location of the receiver and its antennas; he may send his attack packets via line-of-sight propagation.

However, we exclude the following threats, as they are out of the scope of this paper:

- Compromising either the receiver or a legitimate sender, e.g., by malware infection.
- Jamming the wireless channel.

3.2 Basic scheme

Let the two antennas on the receiver R be A_1 and A_2. When the user places the sender very close to the antenna A_1 on the receiver R, R expects to observe a large positive RSS ratio (Section 2.2). Then, when the user moves the sender very close to the antenna A_2, R expects to observe a large negative RSS. By contrast, if the sender is faraway from the receiver, R cannot observe large absolute values of the ratio.

Our scheme requires the sender to be placed close to both the receiving antennas sequentially to reduce the probability of the "walk-by" attack, where the attacker places his sender very close to the receiver by walking by the receiver without raising suspicion. However, it would be very difficult for the attacker to place his sender close to both the receiving antennas sequentially during a an inconspicuous walk-by.

We checked the feasibility of this scheme on a laptop with 802.11n MIMO antennas.[2] We found that when S repeatedly sent packets to R, the RSS values (RSS_1 and RSS_2) measured on R were not constant even when the distance was fixed. Instead, they fluctuated in a typical Gaussian distribution consistent with Equation 3. To improve the reliability of our scheme, we let S send a sufficient number of packets when it is close to each antenna on R, and let R calculate the mean of the RSS ratios of these packets. We also observed that when either R or S was moving, the variation of the RSS values was large. Therefore, to prevent a faraway attacker from causing a large RSS ratio by inducing a large variation of the RSS values, our scheme sets a maximum threshold for the standard deviation of RSS ratios.

[2]It runs Fedora Linux with a modified kernel so that the RSS value of each antenna can be read separately.

1. Initialization: The sender S starts to send UDP packets at constant interval, while the receiver R reads RSS_1 and RSS_2 (RSS measured on Antenna 1 and 2, respectively) of these packets and calculates the corresponding RSS ratio $r = RSS_1 - RSS_2$.
2. The user places S very close to the first antenna of R.
3. The user places S very close to the second antenna of R.
4. Pairing succeeds when R observes a sufficient number of consecutive r values whose mean is greater than a positive threshold r_H and whose standard deviation is smaller than a threshold δ_t, and then a sufficient number of consecutive r values whose mean is smaller than a negative threshold r_L and whose standard deviation is smaller than a threshold δ_t.

Figure 2. Basic device pairing scheme

Figure 2 shows our basic pairing scheme. In this scheme, the sender S is required to send UDP packets to the receiver R with fixed interval. Since the only useful information to R is the RSS values, which R measures when receiving the physical preamble of these packets, the payload in the UDP packets is of no use to R.

3.3 Dealing with RSS inaccuracy

The basic scheme is simple and follows the Log-normal shadowing model in Section 2.1. However, this scheme relies on the assumption that the RSS values read from the device driver are linear to the real RSS values. Unfortunately, the RSS values provided by the driver can be distorted due to several factors. We discuss these factors and describe how we eliminate or mitigate them.

3.3.1 RSS Saturation

The *RSS* value reported by the wireless driver (Intel iwlwifi) is an integer in the range $[-95, -10]$. This is usually much smaller than the dynamic range of the actual received

signal strength. As we moved the sender from a few meters away to closer to the receiver, at first we observed a continuous increase of *RSS*. Then, RSS stopped increasing around the value -10. We conjecture that the *RSS* value reported by this driver saturates at the upper bound of -10.

To overcome this problem, we can reduce the transmission power of the sender. But if we reduce the transmission power too much, we risk saturating RSS at its lower bound. To probe for the best power level, our scheme requires the sender to transmit a sequence of packets using different power levels[3] during the initialization stage of the protocol. The receiver then chooses the power level at which the received packets have the maximum RSS ratio and notifies the sender. Then, the sender will temperarily tune it transmission power to that level before it transmits the subsequent packets for RSS measurement.

3.3.2 Automatic Rate adaptation

Another undesirable artifact that affects RSS measurement is automatic rate adaptation. It allows a Wi-Fi device to automatically select the optimal data rate for the current wireless channel conditions.

Data rate change may trigger the change of the physical layer preamble modulation scheme, which will affect the *RSS* values. For example, 802.11g uses the OFDM modulation scheme when the data rate is 54Mbps. When the data rate is decreased to 11Mbps or lower (5.5M, 2M or 1M), it begins to use CCK, the modulation scheme for 802.11b. Switching between modulation scheme can causes a large variation in reported *RSS* values, and make our scheme less stable. In a multiple antenna system such as 802.11n, the automatic rate adaptation feature might even change the transmission antenna. This will completely defeat our scheme. Therefore, the sender must disable automatic rate adaptation when sending the RSS measurement packets.[4]

3.4 Key generation

The basic protocol in Figure 2 authenticates the sender, but it does not generate a shared secret key for further communication. [12, 28] provide approaches to derive a shared key from the characteristics of the wireless channel.

Alternatively, we could use cryptographic techniques to derive a shared secret. Note that key generation does not affect the device pairing scheme in Figure 2 and in fact can proceed in parallel with device pairing. This is because the device pairing scheme only measures the RSS value in the preamble of each packet while key generation uses the payload of the packet.

We propose a straightforward key generation protocol, where the sender receives a public key from the receiver, chooses a shared secret key, encrypts the key with the receiver's public key, and sends the encrypted key to the receiver. The receiver then decrypts the key. Since we are only concerned with one way authentication (the receiver authenticates the sender), there is no need to verify the receiver's public key.

3.5 Final Protocol

Our final protocol integrates both device pairing and key generation, as shown in Figure 3.

1. The user moves the sender S very close to the first antenna on the receiver R and starts the protocol (e.g., by pressing a real or virtual button on S).
2. **S→R:** $PairRequest()$. S sends a pairing request to R.
3. **R→S:** $PairResponse(K_R)$. R responds with its public key K_R.
4. **S→R:** $PowerQuery(i,n)$, where $i = 1,\ldots,n$ and n is the number of power levels. S sends a sequence of packets from the lowest to the highest power levels.
5. **R←S:** $PowerResponse(l)$. After receiving all the n power query packets, R responds with the best power level l that maximizes $r = RSS_1 - RSS_2$.
6. **S→R:** $RSSMeasuring(E_{K_R}(k))$. S generates a random session key k and encrypts it with R's public key K_R, and continually sends the copies of the encrypted session key to R at fixed interval. Meanwhile, the user moves the S from nearby the first antenna on R to nearby the second antenna on R.
7. **R←S:** $Success()$. R examines the RSS values of all packets (containing the encrypted session key) received at both its antennas. If R detects a sufficient number of consecutive packets whose r's mean is above a threshold r_H and whose r's standard deviation is below a threshold δ_t, then R decides that the sender is nearby R's first antenna. Similarly, R detects if the sender is then nearby R's second antenna. After R verifies both these conditions, R replies with a success message.

The protocol runs above the MAC layer of the network stack. All messages except *PowerQuery* and *RSSMeasure* need reliable transmission, i.e. a message needs to be repeated if it is lost.

[3] A typical driver provides 15 different transmission power levels from 1dbm to 15dbm

[4] Our scheme does not require the sender to interfere with automatic rate adaptation or transmission power during normal wireless communication after device paring.

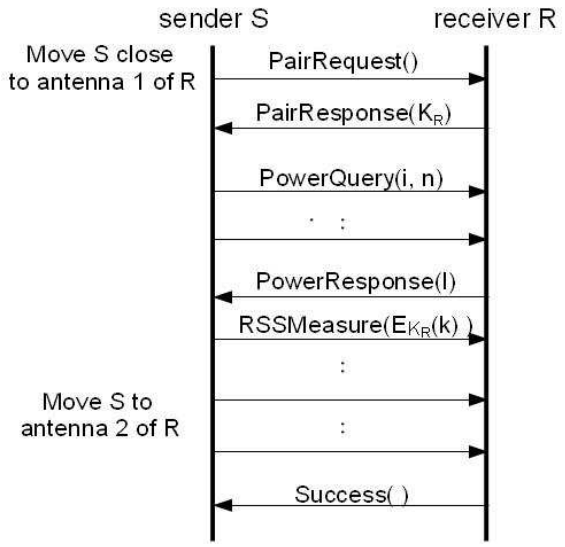

Figure 3. Messages in the final protocol

4 Experiments and results

4.1 Setup

Our experimental system consists of a receiver and a sender, where the sender wishes to be paired with the receiver.

Receiver The receiver is a Dell E5400 laptop running a modified Fedora Linux kernel version 2.6.29-rc5-wl based on the wireless-testing tree. The laptop has an integrated 802.11n Intel Wi-Fi Link 5300 wireless card, and is equipped with three internal antennas. We did not use any of the 802.11n-specific functions on the card – all we needed is the ability to read the RSS values on each antenna individually. We modified the wireless device driver, the kernel-to-user space communication library (*radiotap*), and tcpdump to read the *agc* and RSSI values of each frame received by Antenna 1 and 2, respectively. RSS is computed as:

$$RSS = RSSI - agc - OFFSET$$

where $OFFSET$ is 44, a constant set by this Wi-Fi module, and *agc* (*automatic gain control*) is variable for each packet.

Sender The sender is also a Dell E5400 laptop. Two of its antennas are disabled in the driver, and all the data packets are sent via an external antenna connecting to it.

Antennas We conducted our experiments on the following four types of antennas. In addition to the build-in antenna on the laptop, we also used three types of external antennas, which can be connected to the built-in Wi-Fi card via its IPX/U.fl connectors. Note that our scheme requires no external antennas. The reasons for using external antennas in this experiment are: (1) to measure the impact of the distance between the two receiving antennas on the RSS ratio since we cannot vary the distance between the internal antennas; and (2) to evaluate whether our scheme works on different antennas.

- **Type 1**: These are the internal antennas in the Dell E5400 laptop. After disassembling the laptop, we found that Antenna 1 is fixed at the top left of its LCD screen frame while Antenna 2 is at the top right of the screen frame. We did not use Antenna 3.

- **Type 2**: These are Wi-Fi antennas for laptop mini PCI cards with 61cm (2 feet) IPX/U.fl cables.

- **Type 3**: These are 5 dBi omni-directional Wi-Fi antennas for access points. Each of them has a RP-SMA male interface. We connected them to the laptop using 60*cm* RPSMA female to IPX/U.fl cables.

- **Type 4**: These are 60*cm* RP-SMA female to IPX/U.fl cables, which we used to connect Type 3 antennas to the laptop. Here we used these cables directly as antennas. We tried this type of antenna because on some mobile devices, such as Openmoko freerunner smartphone, the antenna socket is used as a default antenna. Although they allow users to attach external antennas, few users do.

Environment The measurements reported in the rest of this section are from experiments conducted in our lab in a typical computer science building. The lab has several active WiFi access points. We also purposely turned on a microwave oven in a nearby kitchen during the experiment to test how well our scheme tolerates interference. We also repeated the RSS measurements outdoor and found no significant difference from the indoor measurements.

RSS measurement During all the following experiments, we measure RSS values as follows. First, we disable all but one antenna on the sender, so that only one antenna is used to send all the packets. We associate the sender with the receiver in ad-hoc mode, i.e., packets travel from the sender to the receiver directly without going through a base station. Both the sender and the receiver are stationary. The packets are ping packets with 10ms interval. To eliminate the Gaussian noise in the Log-normal shadowing model, we always read RSS from 100 consecutive packets and calculate their mean.

4.2 Effect of distance on RSS

Based on Equation 3, the average RSS value should be a logarithmic function of distance d between the sender and receiver antennas as follows:

$$RSS = P_0 - 10\alpha log_{10}(\frac{d}{d_0}) \quad (4)$$

where P_0 is the RSS value at unit distance d_0.

However, the Log-normal model usually applies when d is much larger than the size of the antennas. In our scheme, when the sender is very close to the receiver, d could be as small as less than $1cm$. So we wish to evaluate how well Equation 4 approximates RSS values when d is small.

During the evaluation, we tried to rule out other factors that may affect RSS. For instance, we always aligned the sending and receiving antennas. We set the sender to use the lowest transmission power $tx = 1dBm$ and disabled the automatic rate adaptation feature. For antenna pairs 1, 2, and 4, we measured RSS values at various distances up to 10cm to avoid the multipath effect. However, since antenna pair 3 has a much larger gain, their RSS is saturated when their distance is smaller than 2cm, so we measured their RSS at distances ranging from 2cm to 30cm. The result (Figure 4) shows that the logarithmic relationship in Equation 4 still approximates the measured RSS values vs distance where the path loss exponent α falls in the range [1.057, 1.365]. P_0 is related with the gain of each antennas pair. It is measured as -11.15, -19.71, -3.59, and -43.21 for antenna pairs 1, 2, 3, 4 respectively.

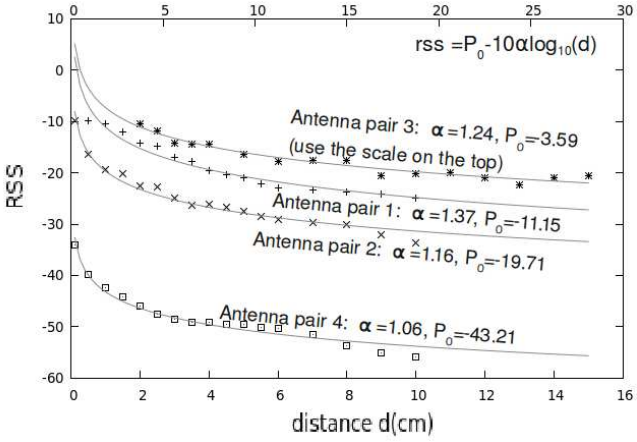

Figure 4. Logarithmic relationship between RSS value and the sender-receiver distance

4.3 Antenna gains

To show that the RSS ratio is independent of antenna gain, we read RSS values when the packets were sent with different transmission power. Our experimental results indicate that the RSS value is a linear function of the transmission power for different antenna distance:

$$RSS(d) = rss_0(d) + tx$$

where tx is the transmission power of the sender measured in dBm and $rss_0(d)$ is the measured RSS value when the sender uses the base transmission power $tx = 0dBm$. We can use $rss_0(d)$ as a gain indicator of the antenna pair at distance d.

Figure 5 plots the RSS values when packets are sent at various transmission power for each antenna pair. The distance between the sender and receiver was fixed at $10cm$. A very small distance tends to cause RSS values to saturate when the transmission power increases, while a very large distance could introduce more interference from the environment, such as the multipath effect (Section 6.1.2). The figure also shows that the gains of the four antennas pairs are ordered as $Type3 > Type1 > Type2 > Type4$. This is consistent with the order of P_0 values measured in the experiment in Section 4.2.

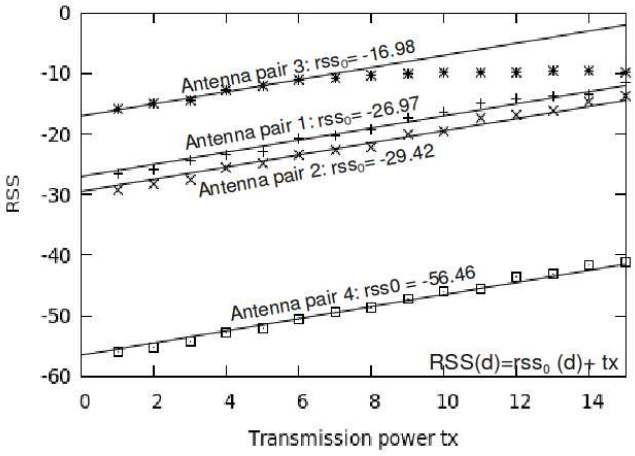

Figure 5. The linear relationship between RSS value and the transmission power

4.4 RSS saturation

Figure 5 shows that RSS on antenna pair 3 no longer increases when the transmission power tx increases beyond 8. This is due to RSS saturation described in Section 3.3.1. To investigate how much RSS saturation can affect our scheme,

especially when the antenna distance d is small, we observed the $RSS - tx$ relationship by different d. Figure 6 demonstrates RSS saturation observed on antenna pair 2. It indicates that RSS saturation occurs with smaller transmission power when the distance d decreases. For instance, when the distance is $3cm$, RSS saturates when $tx > 13dBm$. But when the distance is reduced to $1cm$, RSS saturates when $tx > 6dBm$.

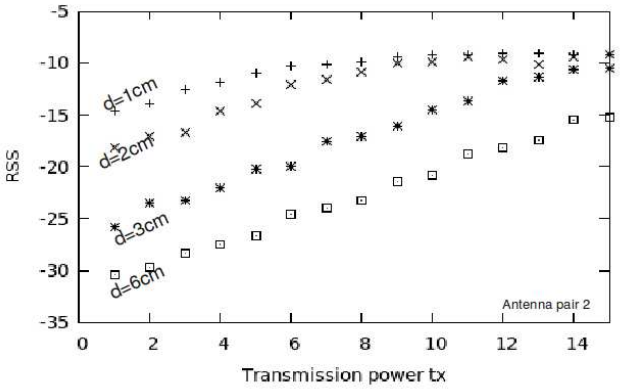

Figure 6. RSS saturation with different sender-receiver distance on antenna pair 2

4.5 Antenna alignment

Because our scheme prefers a high RSS ratio r when the sender authenticates itself to the receiver, users are expected to identify two spots where $\frac{d_1}{d_2}$ is minimum, where d_1 is the distance between the sender antenna and the nearer receiver antenna, and d_2 is that between the sender antenna and the farther receiver antenna. Assume the perpendicular distance from the antenna to the device surface is b_r and b_t for receiver and sender, respectively, the minimum d_1 is $b_r + b_t$. We called the sender and the receiver are *aligned* in this case. In reality, the user may not be able to align the sender with the receiver perfectly. This misalignment would adversely affect RSS because it increases the distance between the sending and receiving antennas and thus decreases the RSS value. Let x be the distance between the current location of the sender and its ideal aligned location with the *dominant* antenna (the antenna on the receiver that the sender should be aligned with). The theoretical RSS value read from the dominant antenna should be:

$$RSS = P_0 - 10\alpha log_{10}\frac{\sqrt{d^2+x^2}}{d_0}$$

We evaluated how much our scheme tolerates the misalignment between the sender and the dominant receiving antenna. We conducted experiments using antenna pair 2 and 4. We set d to be 2cm and measured RSS at different x.

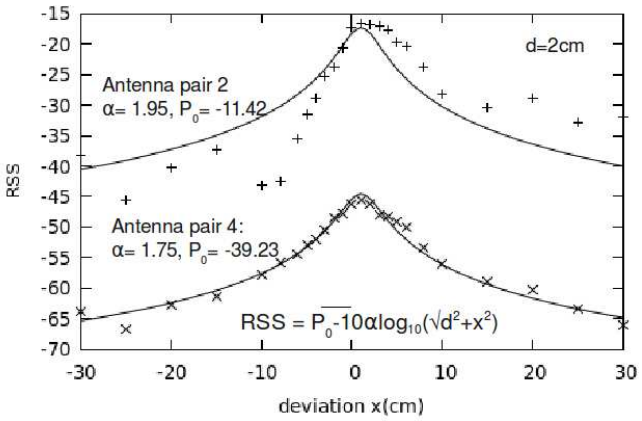

Figure 7. The relation between the RSS value and the offset

Figure 7 shows that RSS value is insensitive to misalignment when the misalignment is small ($< 1cm$), but the effect becomes noticeable when the misalignment increases. To avoid large misalignment, the devices could mark the locations of their antennas on their surfaces. Moreover, when we select the thresholds r_L and r_H in the device pairing protocol in Section 3.2, we need to take into account how much we tolerate antenna misalignment.

4.6 Distance between the receiving antennas

To take advantage of the antenna diversity, most laptops have their antennas mounted on the corners of their LCD frames or the two sides of their bodies. Therefore, the receiving antennas are usually more than 20cm away from each other. However, handheld mobile devices are usually smaller than 20cm. To investigate whether our scheme allows handheld mobile devices with multiple antennas to be used as receivers, we conducted the following experiment. We used a Dell E5400 laptop connected with two external antennas (Type 2) as the receiver. We chose an Openmoko Freerunner smartphone as the sender and placed it only 1cm away from one of the two external antennas on the receiver. Similar to all the previous experiments, the sender phone established an ad hoc connection with the receiver and continually sent ping packets with an interval of $10ms$. The transmission power was tuned in advance to avoid RSS saturation. We measured the RSS ratios of 100 consecutive packets when the sender was aligned with the left receiving antenna and right receiving antenna, respectively. We repeated this measurement for different distances between

the two external antennas on the receiver: 10cm, 20cm, and 30cm. Table 1 shows the mean and standard deviation of the RSS.

L (cm)	RSS on Antenna 1		RSS on Antenna 2	
	$\bar{r}$	δ	$\bar{r}$	δ
10	13.72	0.86	-13.77	0.42
20	14.69	0.46	-16.90	0.44
30	20.49	0.52	-18.04	0.31

Table 1. Measured RSS values under various distance between two receiving antennas. L is the distance between two receiving antennas.

The experiment indicates that even when the distance between the two receiving antennas decreases to $10cm$, which is a reasonable lower bound of the longest dimension of many handheld devices, the RSS ratio is still large enough (13.72) to be usable in our scheme.

5 Prototype

We developed a prototype of our device pairing scheme to evaluate the practicality of our scheme.

5.1 Set up

Sender The sender is an Openmoko Free Runner smartphone running Linux. It has a single antenna and a Wi-Fi module.

Receiver The receiver is a Dell E5400 laptop running a modified Fedora Linux kernel version 2.6.29-rc5-wl based on the wireless-testing tree. The laptop has an integrated 802.11n Intel Wi-Fi Link 5300 wireless card, and is equipped with three internal antennas, although our prototype uses only two of these antennas. We marked the locations of the antennas on the surface of the laptop.

Pairing procedure The sender and receiver share no prior secret. The receiver continuously runs a pairing server program. The user pairs the sender with the receiver via the following steps:

1. The user places the sender next to the left antenna of the receiver.
2. The user starts our pairing program on the sender. Then, the program sends a sequence of packets to the receiver.
3. After receiving a sufficient number of measurement packets that satisfy the pairing criteria below (usually within a few seconds), the receiver notifies the user via a beep. Then, the user places the sender next to the right antenna of the receiver.
4. After receiving another sufficient number of measurement packets that satisfy the pairing criteria , the receiver notifies the user of a successful pairing via multiple beeps.

Pairing criteria The receiver decides whether the sender is close by measuring the RSS ratios (i.e., the ratio between the RSS on the left and right antennas) of the RSSIQuery packets from the sender. In both Step 3 and 4 above, the receiver places the packets into a FIFO queue of size 40 and checks if the RSS ratios of all the packets in the queue satisfy:

- The mean $\bar{r}$ of the RSS ratios exceeds a threshold ($\bar{r} > r_H$ in Step 3, and $\bar{r} < r_L$ in Step 4).
- The standard deviation of the RSS ratios is smaller than a threshold δ_t.

The sender sends about 40 packets per second. To be robust against signal interference, the receiver keeps computing the above pairing criteria (i.e., whenever a new packet arrives, it is inserted into the FIFO queue of 40 packets, and the receiver reruns the pairing criteria on the queue) for 20 seconds until the pairing succeeds or the receiver times out.

In the above criteria, r_H and r_L depend on the distance d between the two antennas on the receiver. In our prototype system, $d = 26cm$. Based on our experiment in Section 4.2, the RSS ratio from a nearby sender should be larger than 16. We set $r_H = 11$ and $r_L = -11$ to leave some room for antenna misalignment. We set $\delta_t = 0.6$ based on our observations.

5.2 Security evaluation

We evaluated the security of our prototype by trying to authenticate the sender at different distances from the receiver:

1. Close-range: The sender is placed immediately next to the receiver, e.g., when the user places the Openmoko phone next to the screen of the laptop where an antenna is located.
2. Mid-range: The sender is between 20 – 100cm away from the receiver.
3. Long-range: The sender is more than 100cm away from the receiver.

Distance range	Close-range	Mid-range	Long-range
Success Rage	90%	0%	0%
Failure Rate	10%	100%	100%
Max Mean RSS Ratio	15.62	6.35	3.43

Table 2. Authentication Accuracy. Authentication in each distance range is tried 20 times.

We estimate that an attacker cannot reasonably place her device within 100cm from the receiver without alarming the receiver's owner. However, we also evaluated if the attacker can successfully pair with the receiver should she get as close as 20cm from the receiver. For each distance range, we attempted device pairing 20 times. In the experiments for mid-range and long-range, at each device pairing attempt we randomly placed the sender within that range.

Table 2 shows that the success rate for close-range ($< 20cm$) device pairing is 90%. The two failed pairings in this range happened when we failed to align the sender's antenna with those of the receiver. By contrast, our prototype rejected all the device pairing attempts when the sender was in either mid-range ($[20cm, 100cm]$) or long-range ($> 100cm$). Table 2 also shows the maximum mean RSS ratios of the packets in different distance ranges. When the sender is in close-range, the maximum mean RSS ratio is above 15, while this ratio drops to 6.35 and 3.43 when the sender is in mid-range and long-range, respectively.

5.3 Timing evaluation

We measured the time that it takes the user to complete a successful device pairing. From the user's perspective, the pairing consists of three steps:

0. Move the sender to the left antenna of the receiver.
1. Click a button on the sender to start the pairing, and wait for the receiver to beep (indicating that the receiver has received enough measurement packets that satisfy its criteria).
2. Move the sender to the right antenna of the receiver, and wait for the receiver to beep multiple times (indicating that the pairing has succeeded).

We did not measure the time for Step 0 because it is irrelevant to the design of our protocol. We measured the time for Steps 1 and 2 shown in Table 3.

The user took an average of 5.29s to complete Step 1. A large portion of this time (3.67s) is spent on waiting for the sender to send 15 power query packets. Currently, to send a packet at a different power level, our prototype implementation in the sender needs to execute the *iwconfig* command, which takes about 200ms each time. To reduce the time spent on Step 1, we could use more efficient ways to adjust the power levels of packets, or to find the best power level more efficiently than a linear search (e.g., a binary search between all the power levels). The user took an average of 6.35s to complete Step 2. Compared to Step 1, the variation in the time for Step 2 is larger because it includes the time for the user to move the sender from the left antenna to the right antenna of the receiver. The average time for step 1 and 2 in device paring is 11.64s. This is faster than or comparable to most other wireless device pairing schemes[10]. Moreover, this scheme requires no user decision and has a fail-safe default: if the user fails to follow the simple procedure, the pairing simply fails.

Distance range		$< 20cm$	$[20cm, 100cm]$	$> 100cm$
Time for Step 1	Ave	5.29s	6.53s	Timeout
	Min	5.06s	5.33s	Timeout
	Max	5.52s	7.72s	TImeout
Time for Step 2	Ave	6.35s	Timeout	Timeout
	Min	2.77s	Timeout	Timeout
	Max	17.59s	Timeout	TImeout
Total	Ave	11.64s	Timeout	Timeout

Table 3. Authentication Time.

6 Security and usability

6.1 Security

6.1.1 Success probability of random attacks

We calculate the probability of successful attack if a faraway attacker just randomly picks two locations during the device pairing. Assume RSS ratio r induced by the attacker follows Gaussian distributions $N(\mu_H, \sigma_H^2)$ and $N(\mu_L, \sigma_L^2)$, the means of n RSS ratio $\hat{\mu}_H$ and $\hat{\mu}_L$ should follow $N(\mu_H, \frac{\sigma_H^2}{n})$ and $N(\mu_L, \frac{\sigma_L^2}{n})$, respectively. The sample variances $\hat{\sigma}_H^2$ and $\hat{\sigma}_L^2$ have distributions proportional to chi-square as $\frac{\sigma_H^2}{n} \cdot \chi^2(n-1)$ and $\frac{\sigma_L^2}{n} \cdot \chi^2(n-1)$, respectively.

Let the threshold of the mean as μ_t and that of the variance as σ_t^2, the attacker's device will be paired only when $\hat{\mu}_H$ is larger than its threshold and $\hat{\mu}_L$ is smaller than its threshold, and the sample variances $\hat{\sigma}_H^2$ and $\hat{\sigma}_L^2$ are smaller than their threshold. Note that for normal distribution, the sample mean and sample variance are independent. Therefore, $\hat{\mu}_H$, $\hat{\sigma}_H^2$, $\hat{\mu}_L$, and $\hat{\sigma}_L^2$ are independent of each other. The probability of a successful attack is:

$$
\begin{aligned}
P_a &= Pr(\hat{\mu}_H > \mu_t) \cdot Pr(-\hat{\mu}_L > \mu_t) \cdot Pr(\hat{\sigma}_H^2 < \sigma_t^2) \\
&\quad \cdot Pr(\hat{\sigma}_L^2 < \sigma_t^2) \\
&= Q(\tfrac{\mu_t - \mu_H}{\sigma_H/\sqrt{n}}) \cdot (1 - Q(\tfrac{-\mu_t - \mu_L}{\sigma_L/\sqrt{n}})) \\
&\quad \cdot \tfrac{\gamma((n-1)/2, (n\sigma_t^2)/(2\sigma_H^2))}{\Gamma((n-1)/2)} \cdot \tfrac{\gamma((n-1)/2, (n\sigma_t^2)/(2\sigma_L^2))}{\Gamma((n-1)/2)}
\end{aligned} \tag{5}
$$

where $Q(x)$ is the Q-function computing the right-tail probability for normal random variables, $\gamma(k,x)$ is the lower incomplete Gamma function, and $\Gamma(k)$ is the Gamma function.

Using the parameters set for our prototype system, the probability of successful attack is less than 10^{-15}.[5]

6.1.2 Attacks leveraging multipath effect

In our experiments on the prototype system, the sender occasionally passed the first phase of the pairing scheme when it was more than $20cm$ away from either antenna on the receiver. This indicates that the receiver has a non-negligible probability of receiving a sufficient number of packets that have stable and large RSS ratios even when the sender is not in its close proximity. This is inconsistent with the result in Section 6.1.1 and is mainly caused by the multipath effect.

Due to the multipath effect, the received signal can become stronger or weaker if there is a constructive or destructive superposition of the signals coming from different paths, respectively. In an indoor environment, multipath effect is often caused by reflection on the surface of the floor, ceiling, wall, furniture, and even people. Using our scheme, when the sender is paired with a nearby receiver, the multipath effect will unlikely affect the RSS values significantly because the sender is very close to the receiver. However, a faraway attacker could take advantage of the multipath effect to cause a large RSS ratio measured at the two antennas on the receiver, therefore breaking our scheme.

We use the following simplified two-path model to show how much multi-path effect can affect our scheme. Assume the signal strength is determined by only two dominating paths: a straight path from the sender to the receiver, and a path reflected on the ground, as shown in Figure 8. Let H_S be the height of the sending antenna, H_R be the height of the two receiving antennas, L be the distance between two receiving antennas, L_{D1} and L_{D2} be the length of two direct paths, and L_{R1} and L_{R2} be the length of two reflect paths. We also define Γ as the reflection coefficient, which depends on the polarization of the radio wave and the reflection angle. According to [25], we have:

$$
r = 10log_{10}\frac{(L_{R1} \cdot cos\Delta\theta_1 + \Gamma L_{D1})^2 + (L_{R1} \cdot sin\Delta\theta_1)^2}{(L_{R2} \cdot cos\Delta\theta_2 + \Gamma L_{D2})^2 + (L_{R2} \cdot sin\Delta\theta_2)^2}
$$

[5] We compute the probability in Matlab, which gives answer 0. Since Matlab supports 10^{-15} precision, we conclude that the probability is less than 10^{-15}.

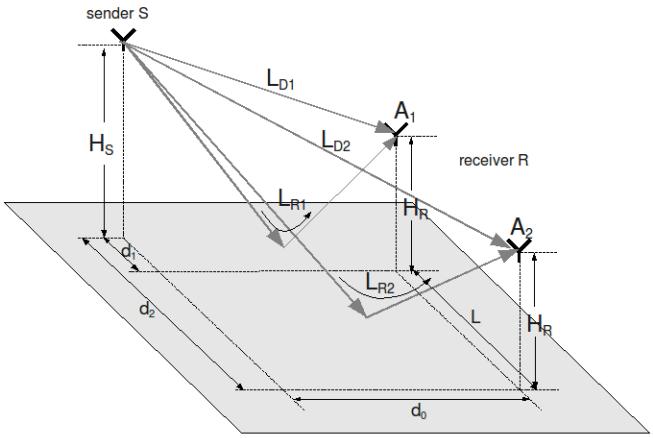

Figure 8. Two paths model

where $\Delta\theta_1$ and $\Delta\theta_2$ are phase delays, which are determined by L_{R1}, L_{D1} and L_{R2}, L_{D2} respectively. In theory, it may be possible for an attacker to choose an appropriate path lengths and reflection angle to manipulate the value of $\Delta\theta$ and Γ to make r a large value.

However, to launch this attack successfully, the attacker's sender must be in the line of sight of the receiver, and the attacker must identify two proper locations and calculate the lengths and reflection angles of all transmission paths to the receiver. This is very challenging, if not infeasible, as the attacker would need to measure the locations, geometries, and surface properties of all the objects in the environment. Even if this daunting task were feasible, we can mitigate this attack by incorporating frequency hopping into our protocol. With frequency hopping, the attacker's optimal path lengths in different channels are most likely different, so it would be very difficult to find a path length that keeps the RSS ratio high in all the channels. Incorporating frequency hopping in our scheme is straightforward: instead of using only one channel, S sends *RSSIMeasuring* packets while cycling through all the channels. However, it is not easy to implement frequency hopping on the platform where we implemented our prototype, because it takes substantial time to switch wireless channels from the user space. We believe that this limitation can be overcome by an implementation of frequency hopping in the device driver or the firmware.

Nevertheless, we conducted an experiment to test this idea of frequency hopping. We observed RSS values by placing the sender randomly at locations that are $2m$ or $4m$ from the receiver. Both the receiver and sender are placed at a height of $30cm$ or $1m$. The packets are sent via different 802.11a channels. Figure 9 shows that at each location, the RSS ratio can be quite large on packets sent from certain channels. The largest observed $r(t)$ is about 10 where the

sender is 4m away from Antenna 1. However, at each location, the r values on different channels vary considerably, and their mean values are close to 0. By comparison, we also measured r on different channels when the sender is very close to the receiver (at 2cm) and show it in Figure 9. This curve shows that r is relatively stable on different channels. This experiment indicates that frequency hopping is able to mitigate the threat of a faraway attacker who tries to exploit multipath effect.

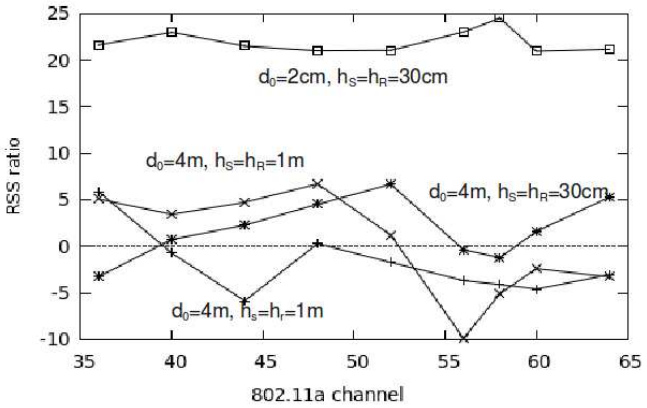

Figure 9. Using frequency hopping to defeat attacks using multipath effects

6.1.3 Beam-forming attack

In theory, a powerful faraway attacker may attempt to form special beams to cause a large difference between the RSS values at the two receiving antennas. In practice, however, this attack would be very difficult, if not impossible. The beam forming attacker would need a narrow-width main lobe (beam). The lobe width is inversely proportional to the size of the antenna arrays. Since the distance between the two antennas on the receiver is usually small (typically less then 1 meter), the attacker would need a very large antenna array, which in many situations would raise suspicion. Moreover, when the attacker is far from an indoor receiver, multipath effect would likely distort the intended beam too much to achieve the required differential RSS on the two antennas on the receiver, unless the attacker knows the accurate channel state information (CSI) from its antenna to the receiver's antennas. This information is measurable only at the receiver's antennas, but in our protocol (Section 3), the receiver never sends the measured CSI to the sender. Moreover, the attacker cannot even get an accurate estimation on the CSI based on its observation of the reverse channel (the channel from the receiver to the sender) because of the following two reasons. First, our protocol does not require the receiver to send messages via both its antennas. Therefore, the attacker cannot measure the reverse CSI of both the channels. Second, even if the receiver sends signals from both its antennas, the CSI of the reverse channel may be different from that of the forward one because reciprocity may not hold due to non-symmetric noise.

6.1.4 Time-of-check to time-of-use attack

Since RSS is measured in the physical layer preamble while the session key is carried in the frame, an attacker might try to attack Step 6 in the protocol described in Section 3.5 by sending his encrypted session key when the receiver begins to receive the frame. However, this attack is nearly impossible. First, it is very hard for the attacker to time his frame at the moment just after the receiver has received the preamble from another user. For 802.11a and 802.11g, a symbol lasts 4 microseconds, including an 800 nanosecond guard interval. If the attacker wants his first symbol to arrive at the receiver just after the genuine sender's preamble, she must be able to control his transmission delay within one microsecond. However, it is nearly impossible to control the transmission delay in such fine granularity. Even if the attacker could achieve this, his frame would collide with the genuine sender's frame, which would cause the receiver to drop the frame. Although the attacker can launch an DoS attack this way, he could launch DoS more easily by jamming, which is out of the scope of this work.

6.2 Usability

We provide an empirical evaluation of the usability of our scheme. We leave formal usability study for future work.

Resilience against interference One advantage of our scheme is its ability to resist interference. Many device pairing schemes require the use of auxiliary "out-of-band" channels, such as acoustic [16], that are subject to environment interference. By contrast, our scheme uses auxiliary information (RSS) in the existing wireless channel. Therefore, it inherits the interference-resistance properties from the wireless channel. All our experiments were conducted in a typical computer science building with several APs and even a microwave oven.

Avoiding user errors Our scheme requires no decision from the user. All the user has to do is to move the sender from one antenna to another on the receiver. If the user fails to follow the instruction, it will result in pairing failure, but not insecure paring. This provides fail-safe default.

Ease of use The relatively challenging part of our scheme for the user is to align the antennas of two devices; failure to align may result in device pairing failure. Similar efforts are required by other device pairing schemes. [14], for instance, requires the user to align the camera of one device to the screen of the other. In [16], users have to move one device along the direction of the other one. Both the above schemes require users to move one device in a 3D space. By contrast, our scheme only requires the user to move the sender in the 2D surface of the receiver. When the locations of the antennas are marked on the surface of the receiver, this becomes a simple task.

Pairing time The experiments showed that it takes an average of 11.64s to pair the devices in our prototype, which is faster than most schemes tested by [10]. Although we (the authors) conducted the tasks ourselves in our experiments, we expect to observe a similar pairing time on ordinary users because our scheme requires a simple movement and no user decision.

Versatility Our scheme requires the receiver to have two antennas separated by a reasonable distance. For example, when a user pairs a smartphone with his laptop, only the laptop needs two antennas. Fortunately, most current laptops, including the ones without 802.11n modules, use multiple antennas to take advantage of antenna diversity. Additionally, there is a trend towards embedding 802.11n Wi-Fi in handheld mobile devices (at least the chip manufacturers are ready [22]). These devices can take advantage of our scheme for secure pairing as well.

7 Related work

Wireless device paring With the proliferation of mobile wireless devices, researchers have proposed many schemes for secure devices pairing. These schemes rely on trusted side-channels to pair the devices with each other. Earlier approaches required the user to be the channel, i.e., they asked the user to enter the shared secret into the devices, but these methods suffer from apparent usability and security problems discussed in the introduction. To avoid these problems, researchers have since proposed newer schemes to use the extra sensory and output hardware present on many wireless devices as the trusted communication channel [10]. We can divide these schemes into two categories in terms of user interaction: (1) those that require the user to decide whether the device pairing succeeds by comparing visual [26] or audio [6, 23] output; (2) those that require the user to initiate the device pairing but let the device decide whether the pairing succeeds via the reading from its sensors (e.g., a camera [14, 21], microphone [17], or accelerometer [8, 13]). Our proposed mechanism falls into the latter group, which has the advantage that it is less fallible to user errors since users do not need to decide the success of the authentication. However, while most wireless devices have some sensory or output hardware, two arbitrary devices may not have the required hardware to provide secure authentication. For example, [14, 21] applies only to devices with cameras. By contrast, our scheme uses the primary communication channel of wireless devices for authentication and thus requires no extra hardware. Although our scheme requires the receiver to have at least two antennas, multiple antennas are increasingly common as wireless device manufacturers are embracing the MIMO (Multiple-Input Multiple-Output) technology. (Note that our scheme does not require the sender to have more than one antenna.)

Distance bounding protocols Distance bounding protocols [3] are cryptographic protocols that establish an upper-bound on the physical distance between two parties by timing the delay between sending out a challenge bit and receiving the response bit. They have been implemented for various wireless protocols [27, 7], but all of them rely on a rapid bit exchange and require precise clocks to measure the delay between messages traveling at the speed of light. Since electromagnetic waves propagate over 30cm in 1 nanosecond, the requirement for such high precision clocks is unsuitable for consumer electronic devices. In fact, many existing implementations [4, 19] are based on ultrasound ranging techniques.

Our scheme can reliably determine the proximity of the pairing devices without requiring high-precision clocks; instead, our scheme measures the ratio between the receiving signal strength at multiple antennas.

Received signal strength Researchers have used Received signal strength (RSS) to detect the sybil attack in wireless sensor networks [5], where they used the RSS ratio between different monitors to locate users. There are many differences between our scheme and theirs, the biggest one being the purpose: our scheme is for deciding whether the sender is close to the receiver, while their scheme is for deciding if the packets with different identities come from the same location. As a result, our scheme enjoys the following advantages. (1) Our scheme needs only two antennas, while in theory their scheme requires at least four to achieve an accurate localization. (2) The precision of their scheme is in meters, while our scheme can reject attackers that are merely 20cm away. On the other hand, our scheme also has to overcome extra challenges: since the user has to hold and move the sender during device pairing and the sender is very close to the receiver, our scheme is more susceptible to radio signal interference and variation. We used statistics and power level probing to overcome this problem. Finally, since our goal is device pairing, we also need to design a

protocol that derive a shared secret.

Hu and Evans use directional antennas to verify proximity to prevent wormhole attacks [9]. By contrast, our scheme does not require directional antennas. In fact, our scheme prefers omnidirectional antennas because they avoid the problem of misalignment between the sending and receiving antennas. Most consumer wireless devices also prefer omnidirectional antennas because the users would not have to orient the devices in certain directions.

8 Conclusion

We have designed a reliable secure device pairing scheme based on device proximity. The scheme takes advantage of multiple antennas built in many modern wireless devices and leverages a characteristic of wireless channels - the power of the received signal is inversely proportional to some exponent of the distance between the sender and receiver. When a nearby sender is very close to one antenna on the receiver, the receiver can observe a large difference between the power measured on its two antennas, whereas a faraway sender would be unable to induce this large difference. We validated our scheme through theoretical analysis and experimental measurements. We discussed factors that may affect our scheme, including antenna gain, antenna alignment, RSS saturation, dynamic rate adaptation and multipath effects. Finally, we evaluated a prototype of our scheme by pairing an Openmoko Free Runner mobile phone with a laptop using threshold values derived from our measurements. The experiment shows that our scheme is easy, fast, and reliable.

References

[1] Eye-Fi SD card wiki page. `http://en.wikipedia.org/wiki/Eye-Fi`.

[2] Wi-Fi protected setup. `http://en.wikipedia.org/wiki/Wi-Fi_Protected_Setup`.

[3] S. Brands and D. Chaum. Distance-bounding protocols. In *EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology*, pages 344–359, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[4] S. Capkun, M. Cagalj, G. O. Karame, and N. O. Tippenhauer. Integrity regions: Authentication through presence in wireless networks. *Mobile Computing, IEEE Transactions on*, 9(11):1608 –1621, 2010.

[5] M. Demirbas and Y. Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 564–570, Washington, DC, USA, 2006.

[6] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human-verifiable authentication based on audio. In *ICDCS '06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, page 10, Washington, DC, USA, 2006. IEEE Computer Society.

[7] G. Hancke and M. Kuhn. An rfid distance bounding protocol. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. First International Conference on*, pages 67–73, Sept. 2005.

[8] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *UbiComp '01: Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 116–122, London, UK, 2001.

[9] L. Hu and D. Evans. Using directional antennas to prevent wormhole attacks. In *Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, Feb. 2004.

[10] A. Kobsa, R. Sonawalla, G. Tsudik, E. Uzun, and Y. Wang. Serial hook-ups: a comparative usability study of secure device pairing methods. In *SOUPS '09: Proceedings of the 5th Symposium on Usable Privacy and Security*, pages 1–12, New York, NY, USA, 2009. ACM.

[11] J. Krumm and E. Horvitz. Locadio: Inferring motion and location from Wi-Fi signal strengths. In *in First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous*, 2004.

[12] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 128–139. ACM, 2008.

[13] R. Mayrhofer and H. Gellersen. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing*, 8(6):792–806, 2009.

[14] J. McCune, A. Perrig, and M. Reiter. Seeing-is-believing: using camera phones for human-verifiable authentication. In *Security and Privacy, 2005 IEEE Symposium on*, pages 110–124, May 2005.

[15] A. Neskovic, N. Neskovic, and G. Paunovic. Modern approaches in modeling of mobile radio systems propagation environment. *IEEE Communications Surveys*, 3(3):1–12, 2000.

[16] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 1–14, New York, NY, USA, 2007. ACM.

[17] C. Peng, G. Shen, Y. Zhang, and S. Lu. Point&connect: intention-based device pairing for mobile phone users. In *MobiSys '09: Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 137–150, New York, NY, USA, 2009. ACM.

[18] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, New Jersey, 2001.

[19] K. B. Rasmussen, C. Castelluccia, T. S. Heydt-Benjamin, and S. Capkun. Proximity-based access control for implantable medical devices. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 410–419, New York, NY, USA, 2009. ACM.

[20] K. B. Rasmussen and S. apkun. Realization of rf distance bounding. *USENIX '10: Proceedings of the 19th USENIX Security Symposium*, 2010.

[21] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel (short paper). In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 306–313, Washington, DC, USA, 2006. IEEE Computer Society.

[22] A. Shilov. Mobile phones with faster 802.11n Wi-Fi incoming. `http://www.xbitlabs.com/news/networking/display/20091110182814_Mobile_Phones_with_Faster_802_11n_Wi_Fi_Incoming_Research_Firm.html`, December 2009.

[23] C. Soriente, G. Tsudik, and E. Uzun. Hapadep: Human-assisted pure audio device pairing. In *ISC '08: Proceedings of the 11th international conference on Information Security*, pages 385–400, Berlin, Heidelberg, 2008.

[24] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ubiquitous computing (supplement to computer magazine). *Computer*, 35:22–26, 2002.

[25] T. Stoyanova, F. Kerasiotis, A. Prayati, and G. Papadopoulos. Evaluation of impact factors on rss accuracy for localization and tracking applications. In *MobiWac '07: Proceedings of the 5th ACM international workshop on Mobility management and wireless access*, pages 9–16, New York, NY, USA, 2007.

[26] S. Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, pages 309–326, 2005.

[27] S. Čapkun, L. Buttyán, and J.-P. Hubaux. Sector: secure tracking of node encounters in multi-hop wireless networks. In *SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 21–32, New York, NY, USA, 2003. ACM.

[28] K. Zeng, D. Wu, A. Chan, and P. Mohapatra. Exploiting multiple-antenna diversity for shared secret key generation in wireless networks. In *IEEE Infocom 2010*, March 2010.