

# Implementation for Simplifying Bluetooth Device Connection Methods

by

Tehyih Debbie Wan

Submitted to the Department of Electrical Engineering and Computer  
Science

in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

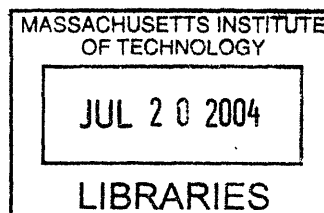
June 2004

© Massachusetts Institute of Technology 2004. All rights reserved.

Author .....  
Department of Electrical Engineering and Computer Science  
May 25, 2004

Certified by .....  
Larry Rudolph  
Principal Research Scientist  
Thesis Supervisor

Accepted by .....  
Arthur C. Smith  
Chairman, Department Committee on Graduate Students



ARCHIVES



# Implementation for Simplifying Bluetooth Device Connection Methods

by

Tehyih Debbie Wan

Submitted to the Department of Electrical Engineering and Computer Science  
on May 25, 2004, in partial fulfillment of the  
requirements for the degree of  
Master of Engineering in Electrical Engineering and Computer Science

## Abstract

This thesis provides a way for users to easily add and remove devices to secure groups, allowing for a more intuitive way of making connections between devices. These groups allow users to seamlessly make connections between their devices once devices are added to a group. Current models for sending data from device to device either involve proprietary cables and manufacturer-specific connectors or complex systems of PINs and passwords to authorize a user/device. In an environment where a user expects to be able to quickly send data between his mobile devices, the time it takes to initialize a connection should not be longer than the time it takes to send the data. The solution proposed in this thesis establishes a model using groups, with group owners and group members. Groups are identified using shared public and private key pairs. The group model establishes implicit trusted relationships between members of a group without having to explicitly authorize a member, simplifying and reducing the number of authenticated links a user must create between his devices.

Thesis Supervisor: Larry Rudolph  
Title: Principal Research Scientist



## Acknowledgments

First and foremost I would like to thank Larry Rudolph, my advisor, for his guidance and inspiration over the past year. Not only has he taught me what it is to do research, but given me the opportunity and flexibility to learn and do what I want. I almost want to apply to do a Ph.D. Really!

Atish, Nancy, Jessica, Angelina, and Albert, I thank you guys for being rocking officemates and listening to my trials and tribulations. I don't think I could have come to work every day and liked it without you guys. Procrastination, anyone?

Albert, thanks for being a great teacher and having the patience not just to help me do something, but to make me learn it as well. You make me want to learn to program.

A huge thank you to my friends, who are always there for me despite those times I get lost in my work. No names because if I forget someone I can't use the excuse that I only had 30 seconds.

Nate, you're my biggest supporter. Thank you for being there through my ups and downs, and for motivating me in both life and work. You push me to understand more about life and myself...and you're always asking me hard questions. Don't ever stop, my brain might get lazy.

Last but not least, and most importantly, thank you Mom and Dad for giving me the opportunity to come to this place and the strength to make it through. Your teaching and your love, and unfaltering support have allowed me to do so much more than a kid could want. Thank you.



# Contents

<b>1</b>	<b>Introduction</b>	<b>13</b>
<b>2</b>	<b>Background</b>	<b>15</b>
2.1	Types of networks . . . . .	15
2.2	Security Systems . . . . .	18
2.2.1	Access Control Lists . . . . .	18
2.2.2	Certificate Authority . . . . .	19
2.2.3	Key Encryption systems . . . . .	19
2.3	Wireless technology . . . . .	20
<b>3</b>	<b>Problem</b>	<b>23</b>
3.1	Too many mobile devices . . . . .	24
3.2	Current network models . . . . .	25
3.2.1	Network with static devices only . . . . .	25
3.2.2	Network with some static devices and some mobile devices . . . . .	26
3.2.3	Network with one static device and many mobile devices . . . . .	27
3.2.4	Network with many mobile devices . . . . .	28
3.3	Current models . . . . .	30
3.4	Requirements for a new system . . . . .	31
<b>4</b>	<b>Solution</b>	<b>33</b>
4.1	Initial idea using a simple group model . . . . .	33
4.2	Group model allowing cross-group membership . . . . .	34

4.3	Groups with no hierarchy and implicit trusted links . . . . .	35
4.3.1	Group Identification . . . . .	37
4.3.2	Validating a Device Name . . . . .	37
4.4	Group Model with Virtual President . . . . .	38
4.5	Group model with 3-tiered Hierarchy . . . . .	39
4.6	The Final Design . . . . .	40
<b>5</b>	<b>Implementation</b>	<b>43</b>
5.1	Use of Bluetooth . . . . .	43
5.2	Technology background . . . . .	44
5.2.1	Making connections . . . . .	44
5.2.2	Authentication . . . . .	46
5.3	Specific Implementation . . . . .	47
5.3.1	Bluetooth User Model . . . . .	47
5.3.2	Design Goals . . . . .	49
5.3.3	Device identification functions . . . . .	51
5.3.4	Group and member validation . . . . .	51
5.3.5	New member services . . . . .	52
5.3.6	Removing members . . . . .	52
5.4	Bluetooth specific implementation trade-offs . . . . .	52
<b>6</b>	<b>Conclusion</b>	<b>55</b>
6.1	Extending functionality to all devices . . . . .	56
6.1.1	The Bluetooth USB Dongle . . . . .	57
6.1.2	Design . . . . .	57
6.2	Human-centric authorization models and usage scenarios . . . . .	59
6.3	Closing thoughts . . . . .	60



# List of Figures

2-1	Bob sends Alice a message that has been signed with Bob's private key. Alice has a copy of Bob's public key and can verify that the message was from Bob by using the public key. . . . .	20
3-1	A mobile user and his networks of devices. . . . .	23
3-2	A network of static devices. . . . .	26
3-3	An environment with a mixed number of static and mobile devices. . . . .	27
3-4	A network with one static device and multiple mobile devices. . . . .	28
3-5	An environment with only mobile devices. . . . .	29
4-1	A simple model for groups showing two groups, club A and club B. Club A has members X, Y, and Z. Club B has members U, V, and W. . . . .	34
4-2	A group model with hierarchies. If club A joins club B, the X, Y, and Z automatically become members of club A. . . . .	35
4-3	A group model with implicit trusted connections (dotted lines) between members of a group. . . . .	36
4-4	A group model with implicit trusted connections where clubs are not allowed join other clubs. B has joined club A, but U, V, and W, which belong in club B, are not automatically trusted. . . . .	36
4-5	When X and Y connect to each other, they create a virtual club, called club A. When Z connects to Y, he is also added to virtual club A . . . . .	39
4-6	When Z and U connect, their respective clubs do not become automatically trusted networks. Rather, a new virtual club is formed that others can join. . . . .	39

4-7	Bob’s PDA belongs to his home network, but also happens to be trusted by his work printer “group.” However, the trust chain does not need to extend such that the cell phone needs to trust the printer. . . . .	40
4-8	A model for 3-tier club hierarchy. Presidents can add executive members. Executive members hold the club private key, and can add regular members. Regular members cannot add others to the group. . . . .	41
4-9	Device X and device Y belong to club A. This diagram shows the token exchange process that occurs when device Y wants to connect to device X. . . . .	42
4-10	Device Y requests to be added to club A. At this point, the devices have already exchanged public keys and determined that they are not in the same group. The user has approved adding Y to club A. . . . .	42
5-1	Bluetooth module components. . . . .	45
5-2	Bluetooth device modes while in the process of connecting. . . . .	46
5-3	PDA and cell phone belong to laptop’s club. PDA and cell phone can connect without additional user authentication because they belong to the same club. . . . .	50
6-1	Block diagram of dongle design. . . . .	57
6-2	Prototype of dongle with two buttons. The “connect” button connects the device on the dongle to the nearest Bluetooth device. The “add/join” button adds a device to the group or joins a device’s existing group. . . . .	58

# List of Tables

2.1 Table showing the categories of wired and wireless networks. . . . . 18



# Chapter 1

## Introduction

As the number of mobile devices increases, so does the need of having a natural way for an owner to authorize connections between them. One natural way is to create groups of devices that mimic the hierarchy of trust between family, friends, and strangers. Current mobile devices are non-intuitive and not simple to connect: they require the user to type in passwords, find cables to connect, or follow arcane instructions to set up secure wireless connections. This thesis creates a way to set up and maintain scalable, secure, groups of mobile devices that require minimal user interaction.

A user should be able to use his mobile device without worrying about how he will get information from one device to another. Interaction should be intuitive, meaning conceptually simple and taking little effort to understand; and easy, meaning the device takes minimal learning to use and operate. If a user wants to send a file from a laptop to a PDA, he should be able to do that without trying to find cables, connect to a network, or find the right application. A user should be able to send a picture from his phone to his PDA without typing in a password. As the number of mobile devices that a user owns increases, he expects to be able to have all of them communicate with each other without trying to find the right cable, the right connector, the network to use, the right application to use. The different types of connectors or cables vary from device to device and manufacturer to manufacturer [4].

Mobile devices, because of an effort to make them secure, require users to jump through many hoops before a connection can be made. No good method exists for a user to easily manage his devices. Passwords and PINs become a hassle as users have more and more devices to keep track of. Elaborate user schemes are created to maximize security for mobile devices. Efforts for wireless communications between mobile devices include the use of 802.11, Bluetooth, or IR. Because of power concerns, security issues, or bandwidth requirements, each of these technologies has their pros and cons [7].

The solution to this problem is to create groups of devices that have both implicit and explicit trust relationships. Creation of the groups require minimal user interface and remain secure through the use of public and private key encryption. This thesis uses Bluetooth as a specific example of how these groups can be applied to a real implementation. The requirements for the solution are:

- the system must be secure from eavesdropping and replay attacks
- the system must have an intuitive user interface
- the system must be easy/simple to use
- the system must be low power
- the system must be scalable to a large number of devices
- the system must be easily implementable
- the system must allow for adding and removing of users dynamically
- the system must not require substantial computational power or large amounts of memory

# Chapter 2

## Background

A user's world of computational devices and communications devices has rapidly changed from a world of static, wired devices, to a world of mobile, wireless devices. As the number of mobile devices in an environment increases, new infrastructures must be designed to support the changes. The same principles and ideas behind wired, static networks still exist in the wireless world, but the requirements are different.

The way in which devices are networked is dependent upon the number of devices, the type of applications the network is used for, and the capabilities of the devices themselves. To help understand the similarities and differences between wired and wireless networks, this chapter describes the different types of networks and some generalizations that can be made about them.

### 2.1 Types of networks

Wired and wireless networks are significantly different in the way they are intended to be used. Wired networks have a core infrastructure that connects the devices on the network to each other via set switch boxes or routers. Wired networks are used for backbone structures where nodes remain fixed in one location. Wireless networks can be either infrastructure, or ad-hoc, where there is no fixed method for connections or a persistent network. The connections are created dynamically when needed. Wireless networks are intended for applications where the nodes dynamically

move around within a network and also in and out of the network. Within each type of network, there are point to point connections, bus-based connections or piconets, and large distributed networks.

Wired networks support large numbers of devices, can transmit high bandwidth information, and are applicable for any number of uses. The devices on a wired network can range anywhere from a mouse or a keyboard to a server on the backbone of the Internet. Devices on a wired network have a small range of physical movement, their wires usually limit where they can be located. However, the wires and sedentary nature also allow wired devices to have unlimited power supplies and virtually unlimited storage capacity. Wired networks can be point to point connections, such as the connections between keyboard and PC. Wired networks can also be bus-based, like a network of USB devices; or large distributed systems, like the Internet or telephone system.

Wired networks can cover large areas or small areas. For example, the backbone of the Internet runs on thousands of miles of fiber connecting the entire world, while the connections between a PC and its peripheral devices is no more than a few feet. There is also a set structure for how the network is connected and how each device becomes a part of the network. Adding new devices to the network is a well defined procedure, whether it be paying an Internet Service Provider (ISP) for an account or talking to a system administrator to set up a new machine, it involves some sort of central system that maintains the network.

Security for wired networks generally uses a model requiring a centralized, trusted unit, as in a trusted third party system or a certificate authority [20]. Attackers can come from anywhere, whether it be from an authorized user of the network or an outsider breaking in to the network. The attacks can range from physical destruction of the connections or devices on the network, to denial of service attacks preventing any sort of communications. Security goals for networks include authorization, to ensure only the appropriate users can access the network; authentication, to make sure the users are who they claim they are, and confidentiality, to make sure only the appropriate users see sensitive data [16]. There are many types of security systems



to solve those problems.

Wireless networks, on the other hand, support functionality for mobile devices. The networks can be very application specific, as with sensor networks, or they can be very broad, as with wireless LANs. Because mobile devices are intended to be carried around or easily moved from place to place, mobile devices are limited by their power supply and physical size. Wireless networks, like their wired counterparts, can also be divided into categories of point to point, bus-based or piconet, and distributed networks. Point to point wireless networks are like cable replacement technologies for wireless mice or keyboards, or even the Bluetooth connection for sending pictures between two cell phones. Bus-based networks are like a piconet of Bluetooth devices, where a master is communicating on the same channel as all of the slaves [12]. Distributed wireless networks are like 802.11 technology, which is the wireless version of Ethernet.

Wireless networks can be ad-hoc or infrastructure networks. Ad-hoc networks are created on-the-fly, with no routing table for communications between devices. Ad-hoc networks are created when a series of devices create peer-to-peer connections and use the connections between nodes to pass information from one device to the next. Infrastructure networks have a set of access points that provide a persistent network. The network is not dependent upon the devices that are present, it always exists. The devices on an ad-hoc network can physically come and go. On the other hand, ad-hoc networks can fall apart if too many of the key hosts leave.

Wireless networks are susceptible to the same attacks as wired networks, but in addition, wireless networks can fall prey to eavesdropping and interference attacks. Wireless networks are much harder to limit access to because of their dynamic nature. Users on the network leave and return, impersonation attacks are a popular way to gain entry into wireless networks [21]. Especially for ad-hoc wireless networks, each node must be able to secure itself from attackers without protection from a system-wide, central security network [11].

The centralized security models do not work well for wireless networks because the devices are mobile. The central trusted party must be available no matter where

Table 2.1: Table showing the categories of wired and wireless networks.

	Wired Networks	Wireless Networks
point-to-point connections	serial cables, keyboards, mice	IR, RF
bus-based connections	USB	Bluetooth piconets
distributed networks	Ethernet	802.11 technology

the mobile device is. There are times when a mobile device might not be in contact range of the trusted entity. In addition, central authorities could be a single point of attack for a security breach into the network if there is no distributed system.

A summary of the different types of wired and wireless devices is given in Table 2.1.

## 2.2 Security Systems

### 2.2.1 Access Control Lists

Access control lists (ACLs) are one method for determining which users are authorized to use the network. The network administrator, or even each individual device, maintains a list of users allowed on the network (or allowed to access the device). Users must prove who they are by presenting some sort of ID and a password that corresponds to that ID. If the ID matches an entry on the ACL, the user is allowed privileges to the network. The IDs can be usernames, IP addresses, hardware device addresses or some other form of identifying token. Access control lists allow for easy maintenance since revocation of a user's access privileges are done by simply removing the user's ID from the access list. Access control lists work well if the lists being maintained are guaranteed to be secure and the administrator of the lists is a trusted entity.

Using access control lists also requires enough storage space to maintain a list of users and corresponding passwords. Access control lists can also be extended to groups for situations where individual entries become unmanageable. Individuals

belong to a group access list, and one identifying name is chosen for the entire group, which can be used for access.

## **2.2.2 Certificate Authority**

A model for authentication of a user is the use of certificates and certificate authorities. Certificate systems require a central trusted server that acts as the certificate authority. This certificate authority decides which users to trust. Trusted users are issued a certificate by the authority. The certificate acts as a token saying to other devices “if you trust this certificate authority, then you can trust me as well.” This system works well if devices do not want to maintain their own access lists. Anyone with a valid certificate will be trusted and it is up to the certificate authority to determine who is trusted. However, this system depends on a central server that must be secure. If the certificate authority is compromised, then all certificates become void and new certificates must be issued. Revocation of a certificate is hard because the user holds on to the token, so there is no way for anyone, even the certificate authority, to take away that certificate. Certificates usually have expiration dates after which the owner of the certificate must return to a certificate authority and renew the certificate or get a new one.

## **2.2.3 Key Encryption systems**

Various types of key exchange systems are also used for network security. Key systems allow devices both to encrypt a message and authenticate a user. Shared secret keys use one key that is shared between various devices that need to authenticate each other. The key can also be used to encrypt a message designated only for those individuals with the shared key. Shared keys are simple to keep track of since there is only one key, and computationally inexpensive, but the disadvantage is that once the more devices that know the shared secret, the more insecure it becomes. Thus shared secret keys are not readily scalable.

Public/Private key pairs are more complex because of the two keys that must

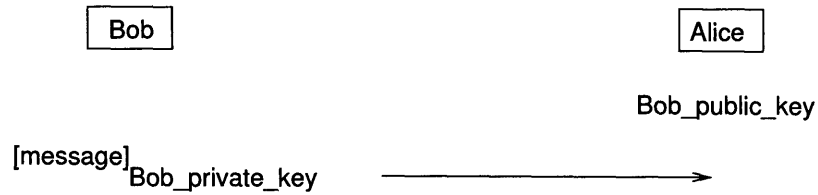


Figure 2-1: Bob sends Alice a message that has been signed with Bob’s private key. Alice has a copy of Bob’s public key and can verify that the message was from Bob by using the public key.

be maintained, and the associated computational complexity of the two keys, but have the advantage that a key can be used either for encryption of a message, or for authentication of a message. A user’s private key can be used to sign a message. The receiver of that message can verify who sent it by using a corresponding public key to verify the message, as shown in Figure 2-1.

Conversely, a user can encrypt a message with a receiving device’s public key, and the receiving device can decrypt that message with the private key. The convenience with public keys is that the system is secure no matter who gets a hold of a public key. There are also public key systems that employ the use of group keys [3, 6, 14]. This combines the idea of group access control lists with keys such that a group can have a shared public key to identify members of the group. The owner of the group issues some sort of token to each member of the group, much like a certificate, so that group members can prove their affiliation.

## 2.3 Wireless technology

There are many different wireless technologies, each designed to fill a certain need in a certain area. Depending on the size of the network and the distance of the communications, and also the amount and type of information passed on the network, the design will vary.

802.11 networks are a wireless version of Ethernet. Mobile devices connect to the rest of the network via access points. The access points contain the radios that send network data across the air. 802.11 networks operate in the unlicensed 2.4 GHz

spectrum, which is the same as microwaves and many cordless phones. 802.11 is fairly power hungry, especially for use in mobile devices that are often battery powered [7]. 802.11 chip size is also fairly large, so it's difficult to use in small devices that are limited in space. However, 802.11 is good for longer distance communications and for high bandwidth applications.

Bluetooth is designed for short range, wideband communications [15]. It is best used for peer to peer communications and exchanges of short messages or single files. Bluetooth is low power and small chip size. Bluetooth also operates in the 2.4 GHz spectrum. While the 2.4 GHz band has a fair amount of traffic, Bluetooth avoids interference problems by using a frequency hopping spread spectrum scheme so that packets are sent on different frequencies determined by a pseudo-random function [12]. Bluetooth has a variety of security models and encryption methods that make it complex to implement.

IR (infrared) is another wireless protocol that is useful for inexpensive, short range, low bandwidth communications [19]. Easily utilized in a system because no government licenses are required to operate in the IR range, IR is also limited because it requires direct line of sight communications. However, the line of sight limitations also make IR a simpler communications method to keep secure.



# Chapter 3

## Problem

The challenge is that a mobile user connects to many devices on a regular basis, and needs to be able to set up a trusted network with those devices, as shown in Figure 3-1. The security checks and authentication/authorization procedures needed to make a connection should be necessary only initially, and should happen automatically thereafter. Solutions to this problem should generalize to devices that are limited in user interface and computational capability. A user should not have to authenticate himself and enter a password each time. When a new device is introduced to the existing network, the user should be able to intuitively add the device as a trusted entity to the network with minimal interaction. The user should not have to count on a central authority being available to add the new device to the network.

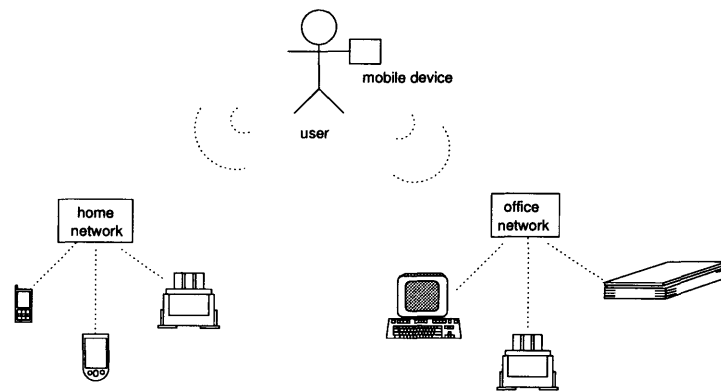


Figure 3-1: A mobile user and his networks of devices.

Current network models are not designed for ad-hoc networks. This chapter discusses the current problem and the challenges that need to be overcome and the requirements necessary to create a network of trusted mobile devices.

### 3.1 Too many mobile devices

Lots of mobile devices need to belong in a trusted network. A user should not have to explicitly make a connection between each of his devices every time he wants to use them. For example, Bob has a PDA that contains his calendar, his address book, and some documents he wants to read. To print wirelessly to his printer, Bob has to make a connection and enter his password to verify that he can use the printer. If Bob wants to update the calendar on his laptop, he has to enter his password to make a secure connection to his laptop. If Bob wants to synchronize the address book to his cell phone, he needs to create a connection and enter a password to authenticate himself to the cell phone. Bob should expect to just be able to use these devices when he needs to, not have to prove he is a valid user each time. There should be a way for the devices to remember that information.

Network connectivity with mobile devices is a difficult problem because of security, reliability, naming, and power consumption. Wireless networks are often insecure because of the tradeoffs between a fully secured network and ease with which a user can connect to it. With mobile devices that cannot remain on indefinitely, networks cannot count on having all devices present at all times to correctly route data from one node to the next. Each node must have all the information necessary to do what it needs to do.

Additionally, current network models are not human centric and do not provide for an intuitive way to connect devices securely. Entering passwords takes too much user effort, and maintaining access lists are infeasible on devices with limited memory. Also, many devices that would lend itself well to wireless technologies are not wireless because of the challenges associated with accessing networks securely using a device with minimal user interface.



## 3.2 Current network models

Different types of networks have different needs. A personal area network in someone's home does not have to worry about security as much as a large local area network at a company [15]. A large local area network at a company does not have the scalability problems with naming that a wide area network might have. The size of the networks and the density of the devices in a network plays a huge role in determining trade offs in a design.

The following section presents a few different network models grouped by the type and number of devices in the network. A discussion on the roles of naming and authentication in a networks is included. Networks can range from having no fixed naming scheme to a set address for each device. Authentication can range also range from authenticating each and every user or to no authentication at all. Comparisons are made between networks of static devices and mobile devices. Static devices are devices that have an unlimited power supply, persistent network connection, unlimited storage capacity, and high computation power. Mobile devices have limited power supplies, are not always connected to a network, have limited storage capacity, and limited computation power.

### 3.2.1 Network with static devices only

These devices are always connected to some sort of network via a wired connection or a wireless connection. The physical location of these devices does not change, and these devices are always on, as shown in Figure 3-2. If one machine is turned off, the other machines are still able to remain connected via some sort of centralized system. A system administrator individually adds every device or user (depending on the system) to the network using some IP address and maintains a secure network. Ethernet is an example of a network with static only devices.

Naming for a static system is simple since the devices do not move around from network to network. A fixed name or address for each device will always be sufficient to find the device. Authorization for a static system is also simple. Not only is there

the option to have a system administrator add and remove users from the network, but there are physical wires connecting the devices. Adding or removing users can be as simple as physically plugging in or unplugging the device.

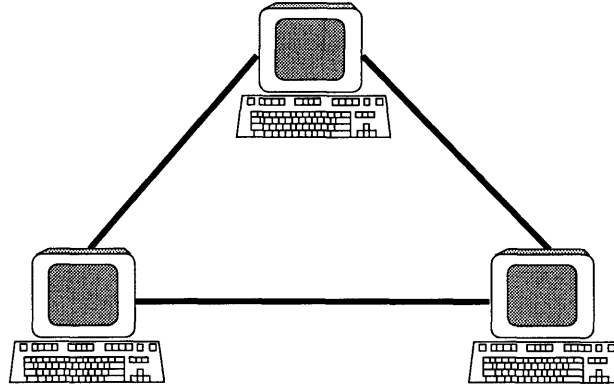


Figure 3-2: A network of static devices.

### 3.2.2 Network with some static devices and some mobile devices

This type of network has static devices that are connected to a fixed network, as described in the previous model, but in addition, has mobile devices that move in and out of the network. Mobile devices connect via a static device, using it as an access point, as shown in Figure 3-3. The entire network, like the previous model, is monitored by a system administrator that controls access to the network. These static devices can be PCs or access points that provide network connectivity.

The above model applies to almost all wireless networks currently in use, like 802.11 or cellular networks. Static access points administered by a central system allow mobile devices to connect whenever they are near an access point. Mobile devices can come and go as they please, knowing that there is always some sort of network present that they can connect to.

Authorization for wireless LANs have been implemented in two ways. One method is to check for authorization during the connection process to the system. This method places the burden on the access points and forces the user to present a secure

key or password to connect to the access point. The other method is to check for authorization after a user has been connected to the network but before the user is allowed to do anything on the network, i.e. send mail, open web browser, etc. Thus, the access points allow anyone to connect, but once connected, the system will check for authorization using passwords or some sort of pre-registration process.

Naming for a system with some static and some mobile devices can be done in a way that the mobile devices use the naming scheme of whatever static network they connect to. Thus, if a PDA is connecting to a PC, the PDA can follow the naming scheme of the PC's network. Because the mobile devices use the fixed network to communicate with other mobile devices, the fixed naming scheme allows them to locate each other.

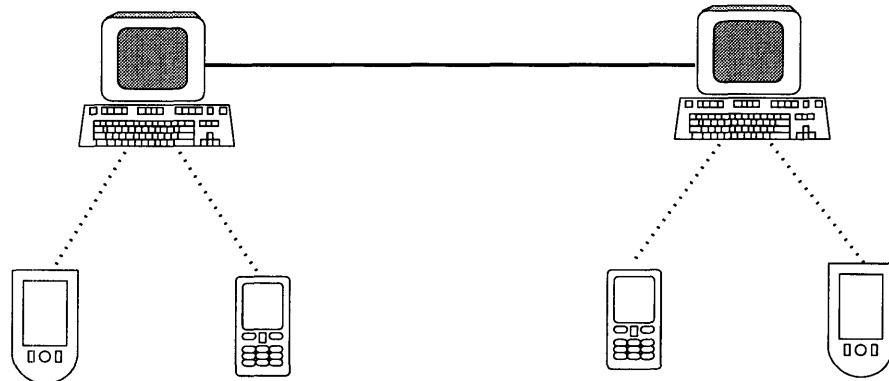


Figure 3-3: An environment with a mixed number of static and mobile devices.

### 3.2.3 Network with one static device and many mobile devices

This network is a subset of the previous network. Naming and authentication can be done as with the previous network. With the limit of only one static device, there is now a concern that the one static device may not be able to support all the wireless devices that need to connect to it. This model works for small networks like home wireless networks or cell towers in isolated areas. This model also works for wire or cable replacement systems. For example, everything in a home network is connected

to the PC—the printer, scanner, PDA, keyboard, mouse, etc. All the wires for those connections can be replaced by a wireless technology.

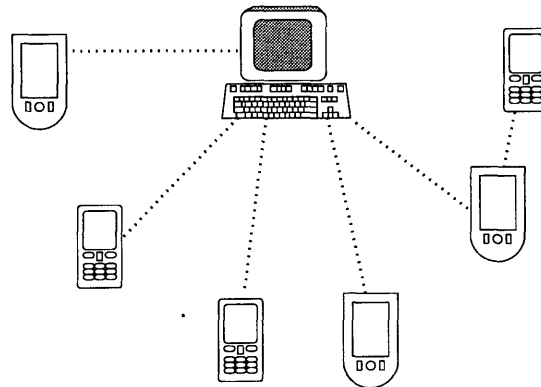


Figure 3-4: A network with one static device and multiple mobile devices.

### 3.2.4 Network with many mobile devices

In a network with many mobile devices, there is no longer the always on, persistent network that exists with the previous several examples. Just because one mobile device connects with another does not mean that they are suddenly part of a larger network. These two devices are isolated and can only communicate with each other. There is no central system or central static device that can help determine how to route packets from one device to another. Two mobile devices could be in the same room and have no idea about each other.

Picking a standardized naming solution for a network of completely mobile devices is challenging because there is no central node to determine the naming system for the entire network. The naming system could be established by the first two devices making a connection, but it might not work well for any subsequent devices. Most mobile networks have no set naming scheme. Devices are often renamed when they join a new network. In addition, prior to connecting, there is no way for a device to know the name of another. It cannot go through some router or switch box to ask for a connection to a certain device. Additionally, authorization also becomes difficult for completely mobile networks. The problems shown in Chapter 2 regarding security

for wireless networks is seen in this situation.

This is a situation where there is no good solution yet. There are many ideas for ad-hoc networks [18, 2], yet few have been deployed for actual use. Despite the fact that 802.11 networks can be implemented as ad-hoc networks, all working implementations have at least one static access point somewhere to act as a gateway to the rest of the network. If a group of laptops connected to each other with 802.11, it would purely be a LAN, with no connection to the rest of the Internet. If a new device comes along and managed to connect to one device, it has no way of knowing how to connect to the other devices. There is no common naming scheme such that a device can just say “find me Bob,” or “send this packet to this IP address.” Even if two cell phones are connected, there is no way for one of those cell phones to act as a gateway to other cell phones.

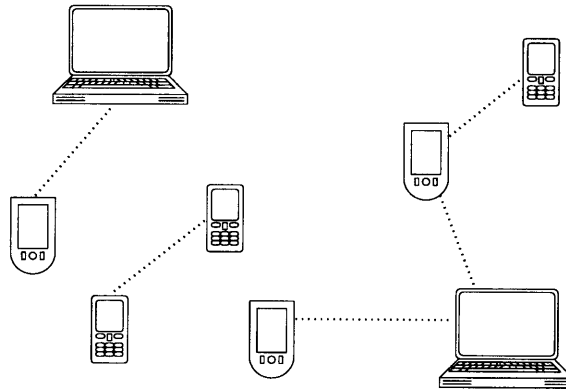


Figure 3-5: An environment with only mobile devices.

Access lists and certificates can be used on static devices and mobile and static networks for security. Mobile only networks require some sort of decentralized security system that does not depend on the presence of a specific node. Mobile to mobile connection is a hard problem because of the requirements necessary to build a functional secure system.

### 3.3 Current models

Many research projects have looked at ways to create secure ad-hoc networks for mobile-only systems. These systems include secure public wireless access for 802.11, cryptographic protocols at various different levels of the network layers, and group validation systems.

However, most projects look at the problem of securely routing ad-hoc wireless networks [18, 13] or depending on a minimum number of entities to create the base for a secure ad-hoc network [20]. While all of these solutions scale well for a large number of mobile devices, they do not scale down well for a small number of devices.

One proposal for providing security to mobile ad-hoc networks is the use of threshold secret sharing, as described in [10, 9]. This model proposes distribution of certificate authority responsibility to members of a network. Any  $K$  members can collectively perform the role of a certificate authority once  $K$  is greater than a predetermined threshold. An entity requesting to join the network can do so by collecting  $K$  partial secrets from the members of a network and creating a complete secret. While this model scales well for large networks and does not require the use of a central authority during operation, it assumes the presence of a central authority to create the secret in the first place. It also assumes that a critical number of secret holders will be present in a network. The goal in this system, and with using threshold schemes for managing cryptographic keys in general, is to find a balance between the number of secret holders it takes to make the system secure, without making that number so high such that it becomes inconvenient to access all the secret holders [17].

Another proposal is based on the PGP “web of trust” model [1, 8] where a user becomes trusted if a current member of the network vouches for that user or certifies his trustworthiness. Each new member of a network must be introduced into the network by a current member. The web of trust model is extended to ad-hoc networks in [11]. This is a localized trust model where a new entity requesting access to a network becomes trusted if some  $k$  number of trusted entities in the network agree within a certain set time period to trust the new entity. If there are not enough

trusted entities within range of the new entity, it must wait for more entities to come within range or move to a location with more devices. This model does not have the problem of requiring a certificate authority to start the chain of trust, but still depends on a device-rich environment.

### **3.4 Requirements for a new system**

While the above ideas create ways to establish networks of secure devices, they also depend on a critical number of devices to be present. The solution needs to be able to support a large number of devices, but also be able to scale down well for two- or three-device networks. The goal is to create a system where authentication of users and authorization of privileges need only be done once to add a member to a network and complexity of the device does not play a role in the type of system implemented.





# Chapter 4

## Solution

The solution to the wireless problem is to use a combination of access control lists and group keys to create clubs of devices that trust each other and do not require user interaction to make a secure connection. Minimal user interaction is required on initial connections to determine whether a device should be trusted or not.

### 4.1 Initial idea using a simple group model

The most natural and user intuitive way to implement the club idea is to have each device be a president of its own club, and the president controls access into its club by adding or removing members, as shown in Figure 4-1. At each initial interaction with an outside device, the user will choose whether to trust the device for future connections or not. Once the device becomes trusted, the president adds the device as a member of the club by adding a member ID to the club access control list.

Each device maintains its own club and carries that information with it. The device does not depend on a central server to tell it who is trusted and who is not. The device can manage that on its own.

While this is a very simple idea to think about, it does not scale well for large numbers of devices. The size of the access control lists grow proportionally with the number of trusted devices. With limited storage memory in small mobile devices, access control lists quickly become infeasible. In addition, this model is one dimen-

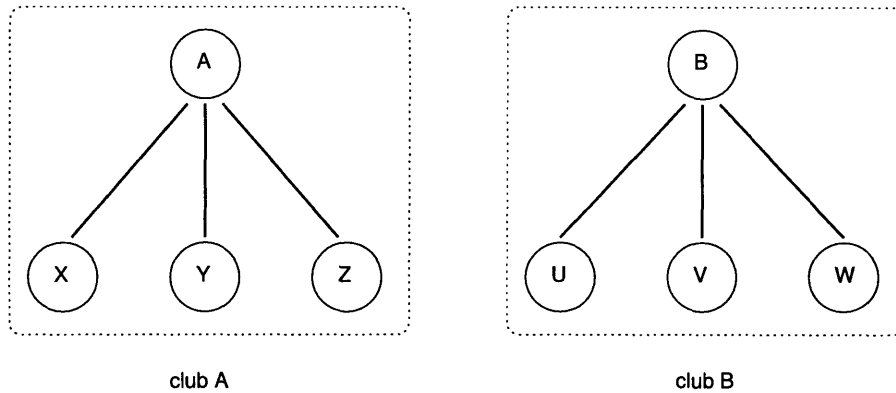


Figure 4-1: A simple model for groups showing two groups, club A and club B. Club A has members X, Y, and Z. Club B has members U, V, and W.

sional. The connections created between devices are still point to point. If devices X, Y and Z belong to device A's club, only device A knows to trust them. Devices X, Y, and Z have no knowledge of whether or not they should trust each other. The user would have to be involved in adding these devices to each other's groups.

## 4.2 Group model allowing cross-group membership

To solve the scalability problem, a model with hierarchies is established, as shown in Figure 4-2. This model builds upon the club idea by letting entire clubs join another club. In this way, each of the devices in a club can be identified by their club name instead of their individual device ID. Thus, if device X, Y, and Z belong to club A and device A joins device B's club, then device X, Y, and Z can also be trusted by device B. Instead of keeping track of every individual device that is a trusted entity, devices can now keep track of a group name, which acts as an identifier for multiple devices. This added layer of hierarchy helps reduce the size of the club membership list.

However, the chain of trusted devices can quickly grow too large and unmanageable. It's possible to soon include everyone in the same club just by joining the club

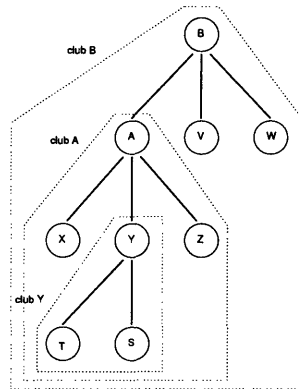


Figure 4-2: A group model with hierarchies. If club A joins club B, the X, Y, and Z automatically become members of club A.

that belongs to another club, that belongs to yet another club. This model does not scale well.

### 4.3 Groups with no hierarchy and implicit trusted links

To prevent the chain of trust from growing too large, clubs will not be allowed to join other clubs. However, the idea of using a club name will allow members of a group to identify each other, as shown in Figure 4-3. The chain of trust is now that, if the president of a group trusts a device, then all the members of the club trust each other. But, the members do not trust anyone outside of the club. If the club president trusts the president of another club, only the president will be added to the club, none of its members will be added, as shown in Figure 4-4.

To identify its members, each club issues some sort of token to each of its members so that they can prove that they are part of a club. It would be too easy to find out which clubs belong to which devices and simply claim membership to one of them if no proof is required. The group token must be some sort of secure token that cannot be faked by a device or copied from another device. Thus the token must be unique to both the group and the device that holds the token. Tokens could be faked if they

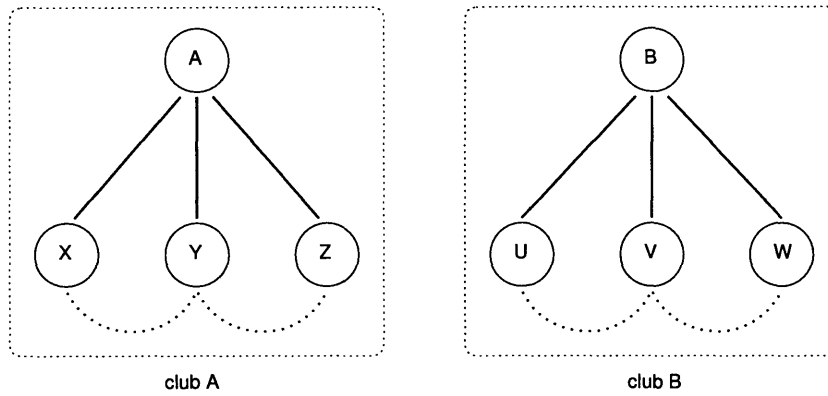


Figure 4-3: A group model with implicit trusted connections (dotted lines) between members of a group.

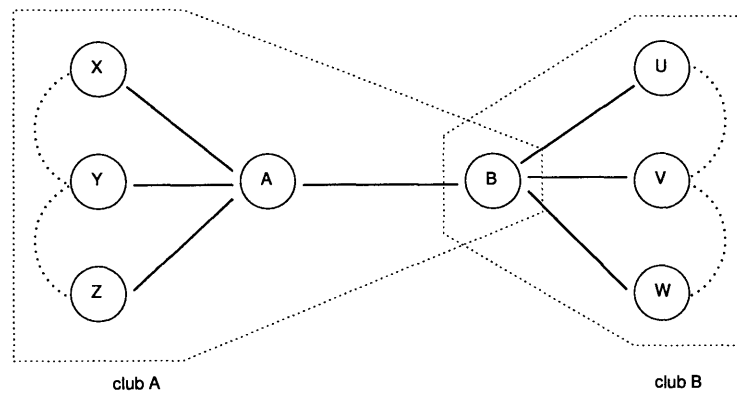


Figure 4-4: A group model with implicit trusted connections where clubs are not allowed join other clubs. B has joined club A, but U, V, and W, which belong in club B, are not automatically trusted.

are easily reproducible, like human readable strings of characters. Tokens could also be faked if they are simple enough that a dictionary attack could be used to guess a device's token.

Since the tokens become the proof of whether or not a device is a member of a club, the presidents no longer need to maintain a list of members. All a device needs is a public group key to verify that another device is indeed a member of the stated group. Devices also do not need to keep lists of trusted devices. The device need only keep the tokens of the groups it belongs to and the corresponding public group keys for those groups so it can verify other members.

### **4.3.1 Group Identification**

Using a plain text name for a group would make it far too easy for a device to spoof the name and claim to belong to a group. The club can use a public/private key pair to identify itself. The club president holds onto the private key, and the members hold onto the public key. The president can then verify any message sent to it by one of its members. However, since the group key is public, anyone can get it, even a device that is not in the club.

The president must have some way to tie the public group key to the device name. This suggests that the club president should sign the device's name with the club private key, creating a token, or certificate of sorts. Thus, to prove that the device is a member of the club, the device must present this certificate. The certificate is the device ID signed by the club private key. Now, any member of the club can verify that the device is a member by verifying the token with the group public key and seeing that the device address matches the device they are talking to.

### **4.3.2 Validating a Device Name**

However, it is still possible for a device to fake the device ID. An attacker could steal a device's certificate and pretend to be that device. Tokens are easily stolen since they are publicly available. To prevent this, each device also has a public/private

key pair. The device must create its own token first by signing its device ID with the device private key. Thus, anyone trying to verify the device need only have the device public key to verify the signature. When a device is added as a member of a group, this token is then given to the club president to sign with the group private key, which will be called a certificate. Now, even if a device fakes the device ID by spoofing a hardware MAC address or other device identification data, the user can still reject the device initially and never add it to a group. In the end, a real person still has to decide whether or not to add a completely untrusted, unknown device as a trusted entity. Thus even if the user trusts the spoofed ID, this model is no worse than what can happen in the current situation.

## 4.4 Group Model with Virtual President

However, with the club idea stated above, each device or user must remember who the president of the club is. Otherwise, a new device cannot be added to a group simply by connecting to members of the group. The dependency again lies in a central trusted authority. While a connection does not need to be persistent, the central authority does need to be present when a new device wants to be added to a network. The model is still not fully distributed. To solve this problem, members of a group should not have a physical owner, but rather a virtual president that all the members know about. Since all the members of a club are trusted, they can be trusted to add new members to the group. Membership in a group is still shown with a token that is the device token signed by the group private key. The level of hierarchy still remains flat in that there is only one level, as shown in Figure 4-5. However, new members no longer need to remember who the club president is or find the club president when they want to be added.

While two clubs still cannot merge their members such that all members of club A trust all members of club B if members of B and A connect, as shown in Figure 4-6, the argument can be made that allowing all the members to trust each other is unnecessary. To use a concrete example, just because Bob's PDA, which is a member

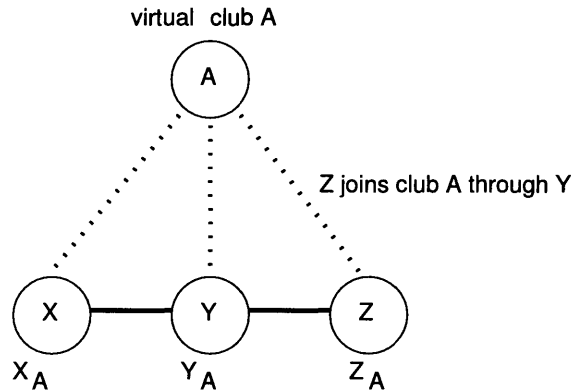


Figure 4-5: When X and Y connect to each other, they create a virtual club, called club A. When Z connects to Y, he is also added to virtual club A

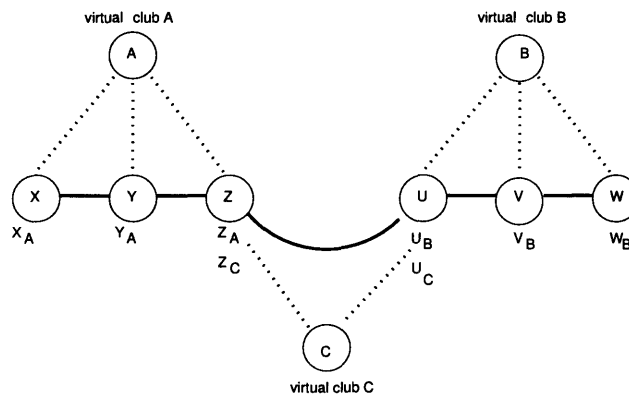


Figure 4-6: When Z and U connect, their respective clubs do not become automatically trusted networks. Rather, a new virtual club is formed that others can join.

of Bob's home group, is trusted by Bob's printer, which is a member of Bob's work group, Bob's printer does not need to trust Bob's home cell phone, as shown in Figure 4-7.

## 4.5 Group model with 3-tiered Hierarchy

How might this scale in such a way that not every member of the club will be carrying around the club private key? A hierarchy can be created such that it resembles a 3-tier structure, with a president, executives, and employees, as shown in Figure 4-8. The president (tier 1) of the club can add executives (tier 2) to the club. Executives can

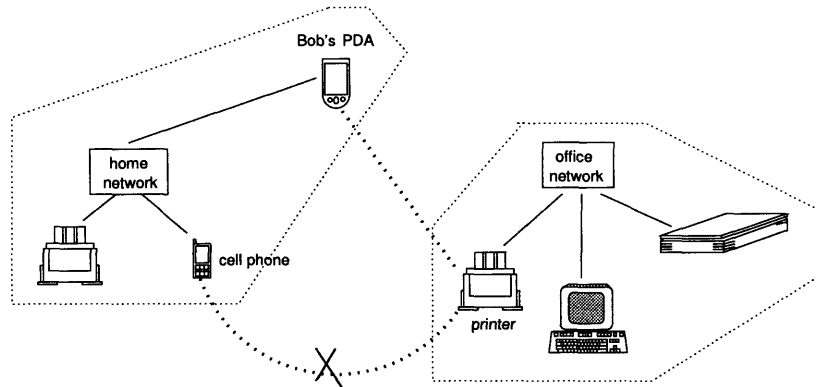


Figure 4-7: Bob’s PDA belongs to his home network, but also happens to be trusted by his work printer “group.” However, the trust chain does not need to extend such that the cell phone needs to trust the printer.

add employees to the club, but employees cannot add new employees. All members of the club, including employees, executives, and the president have implicit trust relationships between one another. Executives and employees of a “parent club” are allowed to become presidents of their own clubs (call them sub-clubs), but members of the sub-clubs are not automatically extended trust relationships to the parent clubs.

Executives and the president hold the club private key, and thus can give tokens to other group members. Regular employees are not allowed to add other members. This limits the number of devices that have access to the private club key while not limiting the number of devices that can join the group. Adding a new member does not require a device to find the actual owner of the group. In addition, this model is still very simple and requires no user knowledge of what happens in the background. The user still only needs to decide whether or not a device should be trusted and added to a group. The new idea is shown in Figure 4-8.

## 4.6 The Final Design

The final design is to use clubs to maintain trusted relationships between all members in the club. An example is of the process for devices to authenticate each other (4-9 and the process to add a new member (4-10). The final design has the following



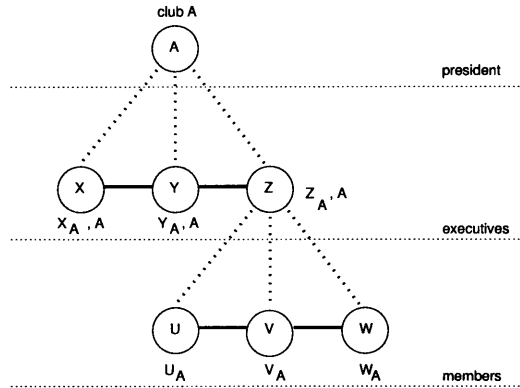


Figure 4-8: A model for 3-tier club hierarchy. Presidents can add executive members. Executive members hold the club private key, and can add regular members. Regular members cannot add others to the group.

specifications:

- No club will be able to join another club. Membership into clubs is only allowed on an individual by individual basis.
- Clubs are identified by a shared public key, the corresponding private key for the club is held by the club president and/or the executives, depending on the implementation.
- The club private key is used to add members to a club by signing a unique token belonging to each member of the club, creating a certificate.
- Tokens consist of a device ID that has been signed by the device private key.
- Certificates are device tokens that have been signed by a group private key.
- Verification will be done through the use of corresponding public keys.

This design is scalable because of the use of groups, and there is no longer a need for large lists. The design is also secure because of the keys used to both verify membership in a group and validity of the device. There is also not an interminable chain of trust that can grow too long to be useful.

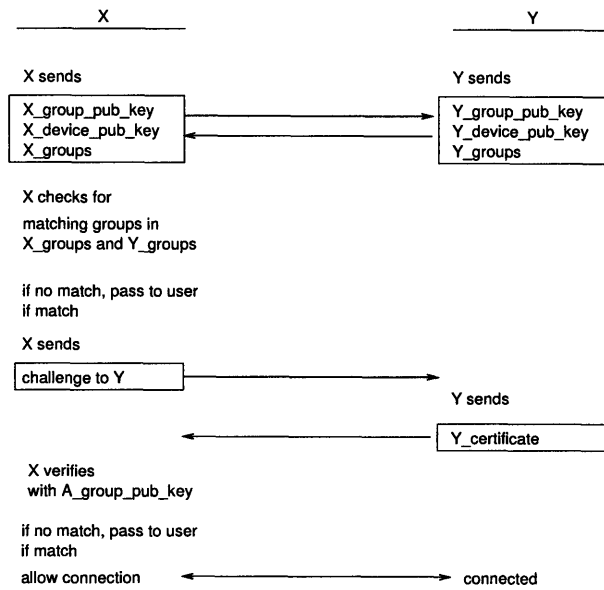


Figure 4-9: Device X and device Y belong to club A. This diagram shows the token exchange process that occurs when device Y wants to connect to device X.

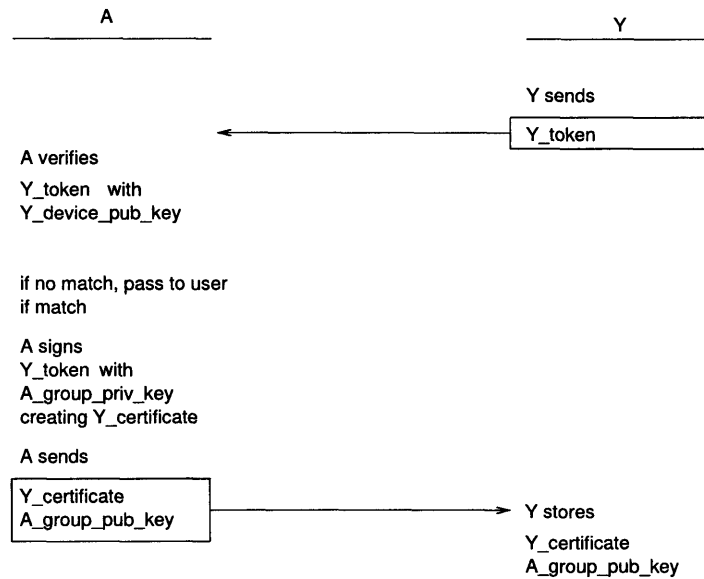


Figure 4-10: Device Y requests to be added to club A. At this point, the devices have already exchanged public keys and determined that they are not in the same group. The user has approved adding Y to club A.

# Chapter 5

## Implementation

An implementation of the idea from Chapter 4 can be done using Bluetooth networks. Bluetooth was chosen because it is ideal for small, mobile devices with limited battery life. In addition, Bluetooth device connections have the specific problem of requiring users to enter PINs and store device pairings in a complex user model. This chapter presents a detailed technical introduction to Bluetooth, implementation details specific to Bluetooth, and a summary of the implemented system.

### 5.1 Use of Bluetooth

Bluetooth was chosen because its capabilities in wireless mobile devices is optimized for an infrastructureless network.

In addition, Bluetooth device connections are a hassle and unintuitive. During a connection between two devices, a user must enter identical PINs on the two devices synchronously, otherwise the process will time out. Unless a user chooses to “pair” the two devices, the process must be repeated each time a connection is made. “Pairing” devices means saving the connection information for future use. While users can save a certain number of pairings for future use, that number is usually very low. A user should not have to enter a PIN every time he wants to update his cell phone address book to his PC address book.

Bluetooth is a radio technology that operates in the unlicensed 2.4 GHz spec-

trum. Created almost 10 years ago by researchers at Ericsson for short-range wireless communication, their original goal was to come up with a technology for wireless cell phone headsets. People soon realized that Bluetooth had the power to do much more and the Bluetooth Special Interest Group was established to formally lay out the specifications for the protocol as they stand today [12].

## 5.2 Technology background

Bluetooth is a radio technology that operates in the 2.402-2.48 GHz spectrum. The spectrum is divided into 79 channels of 1 MHz each. Signals are transmitted using a Frequency Hopping Spread Spectrum scheme such that packet transmissions are distributed pseudo-randomly across the 79 channels. Bluetooth is similar to 802.11 technology, but it is less expensive in terms of material cost and is also lower power to operate. In terms of size, Bluetooth chip sets are much smaller than those for 802.11, and thus are more appropriate for small mobile devices. Bluetooth has a maximum data transfer rate of 780 kbps.

Bluetooth can be divided into three main parts, as shown in Figure 5-1. There is the radio, which takes care of all communications across the air. It sends and receives packets from the baseband processor, which is the next part. The baseband does all the processing between the low level packets that are sent/received over the air and the data that higher level applications want to send/receive. The next level is the link manager, which manages the host controller interface between Bluetooth and the device that is sending data through Bluetooth. The link manager is important for understanding how connections are made.

### 5.2.1 Making connections

The following processes, as shown in Figure 5-2 are done by the link manager.

## Inquiry mode

Bluetooth devices connect by inquiring for other Bluetooth devices and requesting a connection. A device goes into *inquiry* state to ping for other devices in the area. Devices who are listening for inquiries are in *inquiry scan* state. While in inquiry (or any of its substates) state, devices utilize an inquiry-hopping sequence, which is twice the rate of the nominal frequency hopping rate. Devices can respond with their Bluetooth device address and a clock value from the device's Bluetooth clock. The clock value is used to generate a hopping sequence for paging devices. The inquiry-hopping scheme is twice the rate of the hopping scheme used by listening devices.

## Paging mode

When two devices decide to make a connection to each other, one device goes into page state to page the requested device. The requested device must be in page scan state to respond to pages. A device can be in page scan and inquiry scan state at the same time. Once appropriate packets are exchanged in page state, devices are then able to connect. Authentication can be done any time after the devices have connected.

The inquiry and page processes are slow because the devices are hopping on random frequencies in an inquiry (or page) hopping sequence. Inquiring devices hop twice as fast as the scanning devices. A scanning device also waits a random period

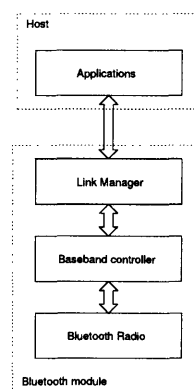


Figure 5-1: Bluetooth module components.

of time after it hears the second inquiry before responding to prevent everyone from responding at once. Thus, there is some latency between an inquiry and the time when another device will actually hear the inquiry.

## 5.2.2 Authentication

Authentication requires users of the two connecting devices to enter matching PINs into the device when prompted. It is up to the implementation of the higher level applications to determine if and when authentication is needed. Authentication can happen at any time while there is a connection between two devices. The matching PINs allow the generation of a temporary link key to secure the connection between the two devices. When it is determined that the devices are trusted entities, a permanent link key is generated from a 128 bit random number and the 48 bit Bluetooth device address. While most implementations require users to manually enter the PIN into the device, this not a requirement of the Bluetooth specification. The PINs can be preset within the device such that it always uses the same PIN, or a random PIN can be dynamically generated when needed and securely exchanged with the other device. Most implementations require users to enter the PIN as that is the most secure method of authentication since nothing has to be sent across the air that could be sniffed [5].

The exchange of PINs is not required every time two devices connect to each other. If two devices have connected in the past, and created a permanent link key (by going through the authentication process) that is still valid for both devices, then the link

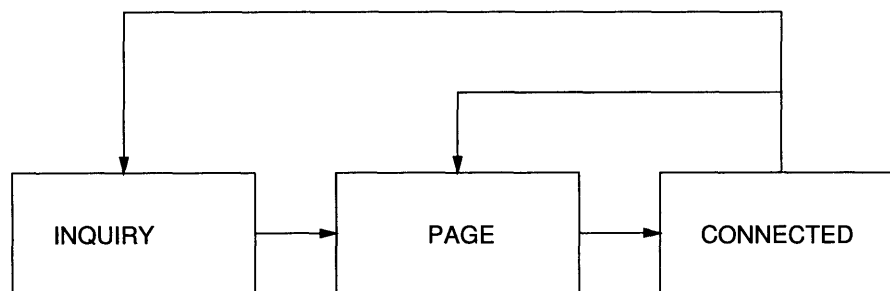


Figure 5-2: Bluetooth device modes while in the process of connecting.

key can be reused for the current session. The link key is used to determine if the devices are the intended target of the connection by using the link key to operate on a random number. The challenger sends a random number to the target device, the target device will then operate on the random number with the link key. The result will be returned to the challenger. If the result is the expected result, the connection is secured, if not, the devices will have to go through the authentication procedure again with a new PIN to generate a temporary link key, and then eventually another permanent link key.

## **5.3 Specific Implementation**

The ideal solution is to have a system where devices in the same group are able to identify themselves as such so that connection and authentication can be done with minimal user interface. The goals are to minimize user involvement in the connection process, make devices connect seamlessly and intuitively, maintain a secure network, and use a simple and straightforward design.

In Chapter 4, several ideas were presented for creating user groups. The two main ideas, one using a flat group structure with implicit trust relationships, and the other using a 3-tier hierarchy, are revisited here specifically in the realm of Bluetooth devices.

### **5.3.1 Bluetooth User Model**

There are three ways one can use Bluetooth devices: in a small personal area network, in a large public network, or in single point to point connections. Depending on the goals of each scenario, the implementation of the solution will have minor changes. The final result is that a slightly less optimal solution for general networks is actually a better solution for Bluetooth use.

In personal area networks, a user will use Bluetooth to link various devices in the home or at the office. Examples of this include replacing the cables between PC and printer, PC and scanner, PC and keyboard, etc. In addition, a user can send files

between his laptop and his PDA via Bluetooth, or even send contact information from his cell phone to his PC via Bluetooth. In a situation like this, a user might intuitively see the PC as a “president” of the devices in his home, a central node to which all the devices will connect at one point or another.

It would make sense to create a flat group structure such that each device authenticates itself to the PC, and the PC adds everyone to the PC group. All devices within the PC group have implicit trust relationships so that additional authentication does not have to be done when the PDA tries to print to the printer or the laptop receives a file from the scanner.

It makes less sense to create a hierarchical 3-tier structure because the user neither cares to know or needs to know about the hierarchy, and also because there are situations where the wrong devices might get added to a group. A scenario exists that a user’s laptop might be an “executive” member in both a home group and an office group. The laptop adds the user’s PDA to the home group as an “employee.” However, the laptop also adds the user’s PDA to the office group as an employee because the user does not choose which group the PDA gets to join—that process is transparent to the user. In this case, another member of the office group, say, a coworker’s PDA, will now automatically connect to the user’s PDA, which is not a secure situation.

In a large public network, a user might download information to his cell phone from a public kiosk using Bluetooth. For this scenario, the flat group structure does not work because the user does not necessarily want to trust all the devices that a public kiosk trusts. However, the hierarchy model does not work either, since it also provides implicit trust relationships between the user’s device and other devices trusted by the kiosk. A user needs functionality such that he can authenticate himself with one kiosk and be able to use all the other kiosks in a building without re-authenticating himself. A model that works well for this situation is the cross-group membership model, where entire groups of devices can join another device’s group. The entire group of kiosks in a building can join the device’s group, and the device can trust only the kiosks.



The last situation is point to point connections between devices. These are scenarios where a user might want to send a picture from his cell phone to his friend's cell phone via Bluetooth. This situation works well for the virtual president group structure. A group of friends who commonly send images or messages to one another would find it convenient to all be a part of the same group without having to make connections to each and every friend. Once a new friend joins a group by authenticating with one user, he is able to connect to all the other friends in the group without making additional connections. The distributed key system is good because a new friend does not have to seek out a specific person to join the group. Any one of the friends can bring a new friend into the group.

### 5.3.2 Design Goals

The situation this implementation focuses on is the first situation of a home group or an office group. Thus, the flat group structure with implicit trust links between members is used. To define a group, there should be a list containing descriptors that identify members of the group. When a device claims to be a member of a group:

- there should be a way to verify that the device is indeed a current member of the group.
- there should be a way to ensure that members of a group cannot be faked or spoofed.
- there should be a way to ensure that member lists are updated.
- there should be a way for members of the same group to detect that they are in the same group without going through a PIN exchange.

Each group will have three tiers: the president, the executive members, and the employees. The president and the executives will hold the private key for the group. The president can only add executives, and executives can only add employees. Employees cannot add new members to the group.

Every device belongs to at least one group: a group that contains itself, with the device itself as owner. The owner of the group controls which members are added and deleted. Devices can belong to as many groups as it wishes.

A simple example of the protocol can be shown with three devices: a laptop, a PDA and a cell phone. Each device can be the president of its own club. For this example, say the PDA requests to connect to the laptop. The laptop will check to see if the PDA belongs to the laptop's club, or if the two devices belong to the same higher level club. If so, the PDA is allowed to connect to the laptop. If no matching clubs are found, the PDA must join the laptop's club.

Next, say a PDA and a cell phone are both a part of the laptop's club. If the PDA and the cell phone want to connect to each other, they can do so because they both belong to the same club. Members of a club cannot add new members, only presidents may do so. Thus, if another cell phone comes along and becomes a member of the PDA's club, the cell phone is not automatically able to connect to the laptop because they are not part of the same immediate club.

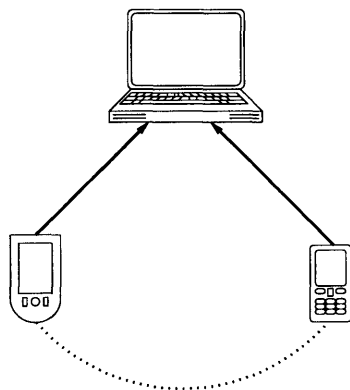


Figure 5-3: PDA and cell phone belong to laptop's club. PDA and cell phone can connect without additional user authentication because they belong to the same club.

The implementation can be divided into three main parts.

### 5.3.3 Device identification functions

Device identification functions are required to create keys and tokens by which devices and groups can be identified. This includes the generation of a public/private key pair for the device, public/private key pair for the club, and device ID token.

#### `my_keys()`

This function uses `RSA_generate_key()` to create both the device's public/private key pair and the club's public/private key pair. It writes the public and private keys into separate files so they can be accessed at later times and also so they can be sent to other devices as necessary. A device token is also created by signing the device's Bluetooth address with the device's private key, using `RSA_sign()`. This token is also saved into a file for access later.

### 5.3.4 Group and member validation

Group and member validation requires functions to sort through lists of groups, exchange public keys for both the device and the group, and verification functions to confirm whether or not the member belongs to the group and whether the device is who it claims to be. The list sorting functions are required to find out if two devices belong to the same group. Public keys need to be exchanged so that the verification functions can use public keys to verify that the tokens presented by members are valid.

#### `verify_member()`

This function reads acceptor and requester club lists and looks for a shared club. If a shared club is found, the acceptor is challenged to present a certificate showing membership in a club. The certificate is verified using `rsa_verify()` with the group public key. The device token is verified with the device public key. If the keys match, allow the connection, otherwise pass the function to the user to decide whether or not to connect.

### 5.3.5 New member services

New member services are a group of functions that allow a new member to be added to a group. This requires functions to sign a device token to produce a certificate showing membership to the group, and a PIN generation function to create device-generated PINs for the creation of a new temporary link.

`add_member()`

This function adds a new member to a group. The function must first verify the signature on the device token using `rsa_verify()` and the device's public key. If the signature verifies, the group president then signs the device token using `rsa_sign()` and the group private key. The signed token, or certificate, is then returned to the device where the device will store it in a file for later use.

### 5.3.6 Removing members

When a device is no longer trusted to belong in a club, there must be a way to remove it. To remove a member, a group generates a new group public/private key pair. While this is a tedious process to add all members back into the group, it is assumed that when a user decides to add a device to a group, he will not want to quickly remove it again.

This function simply erases the old club public keys from its file and generates a new club public/private key pair. All former members of the group will have to rejoin.

## 5.4 Bluetooth specific implementation trade-offs

The need for a link key to establish a secure Bluetooth connection provides two options for how to implement the system. If two devices are members of the same group but have never actually connected to each other before, a link key is necessary to create a secure link. To create a secure link, a permanent link key is needed. If a

link key exists from a previous connection, it can be reused. However, in this case, the devices are already trusted devices so the user should not have to decide whether to trust or not. The end result is that the user should never have to manually enter a PIN unless the two devices do not belong to the same group and the user must decide whether or not to trust a new member.

Thus, there are two ways to determine if a device should be trusted or not. The devices can either check to see if there is a permanent link key created from a previous connection, or check to see if the devices are in the same group. Thus, there are two options for the implementation:

1. check for group membership and device verification first, if those pass, then check for link key or generate a new one
2. check for the existence of a link key, if one exists, continue with connection, otherwise check for group membership.

Option 1 has the benefit that it is an intuitive model. Let the devices check for higher level trust before looking at the lower levels of link keys. However, the processes to check for group membership and then device verification are far more involved than just checking for a link key at the Bluetooth protocol level.

Option 2 is simpler to understand and implement. The permanent link key can be the same as the group public key. Thus, two steps, checking for groups and checking for link keys, can be done in the same step. If neither the group or the link key check out, then the device will need to request a new connection. However, since the link keys are also used to encrypt the connection, using a shared permanent link key makes the connections less secure.

The implementation shown will use Option 1 since the current goal is not to modify the specifications of Bluetooth.



# Chapter 6

## Conclusion

The goal of this thesis is to motivate and implement a solution for creating large networks of mobile devices intuitively. A user should be able to connect all the mobile devices he uses on a regular basis in a way that does not cause him to break the flow of his activities, like typing in a password. The solution to this problem is to create groups of trusted devices that, once added as a member of the group, required no further user interaction when connecting or using.

Current models for creating secure mobile networks are not appropriate for the needs of an average user looking to create a network of the devices that he uses on a day to day basis.

The solution to the problem generalizes well for large numbers of devices and a small number of devices as well. Authorizing and authenticating devices is not dependent upon the number of devices in a network. In addition, the number of connections a user must make is linear with regards to the number of devices in a network as opposed to being squared. The solution provides the user with a simple and intuitive way of organizing devices and creating connections between them such that he can use the devices on demand and not worry about remembering the right password or finding the right cable.

## 6.1 Extending functionality to all devices

The above solution would create a system in which secure connections between devices become very simple to create. The decisions that a user or a device needs to make becomes very clear: trust the device or not. This opens the door for devices with limited computational power or user interface/feedback to create secure connections. If “dumb” mobile devices can be given wireless capabilities, the possibility for natural interaction with wireless becomes realizable on a large scale.

One current problem is the prevalence of “dumb” devices that would benefit from Bluetooth capabilities but do not have them for one reason or another. Examples of “dumb” devices include scanners, MP3 players, printers, keyboards, mice, etc. Gradually, post-purchase hardware is becoming available to make off the shelf products Bluetooth enabled. For examples, there are dongles to make printers Bluetooth enabled, and keyboard and mice sets that come with Bluetooth built in. However, there is still the problem that these dumb devices, i.e. devices without user interface capabilities like buttons or display panels, are far less user intuitive to connect via Bluetooth because there is no feedback.

To bridge the gap between the devices currently on the market and a world where all devices can communicate wirelessly, a Bluetooth USB dongle can be used. The simplification of the connection process makes it easy to create a dongle that would allow off-the-shelf products that are not Bluetooth enabled to become Bluetooth enabled. This dongle will be able to give any USB device Bluetooth capabilities. Unlike Bluetooth dongles currently on the market, this device would accept USB (Universal Serial Bus) devices, i.e., it would allow any device with a USB-B or mini-B male connector to plug into it. Thus many devices like keyboards, mice, MP3 players, digital cameras, printers, and scanners can all be made Bluetooth capable. The devices do not even have to know that there is Bluetooth on the other end of the connection, for all they know, it is still USB, so the products do not require any sort of modification. The processing can be done all within the dongle itself. An example is shown in Figure 6-1.



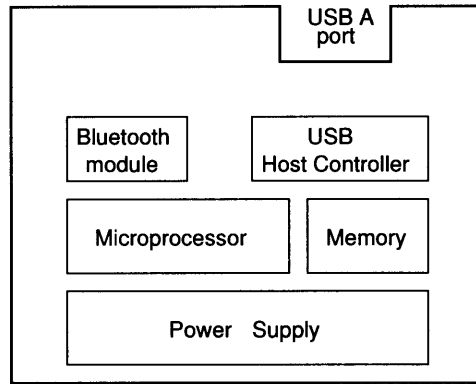


Figure 6-1: Block diagram of dongle design.

### 6.1.1 The Bluetooth USB Dongle

The user has to make two decisions: on one hand the user has to choose which device to connect to. On the other hand the user chooses whether or not to add a new device to a group. These are two functions that can easily be encapsulated onto two buttons that can be added to devices lacking any sort of complex user interface.

There is no need for other options. The user does not need to know whether a device is a president, an executive, or a member. The only difference that makes is one extra connection. If a user wants to connect to a device that is a member of club A, he can make that connection then and get the information he needs at that moment. If a user then needs to connect to an executive in club A, he still has to initialize that connection since he is not yet a trusted member of club A. However, if executive A decides to trust the user, he then becomes a member of club A. The user now has implicit trust relationships between everyone in club A.

The two buttons on this device need to provide the function for “connect to this device” and “allow device to join group.” Proximity can be used to determine which device to connect to. An example of the two-button device is shown in Figure 6-2.

### 6.1.2 Design

However, the design of this dongle is a fairly challenging task. There are a few main design goals:

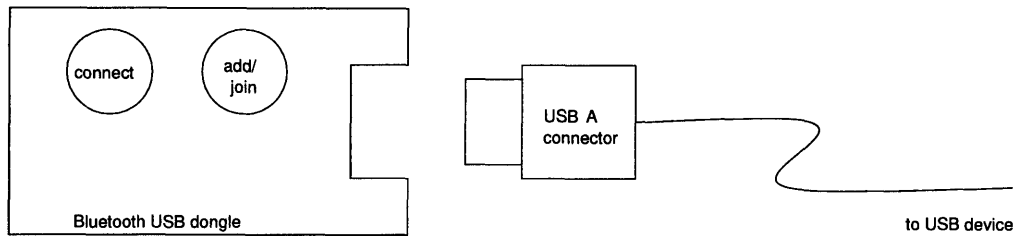


Figure 6-2: Prototype of dongle with two buttons. The “connect” button connects the device on the dongle to the nearest Bluetooth device. The “add/join” button adds a device to the group or joins a device’s existing group.

- the dongle must be wireless, since the point of making something Bluetooth enabled is to get rid of the wires.
- the dongle must be low power and be able to run off batteries.
- the dongle must be able to support the functionality of the design from Chapter 4.

Because the dongle has to accept USB slave devices, the dongle must be designed as a USB host. USB hosts, not to be confused with USB hubs, are required to be able to source a certain amount of current to downstream devices, have enough storage space to store a reasonable number of device drivers on the host, and must have USB-A female connectors. There are two connector types, USB-A, and USB-B. USB mini-B connectors are used on small devices that do not have enough space for a full sized USB-B connector. USB-A connectors are used explicitly for upstream connections, i.e. the USB ports in the back of a PC are USB-A female connectors, and the male connectors on the ends of keyboards and scanners are USB-A connectors. USB-B connectors are explicitly downstream connectors. The USB ports in digital cameras or MP3 players are USB-B ports, or USB mini-B ports.

Some of the challenges faced in designing such a product include the need for drivers for all the possible peripherals that might be plugged into the dongle. Some basic, generic drivers could be stored on board, but in order to make this dongle small, portable, and easy to use, memory on board is limited. Drivers could also be downloaded dynamically as they are needed, but there is still the problem that

once the device plugs into the dongle, the dongle needs to know how to communicate with the device. Another challenge is the need for a fast microprocessor to do the processing necessary both for the USB connection and the Bluetooth radio. Since USB packets are not in a standard serial format, there must be a way for the USB connection to process data from the Bluetooth device.

USB hubs, unlike USB hosts, are simply repeaters that take the signal from the USB host and repeat it to any devices that need to hear it downstream. Thus, the hub basically allows one USB port to expand to many. Hosts have far more functionality and require far more complexity. The computer is the most familiar example of a USB host.

In addition to the USB host capabilities, the dongle must also support Bluetooth capabilities. This requires a Bluetooth module that contains the Bluetooth radio and the baseband processor. The host controller interface on the Bluetooth module must be able to support some form of low level communication with the USB host controller.

## **6.2 Human-centric authorization models and usage scenarios**

The two-button dongle idea is a huge step towards designing human-centric systems for wireless, mobile devices. Two buttons were chosen because of its simplicity and appropriateness for the group keys model. It is obvious that a keypad is far more complex than necessary since it provides no improvement over the current model of entering PINs and passwords. The question becomes whether two buttons are really necessary. Two buttons are still more than a user should have to worry about. Can functionality be contained in one button? The single button can be used only for adding members or joining groups, and connections can be done automatically based on proximity and group membership.

The intersection of wireless devices and mobile personal area networks is an area

for which the solution space has thus far been all but human-centric. There should be a generalized solution for connecting wireless, mobile devices in such a way that the user only has to worry about whether or not he should trust the device as family or as a stranger. While both the two-button idea and the group keys model presented is intended for Bluetooth devices, it can easily be generalized for all scenarios where wireless devices need to connect and disconnect frequently.

Wireless, mobile devices are not limited to just those devices that interact with a computer or a laptop. Remote controls and home entertainment centers also very naturally fall into the category where intuitive ways to link remote controls with the appropriate device are useful. Does one button suffice for a situation like this? Does a user actually want to walk up to the DVD player with a remote control and press a button to “activate” the connection before being able to use the remote? One can argue that this is fine, since the user needs to walk to the DVD player to put in a DVD before he can watch it.

### **6.3 Closing thoughts**

In the end, a complete solution for allowing users to intuitively connect their mobile devices is far from complete, but this thesis has presented one idea which takes a step in the right direction. Mobile devices have become a permanent part of a person’s daily life, and the need to connect those devices to each other is crucial to how useful those devices can be. Current designs have not provided users a natural way to connect mobile devices.

This thesis creates a way for wireless mobile devices to easily connect to one another without the need for complicated passwords and user-visible security schemes. It presents a solution to the connectivity problem by using groups and hierarchy to allow each device to maintain a set of explicit and implicit trust relationships with other devices. The trust relationships allow devices to connect, authenticate, and authorize each other without the user having to remember passwords or PINs. An example of this solution is implemented to solve the current problem with complex

Bluetooth device connection models. The solution not only removes the unintuitive Bluetooth connection model of entering PINs and pairing devices, but provides an even more secure way to connect the devices.



# Bibliography

- [1] Alfarez Abdul-Rahman. The PGP Trust Model. *EDI-Forum: The Journal of Electronic Commerce*, April 1997.
- [2] William Adjie-Winoto et al. The Design and Implementation of an Intentional Naming System. In *17th ACM Symposium on Operating Systems Principles*, 1999.
- [3] Eric Ricardo Anton and Otto Carlos Muniz Bandeira Duarte. Performance Analysis of Group Key Establishment Protocols in Ad Hoc Networks.
- [4] Chatschik Bisdikian. An Overview of the Bluetooth Wireless Technology. *IEEE Communications Magazine*, pages 86–94, December 2001.
- [5] Bluetooth SIG. *Specification of the Bluetooth System, Version 1.2*, 2003.
- [6] Dwaine E. Clarke. SPKI/SDSI HTTP Server / Certificate Chain Discovery in SPKI/SDSI. Master’s thesis, Massachusetts Institute of Technology, September 2001.
- [7] Deborah Estrin, David Culler, Kris Pister, and Gaurav Sukhatme. Connecting the Physical World with Pervasive Networks. *IEEE Pervasive Computing*, 2002.
- [8] Joan Feigenbaum, Matt Blaze, and Jack Lacy. Decentralized Trust Management. In *Proceedings of the IEEE Conference on Security and Privacy*, May 1996.
- [9] Jiejun Kong et al. Adaptive Security for Multi-layer Ad-hoc Networks. In *IEEE International Conference on Network Protocols*, November 2001.

- [10] Jiejun Kong et al. Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks. In *IEEE International Conference on Network Protocols*, 2001.
- [11] Haiyun Luo et al. Self-Securing Ad Hoc Wireless Networks. In *IEEE International Symposium on Computers and Communications*, 2002.
- [12] Brent A. Miller and Chatschik Bisdikian, Ph.D. *Bluetooth Revealed*. Prentice Hall PTR, 2nd edition, 2002.
- [13] Panagiotis Papadimitratos and Zygmont J. Haas. Secure Routing for Mobile Ad hoc Networks. In *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, January 2002.
- [14] Ronald L. Rivest and Butler Lampson. SDSI - A Simple Distributed Security Infrastructure, September 1996.
- [15] K. V. S. S. S. Sairam et al. Bluetooth in Wireless Communication. *IEEE Communications Magazine*, pages 90–96, June 2002.
- [16] Jerome H. Saltzer and M. Frans Kaashoek. *Topics in the Engineering of Computer Systems*. Massachusetts Institute of Technology, 2002.
- [17] Adi Shamir. How to Share a Secret. In *Communications of the ACM*, 1979.
- [18] Godfrey Tan. Self-organizing Bluetooth Scatternets. Master's thesis, Massachusetts Institute of Technology, January 2002.
- [19] Roy Want, Bill N. Schilit, et al. The ParcTab Ubiquitous Computing Experiment. Technical report, XeroxPARC, 1995.
- [20] Hao Yang, Xiaoqiao Meng, and Songwu Lu. Self-Organized Network-Layer Security in Mobile Ad-hoc Networks. In *IEEE International Symposium on Computers and Communications*, July 2002.
- [21] Lidong Zhou and Zygmont J. Haas. Securing Ad Hoc Networks. In *IEEE Networks*, 1999.