

SHAREGEAR: AN INFRASTRUCTURE FOR POCKETPC RESOURCE SHARING

Jatin Sethi, Dept. of Computer Science, SUNY Stony Brook

ABSTRACT

The keyword driving the wireless economy today is 'Mobility'. The wireless world has found a new motivation with the introduction of the handheld in the early nineties and it's growing popularity. Keeping these handheld connected has not been an easy task. Handheld tend to be a lot more mobile than laptops. Initially meant to be a PC companion, the handheld computers are now emerging with their own independent identity. Not only the consumer, the enterprise user also has been depending on the handheld, not only as an organizer but also as a conduit to stay connected with the rest of the world. Today, we see people checking stock quotes, sending e-mails and instant messages, exchanging appointments, and beaming files - all through their handheld. The technology behind these handheld has only recently started maturing and still lacks in efficient sharing mechanisms. It can be argued that sharing is equally, if not more important, on the handhelds than it is on it's desktop counterpart. IR and Bluetooth beaming is only an option for one to one file sharing and is not suitable for people on the move that move in and out of networks. Setting the handheld to access a particular network can be cumbersome if wireless networks are secured by access keys. Situations like these present the demand for a secure sharing mechanism that can be setup automatically and instantly, without much user intervention. ShareGear answers to this demand by presenting an easy to deploy sharing infrastructure that is secure and requires minimal user intervention. This infrastructure, created over a 802.11b ad hoc network, forms the base for a secure file sharing service. Security is provided by encryption and a protocol to authenticate the users of the infrastructure.

1 INTRODUCTION

ShareGear presents an infrastructure for secure file sharing. ShareGear presents an easy to use interface that requires no manual setup for supported interfaces. ShareGear is based around the central idea of a trusted group with something to share. The devices in the group are authenticated among themselves through a shared secret. The secret is manually agreed upon at the time of group creation and all group members holding the group secret are authenticated among themselves. Authentication guarantees that the files shared by any group member would be made available to all

group members. Once trust is established among the sharing members, files can easily be added to the share by any of the group members. ShareGear guarantees replication of the shared file on each of the "Trusted" devices. In doing so it presents the abstraction of a shared network drive. This "Share" exists on every device as a specially designated folder where ShareGear replicates all files that are shared by it's members. Since replication is performed at the time of file sharing, the file does not need to be downloaded and is readily available upon access. ShareGear's architecture allows for other shareable and features to be added to it's infrastructure to further enhance the services and the user experience.

2 WINDOWS CE OVERVIEW

WindowsCE is an embedded Microsoft Operating System designed from the ground up, targeting devices that are unable to run the large footprint of it's cousins: the desktop Windows OS. WinCE's modularity, small footprint and support for multiple processors makes it portable to a myriad of embedded platforms. Unlike it's desktop cousins, Win CE is not backward compatible with DOS and other desktop operating systems. A concise description of WinCE can be found in the words of the author of the best known WinCE book. Doug Boling [R5] describes Windows CE as, "*a lightweight, multithreaded operating system with an optional graphical user interface. Its strength lies in its small size, its Win32 subset API, and its multiplatform support.*" In it's early days, designed primarily as a handheld organizer operating system, WinCE has evolved immensely since version 1.0 was announced in Comdex '96. The latest major version 4.0 is flaunted by Microsoft as an advanced real-time embedded Operating System with secure and scalable networking, real time processing and support for rich multimedia and web browsing capabilities. Even though the extent of it's use, as imagined by Microsoft, is yet to be realized, WinCE is inarguably the most widely recognized embedded operating system. Microsoft provides a good collection of tools and SDKs to facilitate development on WinCE based platforms. WinCE inherits a subset of Win32 API, providing the developers of Win32 platforms a familiar interface. Development for WindowsCE devices is performed on the desktop, and the application downloaded to the device or an emulator for testing. Microsoft provides a platform development tool, Platform Builder, for OEMs to facilitate porting

WindowsCE to new hardware platforms. Support for a wide variety of development environments including Java, C/C++/C#, Visual Basic is available through a number of Microsoft and third-party tools available in the market.

Microsoft defines specific market oriented platforms based on WinCE. HandheldPC, PocketPC, Smartphone and PocketPC Phone edition are such platform specifications. Unlike the base WinCE OS that can be flexibly ported to any platform, the predefined platforms must adhere to Microsoft specifications. PocketPC is inarguably the most visible WinCE based platform for the PDA. PocketPC, a WinCE incarnation, adds a user friendly shell and several other features specifically designed for a more personal handheld user.

PocketPC Limitations:

Two potential issues that haunt developers switching from desktop development to the handheld environment are PocketPC screen size and battery life. To be more precise, these limitations are common to the entire family of handheld computers. The screen size provided by most PocketPC devices is a mere 360x240. Even though the size of the screen available to the applications is much smaller, the size of the controls cannot be reduced proportionally to the screen size. Further, the mouse buttons are not available and all input must be funneled through the stylus. This can be difficult, especially if extensive user input is required. Thus the application must be designed with the small screen in perspective, and requiring minimal input from the user. Battery life is another major limitation on a PocketPC device. Handheld computers are powered by rechargeable batteries. Unless the desktop computers, where power is not a concern, these batteries can power the device only for relatively shorter periods. The Operating System usually puts the device in a low power modes to improve battery life. With improvements in battery technology, the battery lives may become lesser of an issue. However, the screen size will continue to demand close attention from the developer community.

3 AD HOC MODE

802.11 is a wireless LAN specification that defines 1 and 2 Mbps transmissions in the 2.4 GHz frequency band. The standard allows for two types of modulation schemes: Frequency Hopping Spread Spectrum (FHSS) and Direct Sequence Spread Spectrum (DSSS). IEEE accepted the 802.11 specification in 1997. The 802.11b specification extended the 802.11 specification in 1999 and allowed for 11Mbs transmissions over DSSS. Several other extensions have since then been accepted by IEEE. The 802.11 specification also

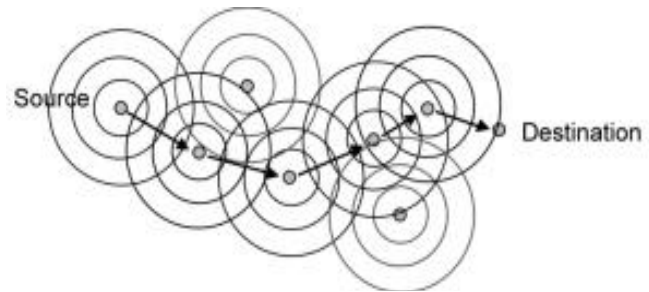


Figure 1. Routing in an ad hoc network is performed by passing messages among stations until the intended receiver receives it.

defined two functional modes for WLAN implementations: Infrastructure and Ad hoc. The WLAN, as we see it today, consists primarily of Infrastructure mode 802.11 implementations. Infrastructure mode requires WLAN cards communicating with Access Points (APs) that are gateways into the wired backbone.

In contrast to infrastructure mode, ad hoc WLANs have not attained comparable levels of acceptance. The ad hoc mode is by definition: “A network composed solely of stations within mutual communication range of each other via the wireless medium (WM). An ad hoc network is typically created in a spontaneous manner. The principal distinguishing characteristic of an ad hoc network is its limited temporal and spatial extent. These limitations allow the act of creating and dissolving the ad hoc network to be sufficiently straightforward and convenient so as to be achievable by non-technical users of the network facilities” [R6]. To what extent an ad hoc configuration is “straightforward and convenient” is debatable. Configuring ad hoc networks can often be challenging and getting different OEM implementations to work together can be difficult. Irrespective of this limitation, ad hoc networks, by making the wireless networks infrastructure independent, are an asset to wireless networking.

4 SOFTWARE ARCHITECTURE

The primary purpose of ShareGear is to facilitate the creation of a secure infrastructure upon which sharing services can be based. The present implementation supports a File Sharing service that allows for files to be shared securely between the authenticated users of this infrastructure. The creation of a group is central to ShareGear. The group distinguishes between the authenticated and other users that may be on the same underlying data link. In this case, the data link is the ad hoc network. The group is not only critical for the working of the sharing infrastructure, but also for its security. The group is created and maintained dynamically and only requires a password as input from the users. ShareGear initialization involves

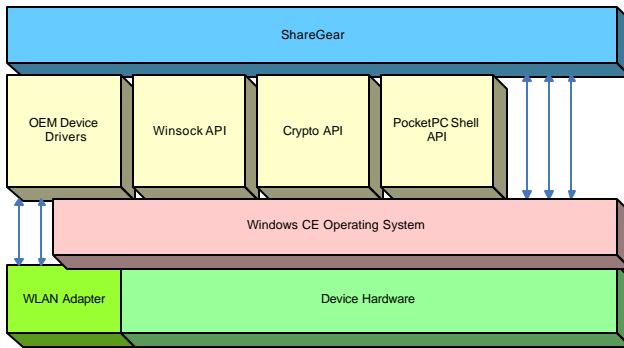


Figure 2. Architectural block diagram outlining ShareGear’s interaction with various components of the underlying platform.

presenting the user with a dialog to enter the group password. The group password allows the user’s instance of ShareGear to authenticate with other members of the group. The group management is handled by a component known as the Control Server. The Control Server is a UDP based server that listens for control messages. Control messages are used to advertise device presence and for authentication. Upon authentication a member is added to a group data structure. This group data structure is the central repository where all membership and status information is maintained. IP Address of the member, handles to the client and server sockets, handle to the thread responsible for sending files and lists to hold status and state information of a file transfer are all part of this data structure. The purpose of maintaining this information becomes clear as we delve into the details of various ShareGear components.

The File Sharing service is responsible for replicating the shared files securely through the entire group. This involves copying each file individually to multiple devices, over the sockets created during authentication. Server and client sockets, for respectively receiving and sending are utilized for transferring multiple files simultaneously between members. This multiplicity is achieved by threading and state maintenance. Thus for every pair of devices in the group, there exist four sockets that provide two channels for bi-directional communication between the pair. Since these sockets are key for the file transfer service to function, they must be actively maintained through the life of the group membership. PocketPC devices, with efficient power saving features, present a problem towards meeting this goal. PocketPC devices suspend power to the accessories when entering sleep mode, thus resulting in a loss of sockets. To workaroud this problem, ShareGear keeps the device from suspending. This is achieved by determining the current suspend timer settings on the device and

then resetting this timer before it signals the device to suspend. The drawback is a strain on the battery life. PocketPC 2003 devices will allow devices to suspend while maintaining power to the WLAN card, thus avoiding the overall problem.

One of the goals of ShareGear is to minimize user interaction in the setup and management of the infrastructure. ShareGear provides built in support for discovery and configuration of supported WLAN cards. Further, the application interface is provided through tabbed dialog boxes. This is a very well accepted interface design approach in the handheld community, given the minimal real estate available on PocketPC displays. The five tabs allow for File selection, Status viewing, Peers viewing, Log details and configurable Options. This user interface has been created entirely in C with the Resource Editor provided by the eMbedded Visual environment. MFC (Microsoft Foundation Classes), although available for PocketPC devices, was not considered as an option because of some known problems. A wide variety of development environments now support development on WinCE based devices. These environments vary in sophistication from scripting based Rapid Application Development Tools to C based tools for OS development. The eMbedded Visual Tools from Microsoft was our de facto choice since WinCE and PocketPC SDKs are based on it, and also because this environment allows maximum flexibility and control over the underlying Operating System. ShareGear was thus entirely implemented in the C programming language.

This purpose of the above discussion was to provide a high level overview of ShareGear’s architecture. Following sections discuss individual components of this architecture in details.

4.1 USER INTERFACE

PocketPC devices, with their small displays, can often be challenging to design graphic interfaces for. The key to designing a good interface on any handheld device is understanding it’s user and minimizing the user’s interaction with the interface. The interface should present it’s components synchronized with the expected relative frequency of their use.¹ ShareGear closely adheres to this principal.

The tabbed interface consists of five tabs in the following order– Transfer, Status, Peers, Log and Options. Transfer Tab is active by default and provides file browsing and selection through a Microsoft File Explorer style tree-based file browser. A transfer can be initiated by selecting the desired file and by tapping on the Transfer button located between the Browser and the Share view.

¹ PalmOS designers deeply believed in this philosophy. Intuitive and friendly GUI is one of the prime reasons for the success of PalmOS. The “Zen of Palm” document delves into the details of Palm’s application design philosophy.

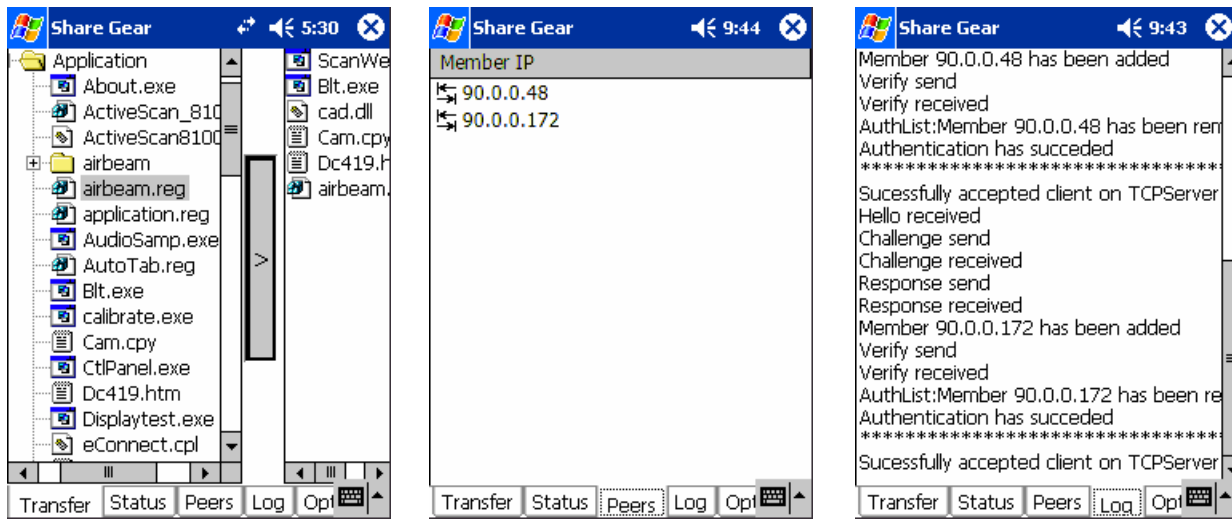


Figure 3. User Interface of ShareGear. Simplistic, tabular and easy to use. The Transfer tab is used to browse and select files to be send to the Share. A file is selected and the button in the middle tapped to initiate the transfer. The Peers tab displays the authenticated members of the group and the Log tab gives a comprehensive log of the operations being undertaken. This log is also written to a text file.

Once the transfer button is tapped, the file will appear in the Share view indicating that the file is now shared. The Status Tab is intended to provide a status for all incoming and outgoing file transfers in progress at any instant.² The Peers Tab provides a list of authenticated devices that constitute this shared environment. The Log Tab displays detailed status information on system setup, authentication and file transfer. ShareGear maintains a verbose log of it's operation . This log, as it shows on the Log Dialog, is also saved to a log file. The log is intended to troubleshoot problems that may arise during setup and operation. The last tab, the Options Tab², is intended to provide configurable options to customize ShareGear behavior. ShareGear is capable of running in the background. The close icon does not exit the application but minimizes it to the taskbar. The Task Bar menu icon when tapped presents menu options to view the full screen application, launch the About dialog and to exit the application. The application runs minimized when the task bar icon is visible.

4.2 WLAN CONFIGURATION

ShareGear aims at minimizing user interaction during setup and configuration of an ad hoc network. Towards this end, ShareGear automates this process by searching for supported WLAN adapters on the PocketPC device it is running on. If one or more supported adapters are found³, ShareGear selects and configures the first supported adapter for it's purpose. Configuration involves setting

the adapter in ad hoc mode, followed by the assumption of an IP address. WLAN Configuration can be seen as a three step process involving Adapter Discovery, Adapter Configuration and IP Address Assignment. A brief discussion on each of these processes follows:

4.2.1 Adapter Discovery

Adapter Discovery involves selection of an adapter from a list of available adapters on the device. This list of available adapters on a device is returned by calling the GetAdapterInfo API. Each member of this list contains information including adapter name, type, MAC, current IP address and the list of secondary IP addresses associated with the adapter. Each installed adapter has a predefined type. WinCE defines seven adapter types: Ethernet, Token Ring, FDDI, PPP, Loopback, SLIP and Other. The "Other" type is used for cards that do not define their type as any of the remaining six in their NDIS implementation. Since the WLAN drivers emulate Ethernet to higher layer NDIS drivers, the adapter type does not give away information on a particular Ethernet card being wireless or not. ShareGear works around this issue by basing the WLAN card detection on adapter names. Adapter names are usually unique to every WLAN card. For example, the Symbol 802.11b cards use Adapter name "NETWLAN1" and Cisco cards use "Cisco1". This method is sufficient since the primary purpose of Adapter Discovery is to configure the card for ad hoc mode. A card with a different name will specifically have to be supported in ShareGear to allow automatic configuration.

² This tab is currently not implemented.

³ Presently Symbol LA4121 is the only supported adapter.

4.2.2 Adapter Configuration

WinCE 3.0 does not define a common API (Application Programmer Interface) that can be used to configure network cards programmatically. Instead Adapter Configuration is performed through OEM specific API. It may not be possible for some commercially available cards to be configured programmatically. Consumer 802.11 implementations may not expose an API to minimize product cost. ShareGear initially supports only the Symbol LA4121 card that exposes a functional interface for configuring the card programmatically. Symbol drivers provide S24DriverExtension API that accepts certain function codes to be sent directly to the driver. ShareGear uses predefined function codes like S24_SET_FUNCTIONAL_MODE and S24_SET_ADAPTER_ESS_ID to configure the adapter upon initialization. The configuration resulting from these function codes sets the functional mode of the adapter to ad hoc followed by selection of an ESSID (“Share”) and setting the first available channel.

4.2.3 IP Address Assignment

Once the adapter configuration is complete, an IP Address is assigned to the adapter⁴. The Adapter’s IP Address is derived from it’s own MAC address. The MAC address, that was earlier returned by the GetAdapterInfo API, is an 8 byte array. The modulus of the sum of the MAC bytes when divided by 255, if resulting in a number between 1 and 254 is to be used as the IP Address for the adapter. If the modulus results in a 0, the calculation is repeated with 128 as a divisor. This ensures that the resulting IP address is between 1 and 254. The uniqueness of the generated IP Address on the ad hoc network is ensured by an OS process known as Gratuitous ARP. Gratuitous ARP or courtesy ARP, refers to determination of the existence of duplicate IP Addresses on a network, by a host that intends to use that IP Address [R9]. In other words, Gratuitous ARP by a host is an ARP (Address Resolution Protocol) request looking for it’s own IP Address. This request on Windows based system is send up to 3 times: once when the stack initializes and two others at intervals of 0.5 and 1 second, respectively. If a reply is received for any of these requests, the IP Address, for which the request was sent out, is reset. ShareGear depends on Gratuitous ARP for determination of a unique IP Address. It does so by checking for the added IP Address, one second after it was assigned to the adapter. If the IP Address was reset ShareGear is forced to go through the above process again to assign another IP address.

⁴ ShareGear uses non-routable IP Addresses between 90.0.0.1 and 90.0.0.254.

4.3 SECURITY FEATURES

Radio waves can travel through air and vacuum. In this respect, wireless networks are a double-edged sword. The very feature, that provides unlimited mobility within wireless networks, is a medium that is open and free for all. The lack of inherent physical security exposes wireless networks to security vulnerabilities that were not known to wired networks. Wireless networks try to compensate for the lack of physical security in higher layers of the network. Several security schemes at various levels have been designed to overcome this vulnerability: the most widely used wireless security schemes being WEP (Wired Equivalent Privacy). Even though WEP secures all traffic on the data-link level, ShareGear utilizes it’s own authentication and encryption schemes. WEP does not stand up to the demands of an on-demand sharing system for several reasons. The most prominent of them being the relatively unavailability of ad hoc implementations of WEP. 802.11 standard WEP description states: “Authentication shall be used between stations and the AP in an infrastructure BSS. Authentication may be used between two STAs in an IBSS.” [R6]. In essence, implementation of WEP over ad hoc is optional and most vendors opt not to implement it. Key distribution proves to be a bane to WEP. WEP does not define a key distribution protocol leaving it to be manual and painful. Moreover, it has been demonstrated that WEP can be compromised based on the vulnerability of the stream cipher (RC4) to known-text attacks. Another security option, Kerberos allows for security at an application level. Kerberos implementation requires a centralized server to act as an Authentication and Ticket Granting Server and as such is not suitable for a mobile implementation required by ShareGear. In absence of a suitable security scheme, ShareGear, designed primarily for ad hoc networks, implements the required security services. The security model of ShareGear can be viewed as a composition of two essential services – Authentication and Confidentiality. Authentication is provided by a challenge-response protocol, and Confidentiality through encryption.

4.3.1 Authentication Model

ShareGear utilizes a distributed authentication scheme. Authentication is performed on a peer to peer basis with each device responsible for authenticating any new entrant to the group. Upon authentication, the authenticated device attains a “trusted” status with the authenticator. A group is initiated with the selection of a password. The group initiator decides on a password that can then be communicated over in a written or verbal format to the

prospective group members⁵. New entrants to the group, upon entering the correct password, are individually authenticated with all other devices present in the group. In other words, the n_{th} entrant will be authenticated with $n-1$ existing members on a one to one basis.

The following few lines describe the flow of messages leading up to authentication. A brief description of the actions performed upon receipt of each of these messages follows. The Control Server is responsible for receiving and processing authentication messages in a serialized fashion. Upon receiving the password as an input from the interface, ShareGear sends out a “Hello” message broadcasting it’s presence. This is followed by the receive of “Challenge” messages from all members of the group. This message presents a challenge to the aspirant member. The aspirant member uses a key derived from it’s password to decrypt the challenge and send a response back. Response is in the form of a “Response” message, followed by which the newcomer presents it’s own challenge to the authenticating device. The reverse authentication process follows in the same manner and serves to authenticate the authenticator. Upon determination of a valid response, the authenticator replies with a “Verify” message to indicate that the authentication was successful and the authenticated device is now “trusted” by the authenticator.

- 1 Hello: The Control Server during initialization launches a thread that sends out three consecutive “Hello” messages to broadcast it’s willingness to join an already established group. There is a delay of one second between the consecutive “Hello” messages. Testing has revealed that the initial broadcast may be lost if the receiving card is in a low power mode. Sending multiple broadcast messages increases the probability of being received. A receiving device accepts an incoming “Hello” as a request to authenticate an aspirant member and follows by sending it’s challenge. A receiving device ignores further “Hello” messages once authentication has been initiated by the receipt of a previous such message.
- 2 Challenge: The challenge is an encrypted random number. This 128 bit random number is generated cryptographically and hence reliably random. Upon receiving a “Challenge” message, the receiver device decrypts the challenge with a key that was generated from the password. Specifically, the key is generated by hashing the password and then cryptographically deriving the key. The decrypted result is then returned with a “Response” message. The aspirant member will then send it’s

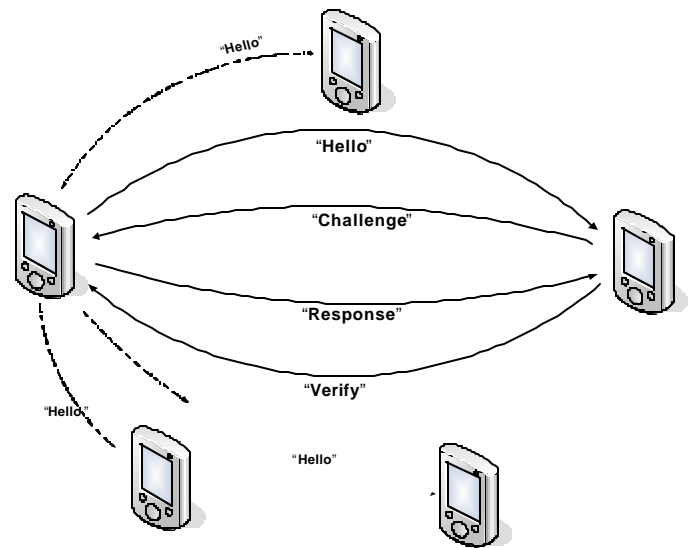


Figure 4. Authentication involves exchange of messages between the authenticator and the authenticated. The device broadcasts it’s presence with a “Hello” message. A “Challenge” is then presented. Upon successful “Response” to the challenge, the authenticator is ensured that the device can be trusted. A reverse authentication is also performed in parallel for mutual authentication.

- own challenge to authenticate it’s authenticator. This is essential for establishing mutual trust.
- 3 Response: A “Response” message contains the unencrypted random number presented with the challenge. The receiver uses the unencrypted random number contained in the “Response” message to determine if the sender is in possession of the shared secret. If the received number is the same as the one that the authenticator had presented in an encrypted format in the “Challenge” message, the authenticated device is considered “trusted” by the authenticator.
- 4 Verify: Upon establishment of trust with the authenticated device, the authenticator sends a “Verify” message to the authenticated device to confirm successful authentication. The authenticated device can now proceed to open it’s client TCP socket.

4.3.2 Confidentiality Model

Confidentiality primarily applies to the sharing service. In this case, confidentiality to file transfers is provided through encryption of the payload in every packet. The payload may consist of file transfer information necessary to identify the transaction that the packet is being sent as a part of. The unencrypted part of the packet only reveals the type of payload and it’s unencrypted size. This is required for proper classification and verification of the received packet by the receiver. The same key, earlier derived from the

⁵ Alternatively, a barcode can be used as a group password, on devices with inbuilt scanning functionality. The barcode can then be passed around to enable the members to join the group by scanning it into the password field.

shared password, is used for encryption and decryption. ShareGear depends on WinCE for its Cryptographic services. WinCE supports a sophisticated Cryptography API (CryptoAPI), an abstraction layer that provides the developers with a uniform gateway to various Cryptographic Service Providers (CSP) that may lie underneath. CSPs implement various cryptographic operations that may include encryption, decryption, hashing, signature generation etc. CryptoAPI hides the details of these operations, thus making the application, to some extent, independent of the type of CSP being used. WinCE (v 3.0) supports only two CSPs: Microsoft Base Provider and Microsoft Enhanced Provider [R10]. ShareGear utilizes the Microsoft Base Provider for its cryptography requirements. This includes encryption, decryption, hashing, key and random number generation. Specifically, DSA Secure Hash Algorithm has been used for hashing and DES (56-bit) for encryption-decryption. A higher bit secret key encryption is available only through the Microsoft Enhanced CSP. Microsoft Enhanced CSP is not available on PPC2002 by default and requires an add-on installation. ShareGear, in order to avoid separate installations of the Enhanced CSP, bases its cryptographic services on the Base CSP which is present on every PocketPC 2002 device.

4.4 FILE SHARING

ShareGear supports sending and receiving multiple files simultaneously. As earlier discussed, file transfer is performed over TCP sockets. A two-way file transfer requires two TCP sockets – one for sending and the other for receiving. A pair of TCP sockets is maintained for every member in the group to facilitate a two-way peer to peer connection. Moreover, members can be sending and receiving multiple files simultaneously. Multiple threads and state maintenance is required to afford multiple transfers over the same set of sockets. File sending operations to each member are handled by a “Send File” thread devoted to that member. However, receiving on all the sockets is performed by a single thread. Further, state is maintained to keep track of all transfers. The group data structure on each device maintains send and receive queues that hold state information on the progress of a transfer at any time. The details of the Send and Receive operations are discussed below.

4.4.1 Sending File

Each member maintains a thread responsible for sending files, for every other member in the group. This “Send File” thread is created at authentication time and is removed only when the member leaves the group. The thread stays dormant until a file to be delivered is queued. Each of these threads also has a queue associated with it. If multiple files are being sent by a sender, they are queued. The thread is activated when one or more files are queued, and the files

delivered in the order that they are queued. Once the queue is empty, the thread suspends. The system is capable of delivering multiple files from a single sender simultaneously, however, to allow for a more uniform distribution of the bandwidth among users, this option is not presently made available to the user. Thus this implementation allows for multiple files to be queued. However, the files are delivered sequentially, one at a time, in the order that they are queued to the receiver’s queue.

4.4.2 Receiving File

ShareGear receives all files simultaneously through a single thread, instead of maintaining a file receive thread for each device, or for each file. This thread, created during ShareGear initialization, actively listens for any receive requests. A receive request is made by the sending device in a specific protocol: the sender first sends a “Request Header” containing size and “Global Identifier” information of the file that it intends to send. (The purpose of this identifier has been discussed in detail in the following section.) The receiver then allocates space for the file to be received and sends its approval back. The sender is then free to send the file, a block at a time.

4.4.3 Shared Repository

The Share folder, a directory that exists on every member device, is a repository for the files that the virtual shared drive holds for its members. The Share folder is created upon initialization and stays behind with its contents, even after the application exits. If a Share folder already exists upon application launch, the files in the folder are ignored and only those files that are added in the current session are considered part of the Share. To share a file, a user must move it into the Share folder through the ShareGear interface. Sharing files in this manner simulates uploading a file to a shared network drive. The shared file is replicated to the Share folders on all trusted devices. With the present implementation, once a file is shared, it cannot be removed from the Share. Any modifications to the file must be performed outside the Share folder. This modified file must then be dropped again into the Share to be made available to other devices. Each file, when shared, is assigned a Global Identifier. This identifier is a 7 to 9 digit number and is unique to every shared file. The identifier is also unique to any shared instance of a file. In other words, the same file shared for a second time will receive a new identifier. A newer file, with the same name and a different identifier, overwrites the older one. The Global Identifier is computed as described below:

$(\text{last octet of the IP address} * 1000000) + 0 \text{ £ n } \text{£}32767$

The above method restricts the file shared by any member to 32767. This numbering ensures that all shared files have a unique identifier

which can also potentially be used to determine the source of the shared file.

5 PERFORMANCE EVALUATION

ShareGear's performance was measured in three respects: Application Initialization time, Authentication time and File Transfer time. All testing was performed over Symbol PPT2846 and PDT8146 PocketPC 2002 handheld devices running on an Intel StrongARM 206 MHz processor, and with inbuilt Symbol LA4121 802.11b adapters.

Application Initialization Time: ShareGear takes approximately 1500 milliseconds to initialize. At first this number sounds pretty high, however, the delay is carefully hidden behind the password prompt. It can be safely assumed that the user will spend at least a couple of seconds to enter the password. The expensive part of the initialization is over by then. To the user the application instantiation is almost instantaneous. The 1500 millisecond delay includes the 1000 milliseconds sleep time to ensure the uniqueness of the IP Address through Gratuitous ARP. Further delay is caused due to the log messages which may take approximately 100 milliseconds per message.

Authentication Time: ShareGear is based on a UDP message authentication. The authentication starts when the first "Hello" message was sent out and ends when the TCP client socket is opened in response to a "Verify" message. Based upon the above criteria the measurements resulted in an average authentication time of approximately 650 milliseconds. Since this number indicates one authentication and the system requires $n-1$ authentications on one device for a group of n members, where the n th member is joining the group, authentication is an expensive task. Further testing revealed that major delays were being caused due to the extensive logging that ShareGear undertakes. Logs are written to the screen and a file. Each such write takes around 100 milliseconds per line of log written. Removing these logs during authentication reduced the authentication time to around 180 milliseconds. Furthermore the power settings of the adapter also have a significant effect on the authentication time. Turning off the power save features on the adapter further reduced the authentication time to around 110 milliseconds.

File Transfer Time: File transfer time, in essence, is the throughput of the infrastructure. Throughput measurements on any wireless network must be performed in a controlled environment. Since performance is relative to a number of factors including number of

devices and noise in the environment. Our purpose was not to measure the throughput of the ad hoc network. Much work has already been done in measuring performance of ad hoc networks [R1]. The purpose was to determine the overhead imposed by threading, state maintenance and encryption. File transferring of ShareGear was compared against peer to peer TCP transfers over WLAN Infrastructure and Ad hoc modes. Peer to peer file transfers were performed using FTP and a TCP file transfer utility based on a design similar to that of ShareGear. This utility did not use encryption. Further, all tests were performed by sending or receiving a 10 MB file. ShareGear consistently delivered the 10 MB file to a peer in approximately 188 seconds, whereas an ad hoc FTP transfer based on the vxFTP client and server took only 110 seconds. As a relative measure, a peer to peer transfer using our TCP transfer utility took consistently approximately 92 seconds. The encryption and state maintenance overheads of ShareGear are modest. Testing also revealed that power saving features that may put the card in a low power mode can result in a multifold increase in this transfer time.

6 RELATED WORK

Over the last few years, considerable progress has been made in connecting handhelds. Handheld Operating Systems lack in providing a "Network Neighborhood" feel to the users who are accustomed to it from the desktop world. The mobility needs of the industries have become so specialized that they are no longer met by the functionality provided by the Operating System. Applications have stepped in to fill this gap between the connectivity technology and limited operating system functionality.

Serial connectivity, the only connectivity option in the earlier devices, is gradually losing its importance against the newer technologies. IRDA communication also gained and lost popularity within the last few years. IRDA is strictly one to one beaming and is inherently slow. Like IRDA, Universal Serial Bus (USB) connectivity, though faster than plain serial communication, is limited by its physical reaches. 802.11 and lately Bluetooth are the buzz-words in today's handheld market. Bluetooth configuration can be a repelling factor when setting up such networks to be of any practical use by other applications. 802.11 can now allow bandwidths up to 54Mbps. 802.11 also provides for ad hoc connectivity between devices. However, the unavailability of a DHCP Server on an ad hoc network makes its configuration too cumbersome for a network to be setup on demand in a reasonable amount of time. In general, present day handheld Operating Systems do not smoothly hide the details that go into configuring these technologies.

An interesting implementation based on a client-server model is Ipra*Cool, a commercial product [R7] that has recently appeared in the French market. Ipra*Cool allows for creation of secure groups called "tribes". These groups can share files and other applications among themselves. Ipra*Cool differs from ShareGear in requiring a central server responsible for security and sharing other services. WShare is another system sharing several design concepts with ShareGear and has been designed for desktop based ad hoc networks [R2].

7 CONCLUSION

ShareGear goes a long way into fulfilling its goal - to provide a secure and reliable infrastructure for sharing resources. However, as with any system, there is a considerable opportunity for enhancement. A more robust implementation along with additional services can be seen as a next step in this direction. The Authentication system, based on messages, can be made more efficient by piggybacking messages between the same pair. The Authentication model can be made more robust by adding resistance to conditions where a device only gets authenticated to a subset of the group. Similarly, the File Transfer operation can be improved by utilizing a reliable broadcast or multicast as the transport. ShareGear is a scalable concept that can be extended to add new sharing services. One such resource feature, Screen Sharing, can be an asset if made available through ShareGear. Additional network technologies can be integrated into ShareGear to further expand the infrastructure and allow sharing even on different underlying technologies. For example, ShareGear can be extended to create a network over an existing 802.11b subnet. In the above scenario, adapter configuration will not be required. However, security and sharing can be scaled to an existing network.

Embedded and Handheld computers are slowly penetrating into our lives. However still far from competing with the desktops, handheld are truly mobile computers and so have their own place in the Information Technology revolution. This paper brings forward a relatively new concept in the world of mobile computing - Resource Sharing. Modern day handheld computers, even when connected to a network, are isolated from each other. ShareGear introduces a sharing infrastructure, something not available even on the newer handheld. Available sharing mechanisms have been rudimentary and specific to the shared object. Moreover security is still a foreign concept in the handheld world. ShareGear, by presenting a secure File Sharing service, addresses both the sharing and security requirements. 802.11 Ad hoc network capabilities have been under utilized due to a lack of user familiarity and lack of vendor support. ShareGear taps into the

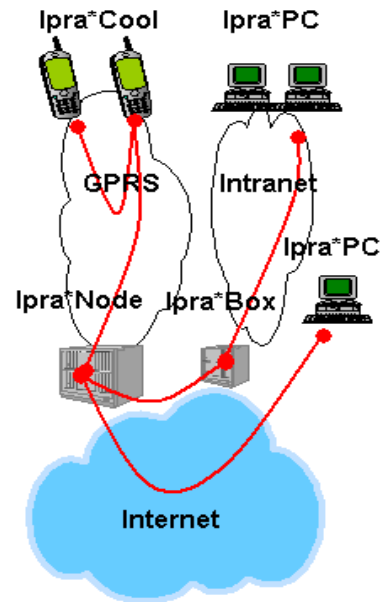


Figure 5. Ipra*Cool is a server based application sharing system that is scalable over LAN and WWAN. However, Ipra*Cool lacks in the dynamic nature of ShareGear. The system requires several components to be configured before it can be used. Groups are not dynamic and are created on the server.

infrastructure independence of ad hoc infrastructure by layering its secure infrastructure over it. This provides a dynamic secure network idle for rapid deployment and a File Sharing service that utilizes the underlying secure infrastructure. In doing all the above, this paper presents a new class of applications intended towards handheld sharing.

REFERENCES

- [R1] Jinyang Li, Charles Blake, Douglas S. J. De Couto, Hu Imm Lee, Robert Moris. Capacity of Ad Hoc Wireless Networks. M.I.T Laboratory of Computer Science.
- [R2] Ashish Raniwala. WShare: An Instant File and Application Sharing System over Wireless LAN. Department of Computer Science Department at Stony Brook.
- [R3] Microsoft Developer Network (MSDN) Collection. Available at <http://msdn.microsoft.com>
- [R4] Windows CE.NET Product Information. Available at www.microsoft.com/windows/embedded/ce.net/evaluation/
- [R5] Douglas Boling. "Programming Microsoft WindowsCE, Second Edition". Microsoft Press, 2001.
- [R6] IEEE Std 802.11-1999, IEEE Standards for Local and Metropolitan Area Networks: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications
- [R7] IPRACOM website: <http://ipracom.nerim.net/ipracom/index.htm>
- [R8] Microsoft Message Queuing Information Site. Available at: <http://www.microsoft.com/windows2000/technologies/communications/msmq/default.asp>
- [R9] Behavior of Gratuitous ARP in Windows NT 4.0. Microsoft Knowledge Base Article – 199773. Available at: <http://support.microsoft.com>
- [R10] Creating a Secure Windows CE Device. Available at <http://msdn.microsoft.com>