Secure Personal Information Management between Desktops and Handheld Devices

Gaurav Gupta and Kavi Mahesh

***Abstract*—With increasing popularity of mobile devices they have become more and more ubiquitous. These devices have become personal information warehouses which can support many interesting applications. We review some advantages and potential problems which arise in using mobile devices to manage personal information and how they are addressed by the proposed solution. A framework is presented describing the architecture of the solution, the various components which comprise it and the resulting procedure which achieves the desired effect. In the end some important design decisions and tradeoffs which were made are examined.**

***Index Terms*—Automatic form filling, handheld devices, information security, personal information management.**

I. Introduction

Most computers today are shared between multiple users. A typical family PC is shared within a house and even personal laptops are often accessed by multiple people. However a mobile phone that typically has features of a Personal Digital Assistant (PDA), also known as a "smart phone," is a very personal device. Not only does it remain with its user all the time but its use by other individuals is generally restricted. Although security is generally a non-issue in such devices, if required, encryption can be used for sensitive data such as medical information, financial account numbers and credit card data. Above points indicate that a mobile phone can thus act as a personal information warehouse for an "anytime, anywhere" access paradigm. However this information may be frequently used in a PC, for example in browsing the Web or in carrying out on-line transactions. Trying to synchronize or access the information on the mobile device from the PC can be a frustrating experience. A good architecture for transfer, synchronization and most importantly automatic utilization of the said data is thus required. We examine a particular case of above as "web page form filling." We then make additional suggestions for the above application before concluding how the same can be adapted to a generic scenario.

II. Related Work

Desktop based software such as RoboForm, iNetFormFiller, FormAutoFill are available and can be used for filling out forms on a web browser (e.g., Internet Explorer) albeit only those fields which a user has specifically marked earlier with chosen data. Moreover, they do not support data management on the phone or its automatic transfer to a PC when required. No solution similar to the proposed comprehensive application yet exists.

Manuscript received December 29, 2008.

Gaurav Gupta is currently a student at the PES Institute of Technology, Bangalore 560085, India (email: gauravguptaster@gmail.com).

Dr. Kavi Mahesh is a professor in the Computer Science Department, PES Institute of Technology, Bangalore 560085, India (+91 98452 90073, e-mail: drkavimahesh@gmail.com).

III. Proposed Methodology

If a user with a mobile phone enters the vicinity of a PC containing the proposed solution, she should be automatically signed in. If a user leaves the vicinity, she is logged off, completely removing any traces of her activity on the computer. The user should not have to manually carry out any activity pertaining to authentication (unless explicitly desired). If multiple users are present, then the appropriate user should be identified and signed in. When a user opens a browser window and navigates to any web page that contains a form, the software automatically identifies if there are any relevant fields, correctly identifies the context of the field and fills it with appropriate data taken from the phone. However, irrelevant fields like an advertisement box containing a text box for email address should be detected and ignored. The entire setup should be feasible with existing commonplace technologies, both on the phone and the PC. It should not create any complications or cause issues with form fill up. It is also desirable that the entire communication and any stored data are securely encrypted. For sensitive fields like those requiring credit card numbers, there should be an additional layer of security. Priority should be given to conserving battery life of the mobile phone and allowances made for any data transmission errors.

Based on the proposed solution, we have created a comprehensive application suite called "MobiFill". Although currently a prototype of work in progress, it successfully implements many of the features mentioned above.

MobiFill was designed with the following goals in mind:

1. Support for a wide range of devices irrespective of the underlying firmware or OS.
2. Automatic detection of users as they enter and leave the vicinity with handling of multiple users in an area.
3. Conservation of battery life.
4. Assurance of complete data security and user privacy.
5. Seamless interaction between the device and the PC with little or no user intervention.
6. To be transparent and simple.

Below is a short description of the software:

The MobiFill PC Application (henceforth referred to as PC Application) looks for devices in its Bluetooth range that

support MobiFill. When such a device is found the user is automatically signed in. The PC Application continuously monitors the device for its presence. If the device deserts the area, it continues to search for other devices. If a device is present and the user, after opening a browser window navigates to a form, the following actions take place:

- The Browser Helper Object (BHO), which is a browser add-on, parses the web page and carries out multiple detection routines.
- If the BHO detects at least one potential field, it requests the PC Application for corresponding data.
- The PC application establishes an OBEX FTP connection with the phone and retrieves the encrypted data.
- The data is verified for consistency and sent to the BHO.
- The BHO decrypts the data and carries out consistency checks to ensure that the fields being filled do not create any undesirable situations for the user (e.g. some pages refresh data when a user selects an item in a list box. Hence directly filling in the data in such situations can result in compatibility issues).
- BHO now starts to fill up the data in the fields.
- To conserve the mobile device's battery, MobiFill can also cache the encrypted data on disk; however the data is only decrypted when the mobile device is still signed in.

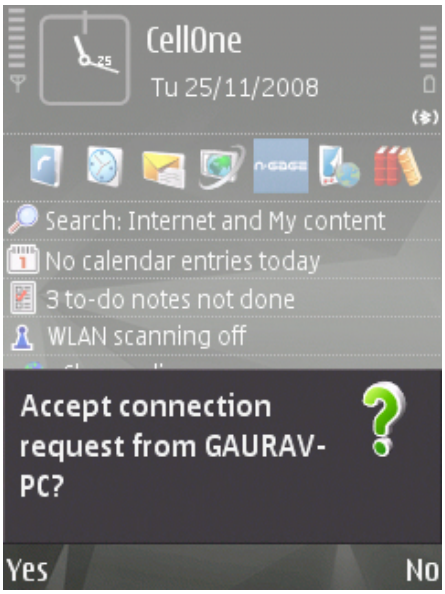

Figure 1: Data transfer check on Mobile Device

There are three levels of authentication in MobiFill:

1. Trusted PC: A trusted PC obviates prompting by the Mobile Device, and hence the data is directly sent to the PC whenever a form needs to be filled up.
2. Trusted session: The user is prompted by the Mobile Device for authentication only at the beginning of the session.
3. Un-trusted PC: The user is prompted by the Mobile Device every time a form is filled. (Please see Fig. 1)

To decrypt sensitive data the user must provide an additional password.

IV. MobiFill Components

MobiFill is composed of three major components:

A. PC Application
B. Mobile Phone Application
C. Browser Helper Object (Internet Explorer add-on)

A. PC Application

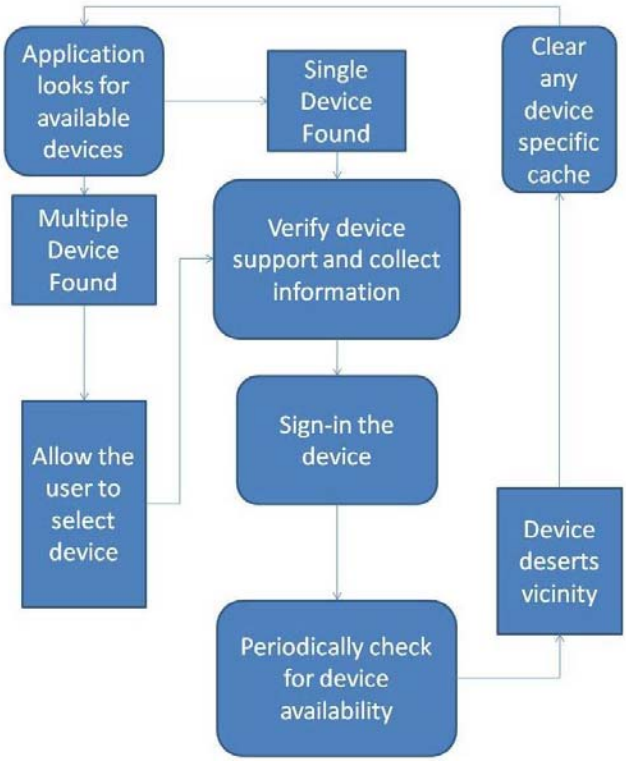

Figure 2: Architecture of the PC application (Part A)

The PC application is responsible for the following major functions (see Fig. 2 and Fig. 3):

1. Discovery and selection of device
2. Interaction with the user:
   a. Provide application interface
   b. Provide user configurable options
   c. Selection of device when multiple devices are present
   d. Activate or deactivate the application
3. Data retrieval from the device
4. Data transmission to the BHO

Procedure Description: The application looks for Bluetooth adapters present on the system. If no adapters are found, the application is deactivated. If multiple adapters are available then the primary adapter is selected. It then starts looking for compatible devices. Various checks including but not limited to platform support, OBEX support and application availability are carried out. If such a device is

found then the user is automatically signed in. If multiple devices are found then the user is prompted to select a device. An option in the application allows the user to force the application to give preference to a particular device. Additional refinements for selection of a device such as profile based look-up, customized group etc. can be introduced to enhance the end user experience. The device is continuously monitored for its presence. When a device deserts vicinity, all device specific cache is cleared. However the application may cache the device name and identifier for faster detection in future. Finally the user is signed off.

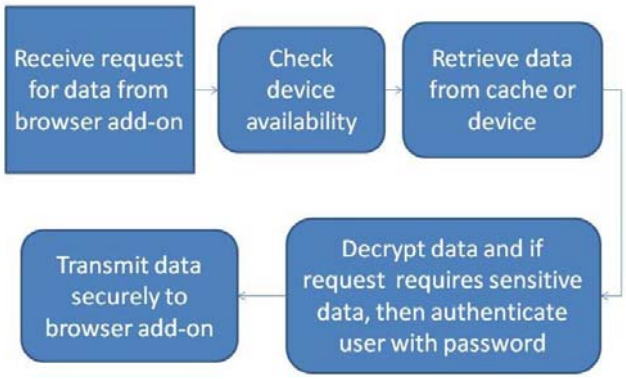

Figure 3: Architecture of the PC application (Part B)

The application presents a number of options (see Fig. 4):

1. The phone may be queried every time a form needs to be filled up, or the data may be retrieved once for each session.
2. A default phone may or may not be used.
    a. Action to carry out in case the default device is not available.
3. Additional confirmation may or may not be required by the PC Application when a form is filled up.
4. The choice to start the application when the operating system boots up.

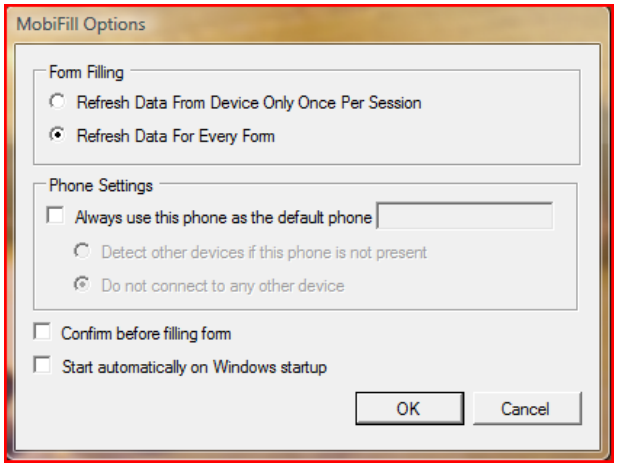

Figure 4: PC Application "Options" Dialog Box

B. Mobile Device Application

The mobile application is responsible for the following major functions:

1. Constructing appropriate user interface.
2. Encryption of data.
3. Saving information in the device data store.
4. The mobile application runs on demand when the user needs to edit the data; to conserve battery, the application otherwise remains closed.

Procedure Description**:** The application on start-up checks whether any user data is present. Available data is processed and checked for corruption or any inconsistency. In case of corruption the application recovers whatever can be salvaged from the data store. A form is constructed containing any previously entered data. The form may appear different on different devices since the interface is constructed on the fly based on the specification. The user can now enter or edit data. The information is appropriately encrypted before being saved. Since the solution utilizes OBEX ftp for data transfer, it obviates the need to keep the application running continuously; this conserves battery on the phone and permits support for devices with limited or no multi-tasking.

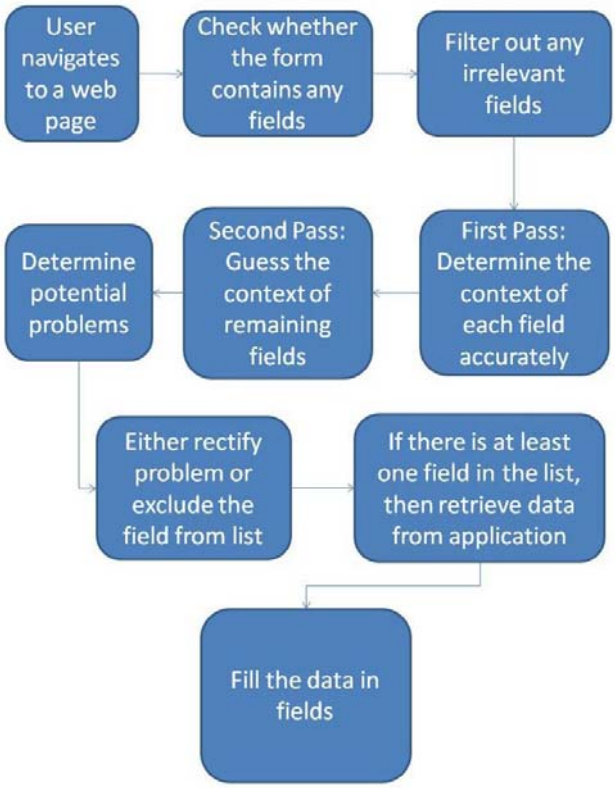

Figure 5: The Browser Add-On Chart

C. Browser Helper Object (BHO)

The BHO is responsible for the following major functions (See Fig. 5):

1. Parse web page.
2. Carry out multi-tiered detection routines.

3. Establish connection with the PC Application.
4. Consistency checks and compatibility routines.
5. Decrypt data.
6. Perform field fill-up.

Procedure Description: The BHO is a browser add-on with no user visible interface. An additional interface to the browser adds to the existing clutter (numerous toolbars/buttons). Further, it may confuse the user. To keep the entire process transparent, MobiFill has a centralized control panel.

The BHO parses each page. The initial phase performs a fast pass to determine if the main routine needs to start. This decision is made on a number of factors i.e. whether the form contains any field, whether all the fields have been rejected as irrelevant (for e.g., advertisement fields) etc. The main detection routine consists of a number of passes. The first pass performs an exact match on the semantic of the field with any available user data; if a field is not an exact match, the BHO checks its cache to see if the user had earlier filled such a field. Failing above, the second pass is performed which guesses the context of the field and assigns a percentage corresponding to the quality of match. Every field type has its own custom routine. All matches above a particular threshold are marked and added to the fill-up queue. Fields in the queue are examined to ascertain if they contain a better match to a field already filled up by user. If so, they are appropriately tagged.

Additional routines detect presence of any incompatibilities or issues that might result. These are either worked around if a solution exists (e.g., simulate a user selection of a list entry rather than direct select) or skipped altogether by leaving the particular field empty. Finally, the BHO requests appropriate data from the PC Application, decrypts it and fills up the fields in the form.

V. Conclusion and Future Work

The proposed solution and its implementation MobiFill allow a user to treat his mobile device as a personal information store with one of the applications being the use of this data for filling online forms. However the same data can be utilized for other applications as well as by other devices. It may also be noted that there is no requirement that the mobile device and the PC run on versions of the same operating system.

The architecture described above can easily be molded for any custom application that utilizes the above approach.

Additional features such as the ones mentioned below can be added to increase the versatility of the application:

1. Login and password management centre: Login/password information entered on various websites can be sent back to the phone and automatically stored there instead of on the PC. This allows for centralized personal storage and availability of login and password information everywhere irrespective of the PC or laptop being used. Another advantage of this approach is the evasion of spyware such as key-loggers on public computers.

2. Support for rudimentary phones that do not support custom applications: On such phones data can be transferred and saved via the PC. However such data cannot be edited on the phone.

3. Automated learning: If the application comes across a field which it does not recognize, then the application can make a note of it, record the information entered by the user and fill in the information when next time such a field comes up again.

In conclusion, as smart mobile phones are becoming universal personal devices, a solution such as MobiFill is essential to enable users to store their personal and confidential information on their personal phones and to manage the information securely, edit it on the go and use it transparently and seamlessly on any desktop or laptop computer. While just one application of such a capability in the form of automatic filling of web forms has been presented here, a variety of innovative applications can be envisioned in the future based on the architecture and core capability of MobiFill.

References

[1] Peter Thorsteinson, G. Gnana Arun Ganesh, ".Net Security and Cryptography", Prentice Hall, 2008
[2] Neal I. Koblitz, “A Course in Number Theory and Cryptography”, Second Edition, Springer, 1994
[3] Andy Wigley, Daniel Moth, and Peter Foot, Microsoft® Mobile Development Handbook, Microsoft Press, 2007
[4] Baijian Yang, Pei Zheng, and Lionel M. Ni, “Professional Microsoft Smartphone Programming”, Wrox, 2007
[5] Joe's Weblog, ASP.net Team, "Automating for Internet Explorer", available from http://weblogs.asp.net/joberg/archive/2005/05/02/405286.aspx
[6] Jia. C, “BHO with C#” available from http://www.codeproject.com/script/Articles/Article.aspx?aid=19971
[7] M. Jakobsson and S. Wetzel, “Security Weaknesses in Bluetooth” available from http://www.rsasecurity.com/rsalabs/staff/bios/mjakobsson/bluetooth/bluetooth.pdf
[8] B. Miller, “IEEE 802.11 and Bluetooth wireless technology” available at http://www-106.ibm.com/developerworks/wireless/library/wi-phone/
[9] Bluetooth SIG, Specification of the Bluetooth system, Profiles Version 1.1, 1 February 22, 2001, available at http://www.bluetooth.com/
[10] Bruce Hopkins and Ranjith Antony, "Bluetooth for Java", Apress, 2003
[11] Andrew Troelsen, Pro C# 2008 and the .NET 3.5 Platform, Fourth Edition, Apress, 2007