

Airscanner Vulnerability Summary: Windows Mobile Security Software Fails the Test

By Seth Fogie, VP [Airscanner Corp.](#) August 14, 2006

Microsoft claims that the Windows Mobile operating system is secure enough for the enterprise. That's not quite true, since unlike Windows XP, handhelds don't have advanced security architecture. For example, Pocket PC has no Kerberos authentication, Encrypting Filesystem, or a built-in firewall. In fact, even the much-touted Mobile2Mobile "secure" signing process for .DLLs and .exes can be bypassed with a simple buffer overflow, thus potentially allowing malware to take over your device**.

However, once you understand limitations, you can then plan your Windows Mobile rollout more carefully. Fortunately, there is a great deal of 3rd party security software out there. Unfortunately, much of it is completely insecure. Sadly, Windows Mobile developers have not yet been held up to the same scrutiny as desktop software developers. For instance, you may think your 'encrypted' or 'secure' data is safe on a Pocket PC because the vendor stated as much, when in reality the data is insecure.

In this paper, we expose some weaknesses in 3rd-party security software for Pocket PC. Note that we are not assigning blame to any of the developers; in fact, some of them responded quickly and were eager to get feedback and to fix the bugs. On the other hand, some were angry, threatening, and even dismissive. For us, it doesn't matter if software has bugs. All software has flaws; that's why you should always use "layered" security. It is the responsiveness of a developer, and their willingness to fix the product, that helps us define a quality developer.

This is not an attempt to criticize any vendors. We selected the target applications at random using the search engines provided by reseller websites. We are also not disparaging the Windows Mobile platform. In fact, we love it and use it every day. We simply want to make it stronger, and more secure. And by raising user awareness, perhaps more people will pay more attention to how their data is stored. The principle of "security through obscurity" has long been a discredit.

Background

According to the 2005 Pointsec Mobile Usage Survey[1] an estimated 22% of PDA owners have lost their devices. Combine this with the statistic that 81% of those lost devices had no protection (e.g. PIN or encryption), and the problem just got worse. Yet the same survey indicates that 37% of PDAs have sensitive information on them, such as passwords, bank account information, corporate data and more.

If you think PDA security isn't a real subject, just consider the possibility that there is someone out there right now with your name, email, phone number, and birth date and more stored on a digital device that was just left in a taxi cab – not a comforting thought.

Thankfully, a security conscious person can find, download, and install a plethora of software that will help them remain productive, yet keep their data secure inside an encrypted file in the event the device is lost or stolen. On the surface, these programs are an excellent idea. Financial information, passwords, credit card numbers, and even project files can all be locked up and secured. In addition, passwords that are entered into the PDA for service oriented programs (e.g. remote access, email, chat, etc.) are protected from prying eyes using masking techniques so an attacker can learn that information. Unfortunately, as we discovered, more often than not the security mechanisms are nothing but an illusion at worst, or terribly flawed at best. The end result is that the user is placing their trust in a broken program that is insecure. This paper will address many of the issues we found and what you can look for when investigating the quality of your 'secure' program.

The Windows Mobile Obfuscation Shell

Before we examine the details of the flaws, it is important to understand the nature of the operating system. The reason for this is because it is our belief that Windows Mobile platform creates an environment conducive to poorly designed security software.

In contrast, if there is a problem on the Windows XP (desktop) operating system, it is fairly easy for you to find out what is happening. For starters, a Ctrl-Alt-Del will allow you access to an informative Windows Task Manager that provides all sorts of information about the programs running on the computer. In addition, it is simple to find out what is configured to run at startup via the 'msconfig' command. Next, you can look inside the registry with 'regedit' or use the command line to quickly access and view files. And if this isn't enough, there are many free tools available that can expose almost anything about the operating system to its owner. All in all, thanks to certain tools, Windows XP is a fairly open operating system.

Now, what kind of details can you find out on the Windows Mobile 5 platform? For starters, the Task List only mentions the names of the open applications that have graphical interfaces. All others are not listed! How can a user find out if there is a hidden program that is eating up memory? Is there a way to find out what executes when the device is rebooted? Not for the average user. In fact, the only way a user can examine what is occurring behind the scenes is via the Visual Studio 2005 program that runs on a desktop system – and only if the PDA is synced up to that same system. There are some third party programs that give access to some of this data, but these are not free or as informative as Visual Studio.

The point is this – average Windows Mobile users are relatively blind about what their device is doing. As this paper will illustrate, there are numerous Windows Mobile

vendors that store sensitive information in the registry with flawed encryption schemes, or even in plaintext! If the end user knew anyone could see this data, what would they say?

History has taught the security community that software vendors will not code secure software unless forced to do so by consumers. The Pocket PC software market is a prime example of this 'law', which is why Airscanner performed this research. No more excuses...

The rest of this paper will be examining many different programs and their flaws. As you will see, blindly trusting a software vendor to keep your data safe is very risky. We hope that our research will help convince you to thoroughly research a product before relying on it to keep you secure.

Protecting the Passwords

When you use a program that requires a password, you assume it will be kept secure. This assumption is dangerous, especially on a Windows Mobile device. Typically, third party passwords are not encrypted. If they are, then it is a fairly simple matter to crack many of the encryption methods, thus exposing the original value. In this section we will highlight how you can find these passwords, with numerous examples to prove the point.

There are several tools that will assist in your registry viewing. The first is the registry viewer included with Visual Studio. This program is not free, but you can obtain a 120 day trial version from Microsoft's website. To augment this program, we also used an internal (Airscanner) tool that dumps the entire registry, and a free program called PHM Registry Editor (<http://www.phm.lu/Products/PocketPC/RegEdit/>).

Plaintext Passwords

The first group of examples stores the user account information in plaintext right under their registry key in the HKLM\Software or HKCU\Software branch. Figure 1 illustrates how a program called Verichat stores your user information.

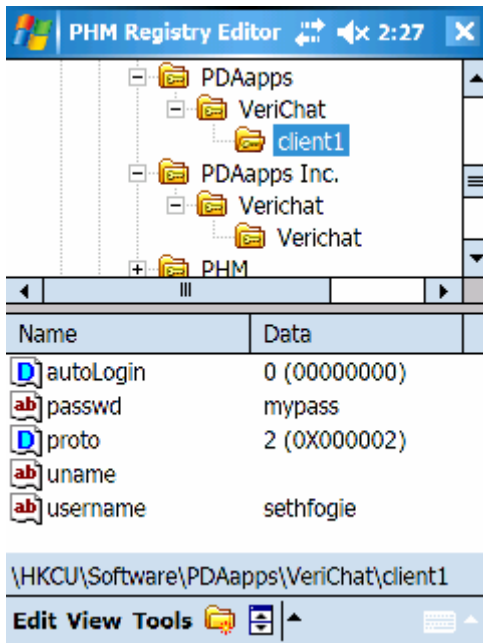


Figure 1: VeriChat User/Pass storage

If you note, both the username and password are very simple to read.

The following is a list of programs that were examined and found to have similar issues. Some store the information in the registry, and others simply keep it hidden in a configuration file.

- Verichat – Chat program
 - HKCU\Software\PDApps\VeriChat\client#
- IM+PPC – Chat program
 - \Program Files\IMPlus\implus.cfg
- Agile – Chat program
 - \HKCU\Software\AgileMessenger
- MSN Messenger Force
- Imov Messenger – Chat program (Enterprise version is encrypted)
- File Transfer Anywhere – File transfer program
 - \HKLM\Software\TTXN\File Transfer Anywhere
- NeoFTP – FTP client
 - \Program Files\neoFTP\FTP_Hosts.lst
- Thunderhawk – Web browser
 - thconfig.txt
- RemoteKeyboard – PC to PDA keyboard
 - \HKCU\Software\TransCreative\RemoteKeyboard\PassCode

The above list represents those products that do not protect the user information. The key thing to realize is if someone was able to gain access to a PDA for even a few seconds, the listed registry entries could be quickly viewed or copied out to an external memory card.

Password Exposure Bugs

To help protect against such easy attacks, some programs do encrypt the user information. Unfortunately, these protections are sometimes flawed, which results in exposed account information. This can occur either through a software bug, or by implementing a weak/flawed proprietary method of encryption. The following illustrates a few examples.

- **BullGuard Antivirus**

BullGuard is an antivirus program that requires a valid account to update the virus database. Each time the update occurs, the AV software sends the email address and password used to register the software via an encrypted channel to their server. This protects that information during transmission. Unfortunately, a weak encryption scheme is used to protect that password that is stored on the local device.

In addition to being able to decrypt existing passwords, we discovered that certain passwords are 'shortened' thanks to a flawed encryption algorithm. Figure 2 illustrates this bug. The highlighted data is where the encrypted password of 'sssssss' should be posted. Note that there is nothing between the semicolon and the 0x0D and 0x0A. As you can see, the password is basically blank! Unfortunately, this represents just one of many such defunct passwords that could be selected.

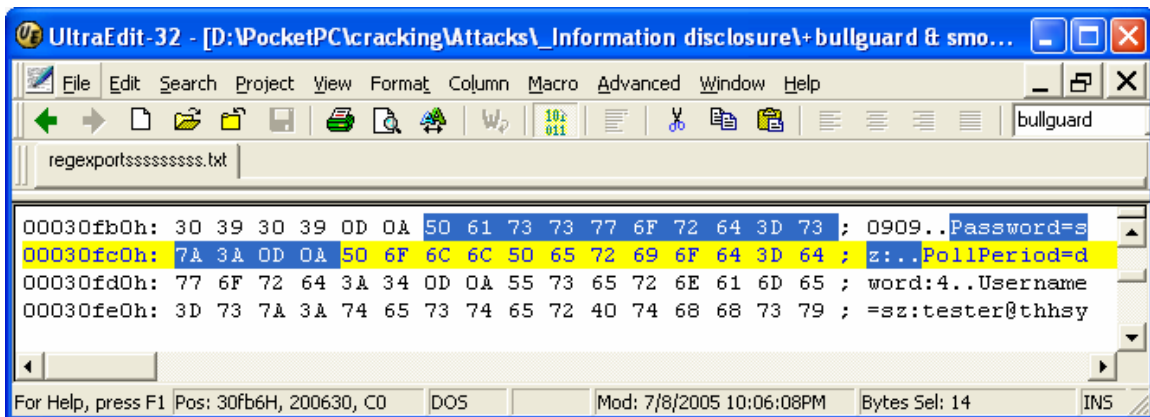


Figure 2: Bullguard Registry Entry

Although not related to password storage, it is important to note BullGuard stores its virus pattern matching information in a plaintext file that lists the virus and its pattern. For example, the following is the entry for the WinCE Duts virus.

WinCE-

Duts.A(frk)=04001be50fe0a0e128f01be508001be50fe0a0e128f01be53380bde854686973

The reason this is a bad idea is because a malicious program can simply patch the virus definition file with an incorrect value, thus ensuring it won't be considered a virus.

Secondly, BullGuard includes an auto delete function that could become an attack tool if malicious program inserted a pattern that matched all executable and dll files on the PPC (i.e. ReallyBadVirus=4d5a9000).

- **Abidia and OAnywhere**

The mobile device is an excellent tool for remotely monitoring services. In the case of Abidia and OAnywhere, this service is eBay.com and Overstock.com account monitoring. Once the PDA software is installed and configured, the application will poll the online auction websites for updates on items selling, buying, etc.

The dangers for this type of program are three fold. First, the user account information must be securely stored on the device. Second, if the program ever has to handle the sensitive data, then it must be able to ensure the confidentiality of that information during program execution. Third, the program must securely transmit the data to the service provider.

In the case of Abidia, the user information is stored in an XML file in the program directory. Fortunately, the eBay account data is encrypted (e.g. ebaypass="2F6DD0EEDA6168A7FE2A3AC47436A8720399FB4797DE422E"). After reviewing the encryption scheme, we determined that it appeared to be secure enough given the time involved to crack it. However, during this investigation, we discovered that the executable file itself could be used to decrypt the password. As previously mentioned, if a program stores a password it must maintain the confidentiality of the data at all times. In the case of Abidia, it was fairly simple to follow the execution path and hook into the program after it decrypted the password, which we then were able to display on the PDA's screen.

Finally, we examined the data communication process to ensure the user account information was securely transmitted. We discovered that the program interacts with an API interface on Abidia's servers, which serves as a proxy to eBay. The following is an actual capture of the plaintext HTTP POST request send from our Windows Mobile device.

POST

```
/api/get.php?user=sethfogie&pass=mypassword&serial=&imei=22363230F84031111800%2D0050BFE45CE5&site=US&dbg=y&name=buy HTTP/1.1
```

```
Host: api.abidia.com
```

```
User-Agent: Abidia-Wireless/2.5.3 (PocketPC; 240x320; WindowsMobile/5.1.70)
```

```
Accept: text/html
```

```
Content-Language: en-US
```

```
Connection: Close
```

```
Content-Length: 93
```

```
Content-type: application/x-www-form-urlencoded
```

In case you missed it, take a close look at the POST string. Abidia does not encrypt the user or password. Since this was all performed over a regular HTTP session, anyone in

the data transmission path – including Abidia – can capture the account information. It is dangerous enough to trust a third party company with user account information, but the fact the username and password are sent as plaintext is very insecure; particularly if you are using a wireless connection and/or a public hotspot.

- **Windows Mobile WEP Key**

The Odyssey client included with the original (WM2003) Dell X50v stores the WEP keys as an encrypted strings in the registry. When the network connection is made to the secure network, the driver pulls these values from the registry, decrypts them, and then incorporates the key into the communication process. However, during this process, the driver writes the decrypted value back into the registry. The problem is not Odyssey's, as that program does encrypt the key, but is instead a flaw in how all three (Windows Mobile, Dell wireless driver, Odyssey) work together.

The following illustrates: Byte 5 - 9 list my entered WEP keys for each entry.

KEY1=aabbccdee

"HTCWEPDefaultKey1"=hex:

01,00,00,00,**aa,bb,cc,dd,ee**,8c,f6,36,1d,af,90,17,5b,00,f6,36,1d,af,00,00,00...

After we notified the vendors, this problem has been fixed in current versions of Windows Mobile and there is a ROM update that will correct the problem for the Dell Axim X50v.

- **PocketMoney**

According to the website, "PocketMoney is the most robust financial management tool for the Pocket PC." With it, you can "Store the institution, phone, account number, expiration date, limit, fee for each account. Now you can even password protect your PocketMoney data from prying eyes!"

To keep the information safe, PocketMoney requires a user to enter a password before opening its data file. An 'encrypted' version of the password is stored in the registry at the HKLM\SOFTWARE\Handmark\PocketMoney>Password key. Unfortunately, the password is protected via a ROT-N function using the following seed value:

0x21 0x70 0x6d 0x6f 0x6e 0x65 0x79 0x21 → NAK p m o n e y NAK.

In other words, the protection of the password (and the financial data) is tied directly to the word 'pmoney' (sound familiar?). Despite the key selection, a ROT-N scheme is always a bad idea because it is trivial to do a pattern analysis on the encrypted data and deduce the key.

In this section we looked at several examples of how *not* to protect user account information. Unfortunately, this problem is wide spread through out Windows Mobile programs. Be sure you understand the dangers associated with trusting a program to keep your user account information secure, and always use unique passwords.

Data Protection Programs

This next section takes a look at programs that implement password protection schemes that are meant to keep data secure. Unlike the previous section that focused only on user account information, this section targets programs that were designed to store sensitive data such as banking transactions, stock information, credit card numbers, and lists of passwords. In this case, an attacker would have access to a much larger chunk of sensitive data that the user is assuming is secure.

Financial Management Programs

This section addresses a common problem that exists in numerous ‘secure’ programs. Although some programs obscure the issue, all of the following titles can all have their security mechanisms bypassed by a small change in the registry. Note how some companies try to hide this fact by placing the registry key in unusual locations, or by burying the flag inside a large registry string.

It should also be mentioned that a malicious user can often just copy the ‘protected’ data file off the target device and onto a device that has no protection enabled. Since the data itself is not truly protected, an alternate device will be able to open it without the need of a password.

- **PocketKeeper**

PocketKeeper is program to manage daily out-of-pocket expenses with multiple accounts different currencies, intuitive register, customizable categories, budget, multiple report charts, and password protection. It has two levels of security – a global level that restricts access to the program, and an account level that secures each account.

Upon reviewing the files associated with this program, it was discovered that both passwords are stored as plaintext in the .dat files of the program directory. Specifically, the global password is stored in config.dat and each account password is stored in its relative account file.

- **PocketMoney**

PocketMoney not only uses a weak encryption scheme to protect the password (discussed in previous section), but the protection scheme itself can be easily disabled by setting the following key in the registry to a 0.

HKLM\SOFTWARE\Handmark\PocketMoney\Active Password = 0

In response to this issue, PocketMoney's vendor rather alarmingly states, "The password in PocketMoney wasn't designed to encrypt data or prevent anyone other than a casual browser from being able to access the data. I suggest the user turn on the Palm's (sic) password protection if they want their palm (sic) secure." We, the users, beg to differ!

- **WebIS Money**

WebIS Money states it includes "...secure password protection to your data to safeguard it in case your PDA is lost or stolen." Unfortunately, this protection can be disabled by removing the following key from the registry.

HKLM\SOFTWARE\Microsoft\Pim\Outlook\IMAP Folders\H11

- **MoneyTracer**

MoneyTracer claims "Encryption of your data by your own password." While the password option is available, it only authenticates the user and does not actually encrypt any of the data, as claimed. To disable the 'encryption', set the following key to '0'.

\HKLM\SOFTWARE\Maction\MoneyTracer\bEnablePassword = 0

- **TinyStocks Stock Manager**

TinyStocks states "Stock Manager can be protected with a 4-digit PIN number." This PIN is stored as a four byte value within a preferences string in the registry. The following lists the location and provides a screen shot of the key with the password set/unset.

HKCU\Software\TinyStocks\Stock Manager\

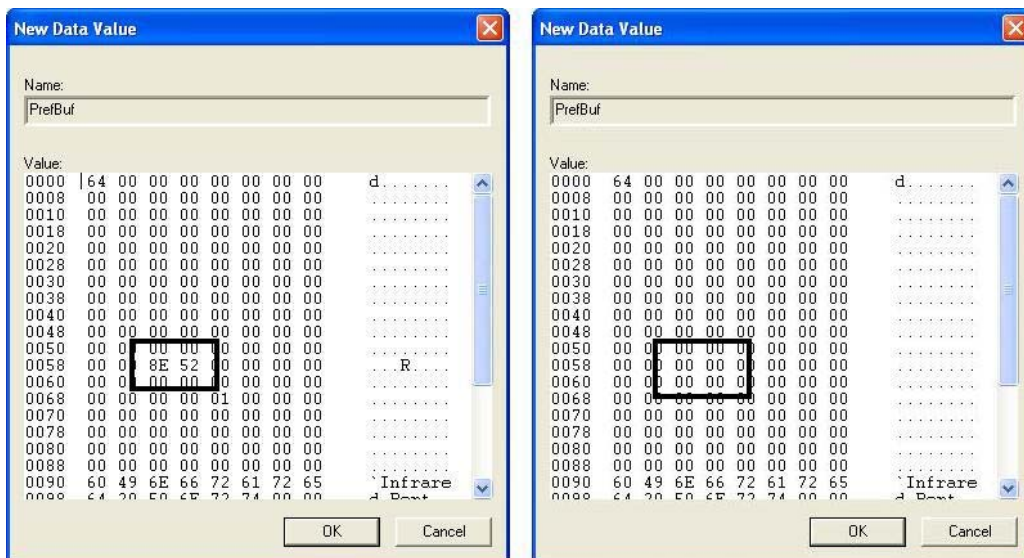


Figure 3: Screenshot of StockManager registry key

When asked about this issue, TinyStocks replied, “The password protection in Stock Manager is not meant to be secure but to stop casual access to the program. The data itself is unencrypted and so it's quite easy to just look at it.”

- **PocketExpense Pro**

PocketExpense Pro creates a .vol file that contains all its financial information. Included in the file are the settings associated with the password option. In this program, all the preferences are stored in a large hex string in the registry. However, it is possible to disable the password by changing the hex at 0x7D94 from 0xF4 to 0xD4.

- **Inspiration**

Inspiration is a project management program that uses ‘built-in security features’ to “...keep files from accidentally being modified when handhelds are shared between multiple users.” Therefore, it is fair to say that the password was never meant to offer any true security.

However, if an attacker wanted to remove the password requirement, they would only have to overwrite the encrypted password value that is stored in the project header. Specifically, bytes 0x95 – 0xA3 need to be set to 0x20 0x00 0x20 0x00 etc.

- **Microsoft Money for Windows Mobile 2006**

MS Money for Windows Mobile 2006 is a financial tracking program that can be used independently or with the MS Money application that runs on many desktops. The program can be configured to require a password when it is launched. However, this password does not encrypt the data, which is stored as plaintext in data stores in the Databases folder.

The password is stored in the registry at HKLM\SOFTWARE\Microsoft\Money2000 CE\Options\Display in an encrypted format. However, the encryption scheme used to protect the password from viewers is a weak proprietary algorithm and can be cracked using the following equation:

$$(((\text{encrypted byte} - \text{A0})/4) * 8) + 24\text{h} - \text{encrypted byte} = \text{password byte (all hex calcs)}$$

Finally, the password requirement can be nullified by deleting the key from the registry, which will cause the program to think the password option is not set.

Password\Credit Card\PIM Management Programs

The following programs are used to store sensitive information, such as password lists, web site login information, credit card numbers and more. Due to the nature of the data,

these programs need to be secure. If an attacker can access the 'protected' information, they will have gained access to a wealth of information.

As illustrated, the previous financial programs do not protect your data. Although most vendors use security as a selling point, in reality a simple registry tweak will allow anyone access to this sensitive data. Even the vendors admit their software is insecure and recommend alternative steps to secure the data.

- **Password Master 1.0 – Free version**

Password Master 1.0 allows you to “Keep all your passwords, Credit Card Numbers and other details in a single place. Carry your money or details virtually everywhere.” According to their website, “Since all the details you enter are sensitive data, the Password Manager helps you to create a Secure Login to the records. You can create a Master Password, which will work as your Master key for all the virtual locks you know.”

Unfortunately, if someone deletes the following key from the registry, the master key will be reset, thus allowing full access to the data.

`\HKEY_CURRENT_USER\Software\Data>Password Master\Pref\dt`

This version of the program is free. The vendor's website provides this tool, but also advertises their Password Master 3.5 version that requires a payment. We look at this version later in this section.

- **Passman 1.2**

Passman 1.2 is a password management program that can create and store a list of passwords. It includes an option for a startup password and also provides for '512bit encryption' of the data. Both protection measures can be cracked.

To bypass the startup password, a malicious user only has to set the startpasswdenabled registry key to '0'.

`\HKEY_CURRENT_USER\Software\passman\preferences\startpasswdenabled.`

However, if the database is encrypted, the actual data will still be secure. Unfortunately, the password used to encrypt the database is itself not properly protected. The following equation will decrypt the password stored in the registry, thus giving an attacker full access to the database.

Assume:

B is byte of password in hex

P is position of target byte (0-5 for this example)

$B - (25 - (3 * P)) = B_{plaintext}$
→ 26 23 20 1D 1A 17 = 111111

The end result is that the password option can be disabled, the password can be cracked, and the database can be decrypted by an unauthorized user.

- **Password Master 3.5**

Password Master 3.5 states it will “Keep all your passwords, Credit Card Numbers and other secured details in a single place. Carry your money or details virtually everywhere. Now includes a Free Desktop Companion!” In other words, it performs much the same function as CodeWallet Pro.

Ironically, like the previous example, Password Master 3.5 also does not encrypt its information using a unique password. Instead it relies on the user provided password to authenticate the operator to the file, and then decrypts the data using an internal algorithm.

Therefore, using the same technique outlined previously, an attacker only has to obtain the secure file and overwrite a few bytes of hex in the header to gain access to that file, and the ‘secured’ contents within. In this case, the hex range is from 0x2A - 0x5B.

In addition to the overwrite vulnerability, this program also was found to have a bug in the ‘hint’ feature that enables a user to obtain their password if they forget it based on a question/answer. However, if the user never configures the hint option, the program will give up the password regardless of a correct hint/answer combination. While this is a security risk, it is based on a software bug – not a broken security model.

It is important to note that Password Master 3.5 also includes a desktop companion that operates in the exact same way as its mobile counterpart. This desktop based program also suffers from the header overwrite bug.

- **CodeWallet 6.0.5**

CodeWallet is one of the premier programs that fall into the category of Secure Information Manager. It will protect your sensitive information, including credit cards, passwords, etc., in an encrypted file that a user decrypts with a password when opening.

During testing, we initially thought that CodeWallet used the same dysfunctional method of ‘encryption’ used by Password Master. However, CodeWallet looked into our report and commented that the while it was possible to open a file, all the data was still encrypted.

After further research, we found that when a Wallet file is created, its encryption is tied to the original password used to create the file. If the password is changed after this, it will only change the authentication requirements, and not affect the encryption.

Unfortunately, the My Sample Wallet included with the program comes with a known password, which an attacker can use against other files based on the Sample Wallet. As a result, anyone who used the Sample Wallet as a template to build their own secure Wallet is vulnerable to the header over write attack.

Miscellaneous Information Disclosure Bugs

Not all Windows Mobile related security problems are related to failed protection schemes. This section will outline several other program and bugs that were found during the research project.

- **Remote Keyboard**

From the vendors website, “Remote Keyboard is a program that connects PC keyboard and mouse to your Pocket PC over ActiveSync connection or TCP/IP network.” This is a handy program for power users who need to enter a lot of text into the PDA.

Once installed, the client on the PC sends out UDP packets containing an IP address to port 23 that are detected by a listener on the PDA. Upon detection, the PDA will connect back to port 8123 on the specified IP address. At this point the PC will query for the correct password, which is provided by the PDA application. Finally, the connection is made and the user can control the PDA remotely from the PC client.

We discovered a few problems with this program that can expose the password used to authenticate the connection as well as capture the clipboard contents of the PC. The first issue was discovered when we created a custom UDP packet that contained our “server’s” IP address and passed it onto the network. The Remote Keyboard listener on the PDA detected this packet, and immediately tried to connect to our computer on port 8123. Upon seeing this, we then created a small and simple ‘server’ that emulated the login process. As guessed, once the PDA had connected to the ‘server’ and negotiated the connection, it sent the ‘server’ the authentication password.

Using this captured password, we then telneted to the PC service running on port 8123 and discovered that the program dumped the entire contents of the clipboard onto the wire after a successful login. The following provides a screenshot of this bug.

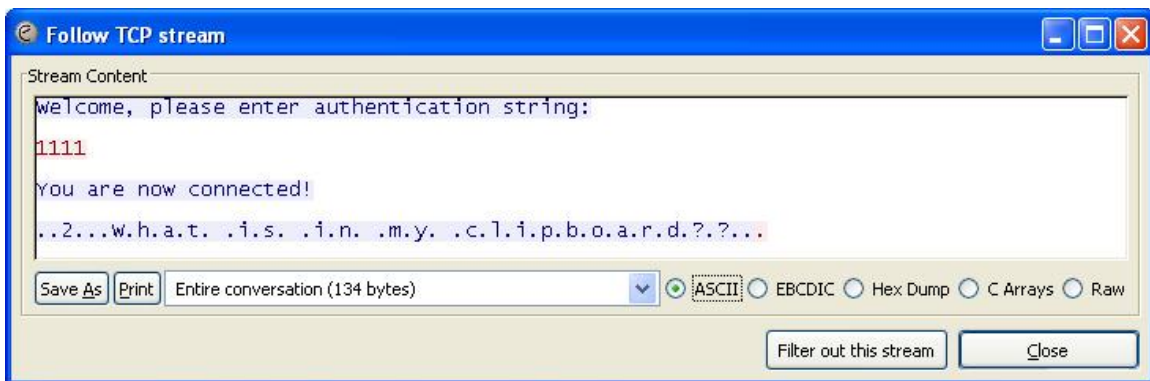


Figure 4: Remote Keyboard capture

- **ActiveSync 3.8**

ActiveSync is 'the' program used to sync a Windows Mobile device to a PC. It is the most-downloaded Windows Mobile software application of all time. Contained in this program are functions used to upload software, sync up emails, and much more. Version 4.0 and above have restricted any form of network based synchronization; however, as many users rely on this feature for their day to day synchronization needs, Microsoft still provides AS 3.8 as a download.

As we discovered in mid-2005, the AS3.8 service on the PC opens up port 990 on any existing interface (i.e. wired, wireless, PPP, etc.). This port allows access to the ActiveSync service, which can be abused to spawn a password box on the PC users screen (figure 5). If a user enters a value in this dialog box, the characters of the password are returned to the attacker, who can then use this data to gain access to the protected PDA or create a connection between an attacker's PDA and the target PC.



Figure 5: Spoofed spawned password dialog box

Suggested Fixes

As this document illustrates, there is a serious problem with regard to sensitive information and the handheld device. The following provides several suggestions as to how you can mitigate the risks we discussed.

Password protect your device

Windows Mobile comes with a password protection feature that will lock the device to unauthorized users. There are also third party vendors who provide a lock and wipe program that incorporates password protection with a memory wipe feature if the wrong password is used. However, it is important to note that a logon will not protect the data on external memory cards.

Encryption

All sensitive data must be secure using a known and proven encryption scheme/program. This is especially important for external media cards often used in PDA's. It only takes a second to remove a card from a PDA. We recommend you inquire as to the encryption scheme used. Windows Mobile includes a MS Crypto API that has so far proven to be

solid. While there could be others, programs that use this API are probably going to be secure.

Limit exposure

Given the statistics, it is recommended that PDA users limit the amount and type of data found on a device. Store files on different media cards, based on their function and only carry them with you when they are needed. By combine preventative security actions with reactive security fail safes (i.e. data wiping password programs), you can mitigate the security dangers even if the device is lost.

Use computer security common sense

The PDA is a hand held computer, and should be treated as one: do not download and execute untrusted software, use antivirus programs to scan/protect your device regularly, use a strong password and change it regularly, and disable unwanted services like Bluetooth. In short, employ the same precautions you would apply to your PC usage.

Summary

This paper examined a wide range of programs that expose your sensitive data. Some store sensitive data as plaintext, other improperly handle this data and give their users a false sense of security – and we are finding more all the time. As illustrated, there are numerous programs that advertise their ‘security’ features to earn your trust, but in reality fail to protect your data from attackers.

Hopefully this paper and the research that went into it will help increase the security awareness for the Windows Mobile community. It is in the best interest of everyone who loves Windows Mobile to advocate stronger coding and greater platform security.

**Airscanner will be releasing a paper illustrating this fourth quarter 2006.

[1] <http://www.pointsec.com/news/release.cfm?PressId=108>

Copyright (c) 2005 Airscanner Corp.

Disclaimer: The information in the advisory is believed to be accurate at the time of publishing based on currently available information. Use of the information constitutes acceptance for use on an AS IS condition. There are no warranties with regard to this information. Neither the author nor the publisher accepts any liability for any direct, indirect, or consequential loss or damage arising from use of, or reliance on, this information.