HOW SECURE ARE CURRENT MOBILE OPERATING SYSTEMS?

Tobias Murmann, Heiko Rossnagel
Chair of Mobile Commerce and Multilateral Security
Johann Wolfgang Goethe-University Frankfurt
D-60054 Frankfurt / Main, Germany
www.whatismobile.de

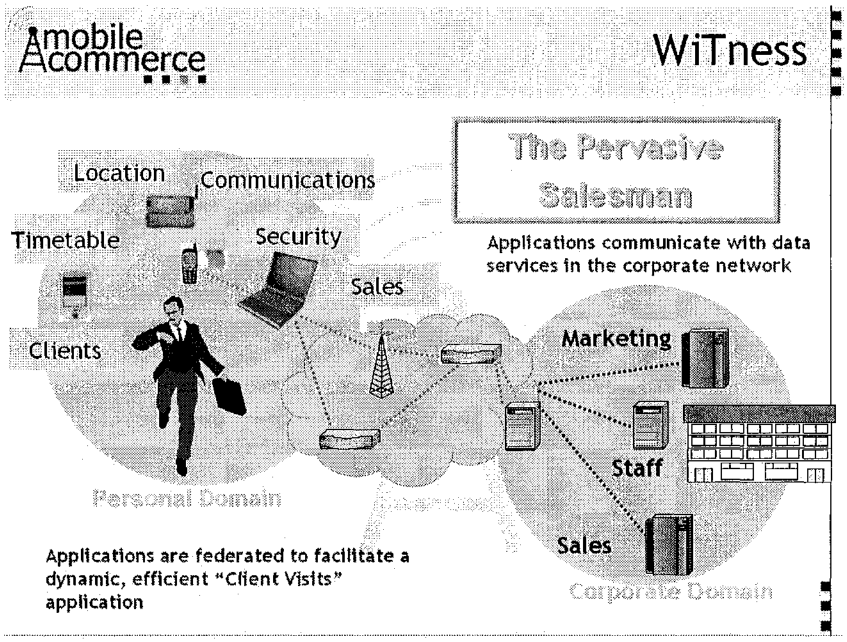
Abstract: There are numerous initiatives to use mobile devices as so-called "trusted pocket signers" to produce electronic signatures. The actual signature is generated by means of a conventional signature card. The mobile device serves as the card reader, storage device for the document to be signed and as a display for the signature application. The operating system used on the mobile device has thus a pivotal importance to ensure the integrity and accountability of the electronic signature. Also mobile devices are used to provide mobile workers with access to the corporate backend. We examined the currently available mobile operating systems in regard to their security and conclude that not a single one is secure enough for "trusted" signing and only partially for secure backend access. We show two possible ways of how to make mobile devices more secure and possibly to enable something close to "what you see is what you sign".

Key words: Mobile Operating Systems, Trusted Devices

1. INTRODUCTION

Mobile devices are becoming ever more capable and are able to open up a broader range of applications in professional environments due to their increasing functionalities. Personal Digital Assistants (PDAs) and Smartphones allow users to access sensitive personal data at any time and any place, making it possible to increase productivity. In the case of mobile devices carrying sensitive data like patient data, customer lists and address data, amongst others, the security of these data must be ensured.

Corporations are using mobile devices to enable their mobile workforce to get access to their backend. Since this company data can be very confidential the access to the backend must be secure. The WiTness project sponsored by the European Union [WiTness2004] aims to provide secure backend access by means of GSM technology. Figure 1 shows an application scenario where a "pervasive salesman" has secure, corporate-controlled access to all data available to him in the corporate information system. Access is controlled by a security module based on a SIM with additional security functionality.



Applications communicate with data services in the corporate network

Applications are federated to facilitate a dynamic, efficient "Client Visits" application

Figure 1: WiTness Pervasive Salesman Scenario [WiTness2004]

But even if the communication and access to the backend are secured, the mobile device itself remains open to possible attacks. If corporate data is stored on the device an attacker could try to circumvent the access control mechanisms of the device in order to get access to the stored data.

There are also some initiatives using mobile devices as so-called "trusted pocket signers" to produce electronic signatures [MobTra2004]. The actual signature is generated by means of a conventional signature card (according to the EC-Directive [EC_esig1999]). The mobile device serves as the card reader, storage device for the document to be signed and as a display for the signature application. Therefore, it must be ensured that the data shown on the display is identical with the data signed by the signature card (WYSIWYS)[1]. The operating system used on the mobile device has thus a pivotal importance to ensure the integrity and accountability of the digital signature.

If the authorization mechanisms, memory protection, process generation and separation or protection of files in an operating system are flawed, an attacker may gain access to the different internal processes. He might take advantage of this situation to generate forged signatures.

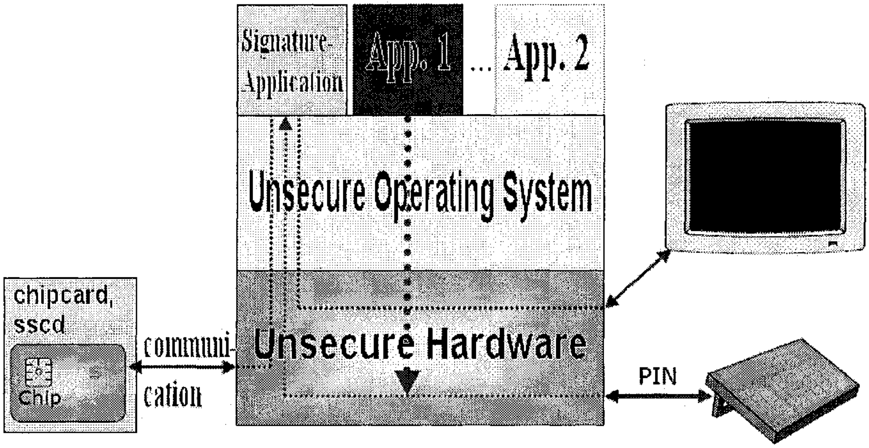


Figure 2: Manipulated digital signature [Federr2003]

Figure 2 illustrates that application 1 as a malicious program can intercept the PIN, for example. An even considerably higher risk exists, however, if the malicious application changes the data to be signed after they are displayed to the user. Due to the virtually unrestricted hardware access, a malicious program is able to manipulate all data transmitted to the signature application before the actual signature takes place.

We examine the current available mobile operating systems in regard to their suitability for both scenarios. Using the mobile device as a trusted pocket signer poses the hardest security requirements (especially in regard to accountability and integrity). From a business perspective the confidentiality of the corporate data seems to be the most important protection goal.

In section 2, operating systems that are currently available on the market are examined, and some important security flaws are pointed out. Section 3 presents a glance at the future and examines how these problems can be

[1] What You See Is What You Sign

solved by means of software or hardware solutions. In section 4, the results obtained are summarized.

2. SECURITY ANALYSIS OF CURRENT MOBILE OS

2.1 PocketPC

PocketPC [Pocket2004] does not provide the possibility to encrypt data. Even the internal communication is not secured. Due to its design, PocketPC neither separates memory blocks nor applications effectively from each other. Each application can adjust its priorities, terminate other applications, access their memory or prevent the switchover into the power-save mode.

Passwords can be deactivated by the user and are frequently deactivated in the standard setting. Also, an attacker can easily take out the external storage medium from the device and steal the data that is stored there. Even worse is the possibility to port malware onto the PDA in this way. This malware could later fake a signature as shown above.

Fake dialogs are possible because of malware. But even an uninfected PDA with PocketPC allows fake dialogs. As the Microsoft operating system supports Active X and Java, these can be used to create fake dialogs.

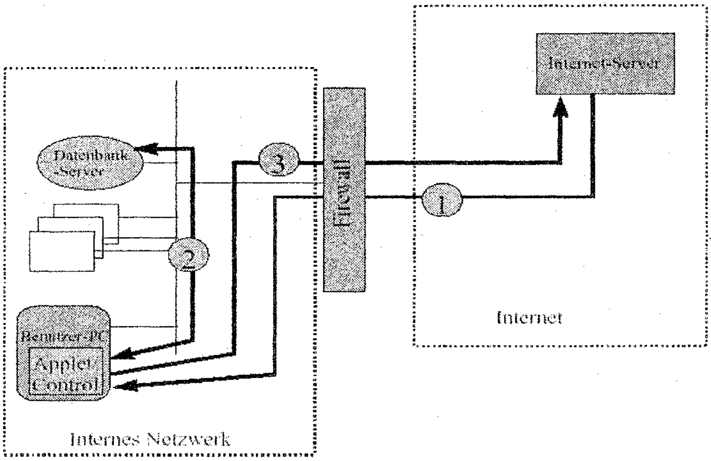


Figure 3: Mobile code attack scenario [FoxHor1991]

1. The user loads the applet (Java) or control (ActiveX) from a web server, which is then executed on the customer's mobile device.

2. The applet/control makes use of the owner's authorizations to gain access to the company's database, and copies data onto the mobile device.
3. The applet/control sends the data obtained back to the Internet server.

In case of a Java applet, the so-called sandbox restricts the applet's access to the hardware and software. However, the user may have granted the applet too many rights, or an attacker may use one of the many security gaps in the Java virtual machine. The user's security may be protected by a code-signing mechanism, with which the origin of programs can be certified. However, with this mechanism, only the origin of a program can be determined, but the actual contents can be harmful. But since the administration of certificates is not possible with PocketPC, any form of mobile code must be deactivated in the setting.

There is a theoretic possibility of hidden backdoors in PocketPC, as the source code is not open. A protection from buffer overflows and against the manipulation of the DMA functionality cannot be provided by means of additional software. Manipulated programs are able to act with all user authorizations, as there is no distribution of rights. PocketPC 2002 exhibits a large number of security gaps which cannot be closed completely by means of additional security software, such as PDA Secure [PDASe2004] or PDA Defense [PDADe2004]. Due to these security risks, PocketPC cannot be used as an operating system for a "trusted" pocket signer. Even for the scenario of the WiTness Pervasive Salesman in Figure 1, PocketPC should not be used. The impossibility of deactivating or bypassing passwords is an essential feature for this scenario. Furthermore, certificate management is necessary. Certificate management can't even be reached with additional software, which shows that PocketPC is only secure enough for private usage.

2.2 PalmOS

Like PocketPC, PalmOS [Palm2004] does not have an effective distribution of rights and separation of processes. There is no secure path between the applications and the kernel, and the communication is vulnerable. Furthermore, the user, as with all operating systems, cannot check if the status of the device is secure. This could be achieved by means of an LED, for example, which indicates if the PDA is in a secure stadium after examination. A more detailed description will be given in section 3.

If an attacker gains possession of the activated device, he can synchronize it with any PC and install malware. Passwords, too, are protected unsatisfactorily in PalmOS. As the source code is not open in PalmOS either, there is a possibility of hidden backdoors. Mobile code can be executed by means of Java on the mobile device so that the "mobile code attack scenario", as shown in Figure 2, applies as well. Palm does not support a certificate management system either so that a manipulated certificate would not be recognized.

Direct memory access is supported by PalmOS through the support of ARM and DragonBall processors.

A manipulated program has the possibility to act with all user authorizations. Just as for PocketPC, security software, such as PDA Secure or PDA Defense, is available. But even if these are applied, there are still security risks that do not make it possible to employ PalmOS as a secure operating system for electronic signatures.

PalmOS shows similar security holes as PocketPC 2002. Without additional software there is no possibility to secure the passwords. Further more there is no certificate management. This points out that PalmOS, like PocketPC, is not secure enough for the Pervasive Salesman Scenario. Due to these risks, PalmOS like PocketPC cannot be used as an operating system for a "trusted" pocket signer.

2.3 Symbian

Symbian [Symbian2004] as an operating system provides better protection than PalmOS and PocketPC 2002. The device can be administered in the corporate network by means of an access control list. By using this list, certain contents can be protected from being accessed by other device management servers so that the data can only be synchronized with a certain server.

So far, no major security gaps are known for the operating system. However, with the Nokia Wintesla maintenance program [UCable2003], far-reaching interventions in the mobile device are possible, even when it is blocked. The attacker obtains full access to all setup options of the device, can unblock it with the knowledge gained and has full access to the stored data. Any security claims for Nokia devices are thus reduced to absurdity.

Mobile code and thus fake dialogs are possible due to the support of Java. In contrast to PalmOS and PocketPC, certificate management is installed as a protection against forged certificates. But the user cannot check the security status of the device and has hardly any possibility to install additional security software on the device.

Symbian devices are thus not suited to generate qualified signatures, as there are security gaps, such as a lack of process separation, and especially since there are tools that are able to bypass any security instruments in Symbian. Without the problem of the Wintesla Tool, Symbian provides more security features than PalmOS or PocketPC. With the features of the access control list and the support of certificate management Symbian supports the scenario of the Pervasive Salesman. But even with the better protection of the saved data, Symbian is by far not secure enough for the "trusted pocket signers" scenario.

2.4 Linux

The Linux operating system provides the user with the largest number of security functions from the operating systems presented so far. Due to the possibility of determining permissions for processes, data, etc., a better protection of the data against misuse is ensured. However, there are still numerous security gaps, such as the DMA functionality, which has to be deactivated manually, or the possibility of performing synchronizations without authentication. Furthermore, there are viruses and worms, even if not directly for mobile Linux distributions. It is clear, however, that these are endangered as well, and protection, for example, by means of rights distribution and against buffer overflows is not sufficient. In addition, virtually no additional software is available for Linux-operated devices so that an additional protection cannot be installed. Also, there are too few Linux devices, and changing from the existing operating system to Linux is time-consuming and risky.

The SUSE distribution in combination with the IBM server was thus only graded EAL2 by the German Federal Office of Security in Information Technology [BSI2003]. PDAs with a distribution built on the standard kernel therefore cannot generate sufficiently secure signatures that would make them legally equivalent to the hand-written signature.

Linux provides the best security features for mobile devices. But as described above, Linux could not provide a totally secure area for a "trusted pocket signer". This is pointed out by the decision of the BSI [BSI2003]. But with the implementation of many security features, Linux is the most secure conventional operating system and supports the Pervasive Salesman scenario.

3. POSSIBLE SOLUTIONS

The two following suggestions for a solution are in their development stages and can presently only be used to a limited extent. Nonetheless, they will provide a better protection of the PC and/or mobile device in the future. The objective of these approaches is to protect the internal processes by means of a strict distribution of permissions in the lowest layers. Only by a system-wide separation of memory, access and input/output rights for processes and applications can a system be protected against any kind of malicious programs. By not giving malicious programs all user rights as in current systems, the solutions presented seek to minimize the risk of damage. Above all, the user for the first time has the possibility to check if the computer is in a secure state and if he is communicating securely with the kernel. This is not possible in current systems.

3.1 Perseus

Perseus is an open source project at Saarland University [Perseu2004]. It is aimed at developing a small microkernel as a secure platform. In addition, the user interface shows the user securely what status the system is in, without a malicious program being able to manipulate it. Generally, a kernel is responsible for the administration of the device, files, memory and processes and is loaded directly after booting. The Perseus kernel is aimed at protecting security-critical applications by isolating the individual processes from each other. Perseus is based on the approach that a normal operating system runs like an application, and therefore the Perseus kernel lies below the operating system in the layer architecture. Only by being embedded below the operating system, which is still needed, can Perseus permit isolated processes to take place system-wide between the applications. Isolated processes are not possible for applications within the standard operating system, however, but only between the individual "secure applications" and the Perseus operating system.

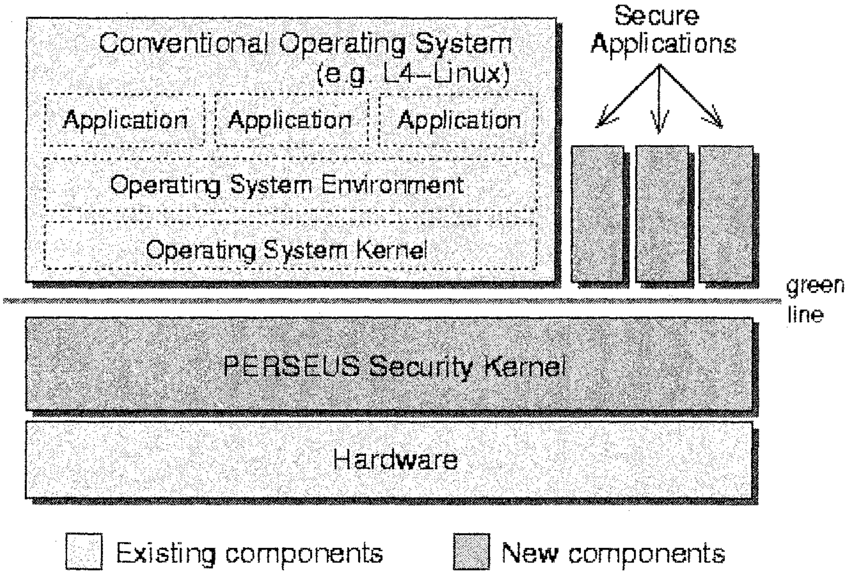


Figure 4: System Architecture Perseus [Perseu2004]

In the Perseus prototype, the trustworthy user interface reserves a line in the upper section of the screen that is permanently under the control of the security kernel.

As the line or LED is under the sole control of Perseus, it cannot be misused by a compromised operating system. If the display indicates that the user is communicating with the Perseus kernel, the control of the display and keyboard solely lies with the security kernel.

3.2 Trusted Platform Module

The "Trusted Platform Module" (TPM) was specified by the "Trusted Computing Group (TCG)", formerly "Trusted Computing Platform Alliance (TCPA)" [TCPA2004].

The TCG hardware consists of two tamper resistant modules called TPM and CRTM (Core Root of Trust for Measurement). Both of them will only be of use if an operating system is used that supports them. Currently there are two operating systems being developed that will support TCG hardware. Microsoft is developing a security technology called NGSCB (Next Generation Secure Computing Base) that will be included in the Longhorn operating system and there are also initiatives to develop a Linux distribution that supports the TCG security modules [MaSmMaWi2003].

The TPM hardware module can be regarded as an extended smart card on which secrets inside and outside of the TPM can be produced and stored [Pearso2002]. These secrets are symmetric and asymmetric keys that are used to ensure the trustworthiness of files, signing of data and the authentication of third parties on the platform. Furthermore, hash values are examined to identify the trustworthy hardware and software components and are stored in data integrity registers. For TPM to be active its hardware must be switched on and the software activated.

For each component (BIOS, OS-Loader and Operating System) a hash value is generated and transmitted to the TPM when the system is started. These values are stored in the "platform configuration register". It is then examined if the currently established hash values are identical with those stored on the TPM. If this is the case, the user can assume that the components and/or the data stored on them have not been manipulated, as otherwise the hash value would have changed and the system or the software would have informed the user. This way an authentication chain is established starting with the CRTM.

The operating system can then build a trusted space (i.e. the "nexus" of NGSCB) for security critical applications in which the applications are separated from each other, and any access from the outside into the "trusted space" is prevented. Uncertified programs, such as a virus or Trojan Horses do not have access to the trusted space.

4. CONCLUSION

The mobile operating systems available today are not suited to produce legally binding electronic signatures. None of the operating systems support secure input/output of the data. In addition, there are still a large number of open security gaps in these operating systems.

The producers of operating systems will have to implement the solutions offered with Perseus and the TCG in future versions or develop comparable solutions. Only then will it be possible to use mobile devices to a larger extent than now, and also employ them in security critical areas.

Until then, the use of mobile devices will continue to be connected with enormous security risks and will require careful consideration. Above all, however, the use of additional software, such as PDA Defense, is highly recommended at the moment, as this eliminates at least a large part of the security risks. The use of a large amount of security software, however, is too demanding for the average user, causes additional costs and is connected with high administrative effort.

REFERENCES

[BSI2003] Bundesamt für Sicherheit in der Informationstechnik (2003): "BSI-DSZ-CC-0216-2003" at: http://www.bsi.bund.de/zertifiz/zert/reporte/0216a.pdf

[EU_esig1999] DIRECTIVE 1999/93/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 13 December 1999 on a Community framework for electronic signatures

[Federr2003] H. Fedderath: Digitale Signatur und Public Key Infrastruktur, http://www-sec.uni-regensburg.de/security/5PKI.pdf

[FoxHor1991] Fox, D.; Horster, P. (1999): "Datenschutz und Datensicherheit" in DuD, Verlag, Braunschweig, p. 194

[MaSmMaWi2003] R. MacDonald, S. Smith, J. Marchesini, O. Wild: Bear: An Open-Source Virtual Secure Coprocessor based on TCPA, http://www.cs.dartmouth.edu/~sws/papers/msmw03.pdf

[MobTra2004] Mobile Electronic Transactions http://www.mobiletransaction.org/index.html

[Palm2004] Palm Website, http://www.palm.com

[PDASe2004] PDASecure – The encryption software, http://www.pdasecure.de/

[PDADe2004] PDA Defens Website, http://www.pdadefense.com/

[Pearso2002] Pearson, S., et al. (2002): "Trusted Computing Platforms - TCPA Technology in context", Prentice Hall PT., New Jersey, p. 5

[Perseu2004] B.Pfitzmann, C. Stüble: PERSEUS: A Quick Open-Source Path to Secure Electronic Signatures, http://www.perseus-os.org/

[Pocket2004] Windows Mobile – based Pocket PCs, http://www.microsoft.com/windowsmobile/products/pocketpc/default.mspx

[Symbian2004] Symbian OS – the mobile operating system, http://www.symbian.com

[TCPA2004] TCPA – Trusted Computing Platform Alliance, http://www.trustedcomputing.org/home

[UCable2003] WinTesla v.5.31 Nokia Service Software for Windows, http://ucables.com/nokia/service/wintesla.html

[WiTness2004] European IST Project „Wireless Trust for Mobile Business“ (WiTness), www.wireless-trust.org