

# PDA OS Security: Application Execution

Jukka Ahonen  
Helsinki University of Technology  
Telecommunication and Multimedia Laboratory  
Jukka.Ahonen@iki.fi

## Abstract

Many people have adopted PDAs into their daily use. Consequently, the interest has grown to do more with these devices than just use them as calendars and phone books. In the mean time, the devices have become more powerful and enabled connectivity to networks.

On one hand, the technical development and users' demand give a great potential for third party developers. On the other hand, however, they also introduce challenges in protecting users' assets. In order to execute third party software safely on a platform, the operating system has to provide permission management based both on the user and the program, as well as enforce memory protection. All the three most common PDA platforms support granting permissions based on user's identity and two of them have memory protection. However, only one of them deals with access rights delegated to programs and even that approach is rather inadequate.

Some security flaws can be addressed by means of third party add-on solutions. Nevertheless, some issues, like the lack of access control based on the application running, have to be solved on operating system level, because they are closely related to the core operating system design. Permission management scheme based on some kind of authorization certificates could be a solution to the problem.

## 1 Introduction

When first introduced, Personal Digital Assistants (PDAs) were above all devices that allowed people to store appointments, contact information and other personal notes in a digital, organized form and keep the information handy whenever needed. During the past few years these gadgets have gained a lot of satisfied users. Nevertheless, the users wanted more. They wanted among other things to take larger documents along with them, play games, use information services and access the endless information in the Internet, just to mention a few. The demand has led to more and more powerful hardware and open operating systems, that allow executing third party applications on them.

On the other hand, the platforms have not only become more powerful but they have also introduced new possibilities with TCP/IP networking, build-in cell phones and expansion slots etc. Connection methods like infrared ports, bluetooth, and internal cell phones have influenced the use of PDAs. The devices are used in an increasingly connected way. This naturally introduces possibilities for new kind of applications. PDAs have became tools to access information whenever, wherever.

Consequently, PDAs are no longer merely digital calendars or phone books. They are shipped with more sophisticated applications, such as word processors, web browsers, music and video players, email clients, games etc. In addition, because third party software can be run on them, the users are not forced to content themselves with the software bundled with the device. There are numerous applications available in the stores as well as downloadable from the Internet.

Nevertheless, as increased resources, connectivity and openness enable a number of new ways to use handheld devices, at the same time they introduce security challenges. In other words, if the we are able to access our information anytime, anywhere, there is a fair change that others try to access it, too.

The goal of this paper is to evaluate the existing PDA platforms from security point of view and come up with a conclusion on their capabilities to restrict illegal access to resources, and on their readiness for safe execution of third party software, in particular.

In order to achieve the goal, I discuss the requirements to execute third party software in a secure way, summarize the three most widespread PDA operating systems in terms of security features and finally propose some alternative approaches as improvements.

The rest of this paper is structured as follows. In second section I discuss what is needed in order to execute third party applications safely and summarize the security requirements. I continue in third section by introducing the three most commonly used PDA platforms and discussing the security models and approaches used. In fourth section I summarize the platforms studied by comparing them to each other. Based on the requirements agreed in second section and the current situation discussed in the following two, I propose some approaches to improve security in fifth section. Finally, in sixth section I make a conclusion on the study.

## **2 Third party application execution**

Open environments make it possible to run third party software on PDA devices. In this section the operating requirements for that will be discussed. The emphasis will be on security requirements. Many of them apply also to closed environments. However, they become more difficult to manage, when there are more parties involved.

User's assets on a PDA device can be roughly divided into data (that can be stored either in the device itself or somewhere else accessible through some kind of network) and operations that can be performed on the data (software). To protect these assets we have to ensure that the data can be accessed and the operations can be initiated only by legitimate parties.

Availability, confidentiality and integrity can be defined as the three basic dimensions of security. Availability means that the system is able to perform the requested tasks and provide results accordingly. Confidentiality ensures that the information is available only to authorized parties. Integrity, on the other hand, means that information cannot be modified by illegitimate parties. [1]

## 2.1 Availability

In the context under discussion availability has a role merely in ensuring that no application is able to prevent other from carrying out their tasks. This could happen for example, if an application could indefinitely reserve resources that are crucial for other applications or even the operating system. The critical resource could be for example processor time or memory. One example of compromising availability would be if application A could access the memory of application B and invalidate B's internal data structures and therefore restrict B from continuing executing.

## 2.2 Confidentiality and integrity

Confidentiality and integrity are the main concern in this discussion. They are discussed in same subsection, because they are closely related. While confidentiality ensures that only legitimate parties can access resources for reading, integrity ensures the same for write access. In order to execute third party applications securely access to the information should be denied from illegitimate users. However the access to the data should also be granted in individually to different software modules. This, of course, prevents malicious programs from harming the system, but more importantly also disables hostile software like Trojan horses from accessing confidential information.

Identification and authorization are security properties whose main purpose is to ensure that operations are initiated by intended parties [1]. In other words, they are needed to achieve confidentiality and integrity. Access control based on user identity is needed. In other words, the user is identified (e.g. password) before granting access to the system. Nevertheless, adequate permission handling is also necessary to allow users to access only data and resources that they have access to. Although PDAs have typically only one main user, she could well use several roles in the system. Those roles would be considered as separate users here.

However, ensuring confidentiality and integrity on user level is not always enough. User cannot easily check what one particular program actually does. So users can unknowingly initiate operations that compromise identification or authorization. In the other words, user level permission handling basically assumes that the user trusts the third party software she installs and executes.

On the other hand, software is nowadays more and more often downloaded from the Internet and it's difficult or in some cases impossible to ensure the source of the applications, not to mention the operation they perform. And even if a user can sure of the authenticity of the source, she still would have to trust that source. In other words, permissions should be handled also based on software modules.

While an application is running, all or part of its data exist in memory. If applications could access each other memory, the confidentiality (read access) or integrity (write access) would be compromised. Therefore applications should be executed in protected memory.

## 2.3 Summary

As a conclusion, in order to protect PDA user's assets the following requirements should be met by the operating system at the minimum:

- Application memory should be protected.
- The user should be identified before authorizing access to the device.
- It should be able to grant access rights to different software modules individually.

## 3 PDA Platforms

In this section the properties of existing PDA operating systems and the approaches used in them are summarized. There are several different PDA operating systems available. However, the three most widespread platforms are so prominent that more detailed analysis is restricted to those three, namely Palm OS, Symbian OS and Windows CE/PocketPC. Each of these is discussed in separate subsection. The emphasis is on security issues and requirements we came up with in section 2 of this paper, in particular access control, data encryption and memory management are discussed.

### 3.1 Palm OS

The Palm OS ideology is to be simple to use and as open as possible. The design doesn't try to replicate features from PCs or other different kind of platforms, but the goal has been to make a handheld that is easy to use and has only those features the user needs. Palm has acknowledged the fact people use PDAs more to access information in short bursts rather than in long sessions like they do with desktop PCs. Palm has a strong support for openness. In fact, there are a lot of third party applications available for the platform. There are also a lot of hardware manufacturers with different kinds of devices. [3]

Palm OS was initially developed for devices with rather restricted resources. The idea was to make the devices cheap enough so that as many people as possible would buy them. Therefore, the operating system had to be designed to cope with limited memory and processing power. As a consequence, some compromises had to be done in the design.

#### 3.1.1 Access control

The latest version of Palm OS (4.0) support automatically locking the handheld. A password is then needed for accessing the device [4]. After entering the password the user is able to access all the data in the device. In other words, Palm has a very simple user identification and authorization scheme based on user identity. However, it supports only one user; there is no username needed to access the device, only password. Identification and authorization protect both confidentiality and integrity to certain extent, because no data can be read or modified if the user hasn't been identified and authorized accordingly.

### **3.1.2 Encryption**

The current OS (4.0) version supports encrypting sensitive data and still keeps it easily accessible [4]. On the other hand, there are several third party products supporting encryption available for earlier operating system versions. Data encryption helps ensuring confidentiality if someone is able to extract the raw data from the device. This feature is especially important, if memory cards or some other kind of external or removable storage is used. In addition, the devices are typically backed up to PC's hard disk during the synchronization process. Encryption protects confidentiality if someone is able access the backup data stored in the PC.

### **3.1.3 File System**

Palm OS doesn't enforce access permissions for files. In fact, it doesn't have a hierarchical file system at all. Application data as well as executables themselves are stored as so called Palm OS databases in the RAM memory. The databases are identified by creator ID and database name. The creator ID normally identifies the creator of the executable that created the database. In practice, however, any application can open databases with any creator. So there is no permission control based on software modules enforced in the operating system.

### **3.1.4 Memory management**

In Palm OS, RAM memory is used both as runtime memory for application execution and as permanent storage for executables and data. In the beginning of application startup, the program is not loaded to other kind of memory, but is executed straight from where it is stored. No virtual memory is used. All applications use the same memory area so memory access is not protected from eg. buffer overflows [7]. In fact, everything is stored in the same memory, with the exception of operating system databases, such as the binaries of standard applications. They are stored in either ROM or Flash memory depending on hardware. Even though that memory cannot be written, it can be accessed from any application. There are e.g. software available that can be used to beam software that is stored in the ROM memory to another Palm device.

## **3.2 Symbian OS**

Symbian OS was previously known as EPOC. The EPOC operating system was initially developed by Psion for their own PDA devices. However, Symbian was founded to take care of further EPOC development. Symbian is owned by Ericsson, Motorola, Nokia, Panasonic and Psion. Recently the name of the operating system has been changed to Symbian OS.

### **3.2.1 Access control**

Access control is implemented by requesting a password before granting access to the device. This ensures confidentiality and integrity on some level, because no data can be read (confidentiality) or written (integrity), without identifying the user and authorizing the access to the device based on that. This is, however, only a limited approach because after identification the user is able to access all the data on device. And even more importantly, all applications launched by the user can access all the data.

### **3.2.2 Cryptography and encryption**

In Symbian OS, encryption and decryption of files is not part of the operating system by default. However, there is a cryptography module available for application developers to allow easy implementation of cryptographic features into their software. Symbian OS Cryptography module consists of symmetric and asymmetric ciphers as well as hash functions and random number generator [5]. However, not all products ship with all the cryptography support because of government restrictions. Symbian OS licensees i.e. device manufacturers can choose which parts of the cryptography module it includes in its products sold in each country. [5]

### **3.2.3 File System**

There are several different file systems used in Symbian devices. Different one is used for files stored in internal flash memory than for those stored on removable memory card, for example. An entity called File Server is built on top of them. It provides a uniform interface for accessing all the files, as well as controls all file access. Nevertheless, there is no permission management based on the application initiating the file access request in the file server.

### **3.2.4 Memory management**

Symbian OS emphasizes the importance of sophisticated memory management on devices with restricted memory resources. Operating system kernel runs in a privileged mode and is responsible for allocating memory to applications running in unprivileged user-mode [5]. Each program can access only memory allocated to it by the kernel. In other words, programs are protected from each other in terms of memory access. In addition, Symbian OS has strict programming guidelines concerning memory allocation. All built-in applications are promised not to fail under any circumstances due to running low on memory and all third party applications are encouraged to do so as well.

## **3.3 Windows CE, PocketPC**

Windows CE is Microsoft's operating system for embedded devices. The ideology behind has been to make it as similar with the other Windowses as possible. The APIs are to large

extent similar to those used in traditional Windows programming. The reason for this is to enable both easy adaptation for application programmers that have experience in Windows on PC platforms and easy porting of existing applications to the embedded platform. The discussion in this paper is restricted to the Windows CE 3.0 version for devices that have pen input, which is the current version. This release is commonly referred to as PocketPC.

### **3.3.1 Access control**

There is an access control mechanism in PocketPC that allows users to lock the devices. In order to access the locked device, a PIN code (kind of a password) has to be entered. This applies both accessing the device through its own user interface as well as connecting it to a PC. In other words, the user is identified before authorizing the access to the assets in device. This protects confidentiality and integrity to certain extent.

### **3.3.2 File System and trusted environments**

Windows CE has a similar kind of hierarchical file system as used in Windowses on PC platform. However, the operating system supports only one user, so there is no use of file access control based on user identity. In addition, the file systems doesn't support granting access rights for each module individually.

Nevertheless, it's possible to create so called trusted environments. In a trusted environment, operating system kernel verifies the signature of the software modules (i.e. applications and dynamic libraries) before loading them. There are three possibilities here: the software module may be trusted without restrictions, trusted with the restriction that no privileged function calls or registry access can be done or not trusted at all. In the last case the kernel refuses to load the module altogether. [8]

The OEM provider, i.e. device manufacturer in practice, is responsible of enabling this feature in its products. It has to provide code for verifying the signature of the module. Therefore, it is up to the manufacturer to choose which kind of signature or certificates are used, if any.

In other words, Windows CE supports granting certain permissions on program level instead of only on user level. However the approach used here is rather coarse grained, because there are only three possibilities: full access, sand-box, and denial.

### **3.3.3 Encryption**

PocketPC doesn't support encrypting and decrypting files on operating system level. Nevertheless, application programmers are provided with means to implement software with cryptography. There are APIs for plain encryption and using security protocols alike.

- CryptoAPI provides application programmers with means to encrypt and decrypt data as well as implement authentication using digital certificates [8].

- Security Support Provider Interface (SSPI) is an API for accessing different security related dynamically loaded libraries (DLLs) in a uniform way. Microsoft has named these DLLs as Security Support Providers (SSPs). In other words, applications can use the services SSPs provide without knowing the details of the security protocols used. [8]
- Windows CE supports the Secure Socket Layer (SSL) versions 2.0 and 3.0 that can be used for secure network connections. [8]

Providing these means doesn't actually improve operating system security itself rather than supports developing security aware applications for the platform more easily and in a uniform way.

### **3.3.4 Memory management**

In windows CE operating system, the memory management is virtual memory-based and system makes full use of a MMU (memory management unit) [6]. In practice this means that application programs are run in protected memory areas. Because each application uses its own virtual addresses they are not able to access each other's memory areas.

### **3.3.5 Summary**

Instead of plain password based access control, Windows CE supports verification of module signatures and restriction of executing privileges based on that. So to certain extent the confidentiality and integrity are ensured also on program level.

## **3.4 Other Approaches**

In this section we analyzed more carefully only the three most widespread platforms. There are of course several other approaches available. Few of them are mentioned here without any deeper discussion.

### **3.4.1 Closed environment**

The most effective way to prevent security faults caused by third party application execution is probably disallowing them completely, in other words, making the operating system closed. How this compromises the usability of the devices greatly, because users have to content themselves with the functionality and applications included in the device. Also upgrading the operating system can be difficult or even impossible.

### **3.4.2 Embedded linux and xBSD**

There are also unix-like operating systems available for PDA devices. However, installation configuration and even use of such devices requires some knowledge on unix-like

operating systems. Most PDA users do not have the knowledge and passion needed to operate them. On the contrary, they want the devices to be as easy to use as possible. In fact, unix-like PDA operating systems are quite rarely used and are therefore not discussed further in this paper.

## **4 Operating system comparison**

The three platforms discussed are compared here in terms of their ability to protect user's assets and their readiness for secure third party application execution, in particular. No new concepts will be introduced here, instead the approach will be a comparison based on the previous section.

### **4.1 Access control**

All evaluated PDA devices can be protected against illegitimate users by requiring password-based identification before letting anyone access the device. PDAs are typically personal devices and support for only one user is in most cases adequate. However, by not supporting several users the operating systems force the user to have only one role in the system. For example it's not possible to log into the device with very limited privileges to ensure that security is not compromised during the session.

### **4.2 File System**

Palm operating system differs from the other two in that it doesn't have a hierarchical file system at all. All the data as well as applications are stored in Palm OS databases. Both PocketPC and Symbian OS, however, organize files in hierarchical manner. Because none of the three operating systems support more than one user, granting access to files based on user is of no interest here.

### **4.3 Encryption**

The current Palm OS version (4.0) supports encryption and decryption of Palm OS databases. The counterpart for this kind of functionality in the other two would be implicit encryption and decryption of files. This is, however, not available in them by default. Nevertheless, both Symbian OS and Windows CE provide third party application developers with cryptography libraries for implementing encryption and decryption on application level. Such libraries are also available for Palm OS.

### **4.4 Memory management**

From security perspective Palm's memory management scheme is the weakest, because applications don't run in protected memory areas. Both the others, however, restrict applications from accessing each other's memory. Either programming failure in an application

or a hostile application may compromise either confidentiality, integrity, or both by accessing other applications' data in memory. As stated in section 2.1, unprotected memory compromises availability as well.

#### **4.5 Access control based on software modules**

Windows CE is the only one of the evaluated operating systems that supports any kind of approach where access rights can be granted for each software module individually. Neither Palm or Symbian OS has any other resource access management system except for the one based on identifying the user. Also, the approach used by Microsoft is quite simplistic and access to individual files cannot be for example granted or denied based on the software module.

### **5 Improvements**

In the previous sections, requirements to run applications safely have been discussed as well as the approaches used in existing PDA systems have been summarized. The biggest shortcomings in the three evaluated operating systems discussed are the following:

- The absence of memory protection in Palm OS.
- The lack of implicit encryption of files in PocketPC and Symbian OS.
- Nonexistent or inadequate approach to allow granting access rights for different software modules individually.

The first one mentioned basically requires a substantial design change in Palm operating system itself and is not discussed more carefully in this paper. However, this section introduces some ideas to improve the situation in the other two cases. The discussion is divided into two subsections: one discussing and introducing an example of an already existent third party add-on encryption solution and the other discussing one possible way to grant access rights on program-by-program basis.

#### **5.1 Third party Security Solutions**

The scope of this document includes mainly the security features implemented to PDA platforms on operating system level. However, some of the security requirements that are not met in operating system design can be addressed using third party software. For example, F-Secure has a product called FileCrypto for Symbian OS. It provides means for strong encryption of any selected information on device [9]. Similar kind of software can solve the same issue on Pocket PC. On the other hand, Palm OS version 4.0 supports similar functionality in the operating system itself, as discussed in section 3.1.2.

However, it is not possible to solve all security problems by adding third party software. For example, memory management is an integral part of the OS in use. Consequently, third

party solution cannot be used to protect the application memory on Palm. Rather, changes in the operating system design itself would be needed. This motivates the importance of operating system design in security issues. If operating system design takes into account the security requirements, third party security solutions are either not necessary or they are easy to implement.

## 5.2 Managing access control based on software modules

Controlling access based on user identity has been well studied and its importance is acknowledged. However, that is not enough to get rid of unauthorized access. The problem is the so called Trojan horses. They are programs that seem to do something useful, but in fact they do something to bypass the access control and so gain access to protected resources. In addition, they may or may not do the task they seem to do. The traditional access control based on user does not help here, because legitimate users execute Trojans with their own access rights. [2]

In her Master's thesis Partanen evaluates how authorization certificates can be used to delegate permissions to Java classes [2]. Similar kind of approach could be used in PDAs on operating system level. Instead of delegating permissions to java classes, we could delegate them to PDA applications or other software modules, such as dynamic libraries. Eventhough this approach is a bit more coarse-grained than with java classes, it would improve the security situation with PDAs. Only programs with valid certificates would be able to access restricted resources and initiate restricted operations. However, the access wouldn't have to be denied completely from untrusted programs. They could be executed in a sandbox-like environment, where they couldn't harm the system in any way. In fact, this approach is similar to the mechanism that already exists on Windows CE, except for the significant difference that access rights to individual resources could be granted for modules instead of just choosing between two security levels.

In a business organization, the IT support department could be responsible for evaluating potential software and granting the certificates for those applications to be approved for use in the PDAs of the employees. The devices would be configured in a such way that without a certificate granted by the company, application could be executed only with very limited rights.

## 6 Conclusions

Users' demand and hardware improvements have led to quite powerful, connected devices with open operating systems. The development has introduced new possibilities to develop interesting and useful third party applications for handheld devices. However, it has also introduced new challenges to manage assets on the devices in a secure way.

In order to allow safe execution of third party software, an operating system should ensure the three basic security properties: availability, confidentiality and integrity. Availability is achieved by making sure that no application is able to prevent the execution of other applications by, for example, reserving too many resources. Both confidentiality and integrity are ensured by adequate access control based on both user and software module as well as by

using encryption.

All the three dominant platforms support access control based on user identity. Palm operating system is the only one that doesn't protect the application memory from other applications. Pocket PC, on the other hand, is the only platform that has some kind of access control management scheme that is based on the application running. However, the approach used is rather coarse-grained.

Third party solution can be used to add some security features to the platforms. However some features cannot be added because they are related to the core design of the operating system. This motivates the importance of operating system design when security is concerned.

The biggest issues seem to be the lack memory protection and the lack or inadequateness of mechanism to grant access rights for different software modules individually on all three platforms. In order address the latter, an authentication certificate based approach could be used to assign resource access permissions to different PDA applications or software modules.

## References

- [1] Pekka Nikander. Modelling of cryptographic protocols - A Concurrency Perspective. Licenciate's Thesis, Helsinki University of Technology, 1997.
- [2] Jonna Partanen. Using SPKI certificates for access control in Java 1.2. Master's Thesis, Helsinki University of Technology, 1998.
- [3] The Philosophy Behind the Palm OS [online] <<http://www.palmos.com/platform/philosophy.html>>
- [4] The Palm OS Platform: A new form of computing. [online] <<http://www.palmos.com/platform/palmos4.html>>
- [5] Symbian Technical Paper: Generic Technology v6.1 [online] <<http://www.symbiandevnet.com/techlib/techcomms/techpapers/papers/v6/over/gt2/index.html>>
- [6] Windows CE For OEMs: Virtual Memory [online] <<http://msdn.microsoft.com/library>>
- [7] Kettula Arto Comparison of Mobile Oses [online] Helsinki University of Technology, 2000. <<http://www.tml.hut.fi/Opinnot/Tik-110.501/2000/papers/kettula.pdf>>
- [8] Maricia Alforque. Creating a Secure Windows CE Device. Microsoft Corporation, October 2000. <<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnce30/html/winsecurity.asp>>
- [9] Handheld Solutions: F-Secure FileCrypto for Symbian OS [online] <<http://www.europe.f-secure.com/wireless/symbian/fcsymbian.shtml>>