# CryptoPalm: A Cryptographic Library for PalmOS

Georgios C. Alexandridis, Artemios G. Voyiatzis, and Dimitrios N. Serpanos

Department of Electrical and Computer Engineering
University of Patras
GR-26504, Patras
Greece
GAlexandridis@myrealbox.com, bogart@ee.upatras.gr, serpanos@ee.upatras.gr

**Abstract.** PDAs and other handheld devices are commonly used for processing private or otherwise secret information. Their increased usage along with their networking capabilities raises security considerations for the protection of the sensitive information they contain and their communications.
We present CryptoPalm, an extensible cryptographic library for the PalmOS. The library integrates a large set of cryptographic algorithms and is compatible with the IEEE P1363 standard. Furthermore, the library offers performance comparable with that of independent, application-centric implementations of the cryptographic algorithms. CryptoPalm is beneficial for PalmOS software developers, since it provides established cryptographic algorithms as an infrastructure for meeting their applications' security requirements.

## 1 Introduction

The PalmOS platform is a well-known and wide-spread PDA platform. It has been used by PalmOne (formerly Palm Inc.), Sony, and Handspring for PDAs of varying capabilities. PalmOS dominated the market of PDAs, reaching a world-wide penetration of 68% by 1999. Nowadays, the PalmOS operating system is incorporated in smartphones i.e., mobile phones with PDA capabilities.

The increasing use of PDAs for processing private information and for securely accessing enterprise information drives the need for support of cryptographic operations in them. Furthermore, PDAs are complete computing systems that can interface with larger systems through communication ports like infrared ports, modems, and wireless network cards. Combined with their "personal" character, PDAs can be an attractive means for two-factor authentication methods.

Given the large installation base of PalmOS-based devices and the numerous applications, it is desirable to have a cryptographic library that incorporates most common operations, like public and secret-key algorithms and cryptographic hashing functions.

In this paper, we introduce CryptoPalm, an extensible library of cryptographic operations for the PalmOS. The library incorporates some unique characteristics. At first, it implements all the aforementioned algorithm families under a common programming interface, allowing a developer to choose the best algorithm for his needs from a set of available ones. At second, it offers compatibility with the IEEE P1363 standard for public-key cryptography, ensuring the correctness and validity of the implementation [1]. At third, it provides an extensible platform, where more algorithms can be incorporated, under the same programming interface, if needed. At fourth, it achieves comparable performance with that of independent implementations of specific algorithms. Finally, the library is compatible with all PalmOS versions, from 3.0 up to 6.0.

The paper is organized as follows. Section 2 provides a short presentation of the PalmOS platform and a review of previous attempts of cryptography implementation on the PalmOS platform. Section 3 provides an analysis of the implementation of the CryptoPalm library and the performance optimizations we incorporated, while Section 4 provides a comparison between the CryptoPalm library and independent implementations of specific algorithms. Finally, Section 5, presents the conclusions and the directions of this work.

## 2 Cryptography in the PalmOS

Palm PDAs are small-factor, battery-operated devices, in the size of a wallet. They offer functionalities such as calendaring, todo lists, and addressbook. The first generation of PDAs are based on the Motorola M68000 processor family, operate at 16 and later 33 MHz and have 512KB–32MB of memory. The word size is 16 bit, stored in big-endian form. They do not have a hard disk or a file system and communicate with a personal computer for data synchronization through a serial port. The second generation of Palm PDAs are based on the ARM 4T processor family, operating in much higher frequencies. The word size is now 32 bit, stored in little-endian form.

Palm PDAs run PalmOS, an operating system with a pre-emptive mutlitasking kernel. The user applications run, until PalmOS 6, as a single task and thus, no more than one user applications can run concurrently. PDAs based on the PalmOS enjoyed high market penetration. The manufacturer provides all necessary programming tools to third parties for developing new applications on top of the operating system. By the end of 2002, more than 20 million PalmOS-based devices have been sold and more than 10,000 third-party applications have been developed.

Among these applications, there have been attempts to implement various cryptographic algorithms for the PalmOS. The cryptographic algorithms are used for encrypting private information stored in the device and for supporting the SSH protocol over TCP/IP connections. The latter was based on pilotSSL, an attempt to port the SSLeay library to the PalmOS [2],[3]. Both SSLeay and pilotSSL have ceased development for quite some years now; the pilotSSL is far from a complete cryptographic library. Daswani and Boneh experimented on the

capability of the Palm devices for supporting secure electronic transactions [4]. They conclude that there are some functions, like RSA key generation, that are not feasible to run on PalmOS and support the view that cryptography in such environments should be based on elliptic curve cryptography. To support this view, they provide comparisons between the pilotSSL library and a commercial SDK for elliptic curve cryptography. Lately, Copera Inc. has introduced AESLib, a high-performance implementation of the AES algorithm for the PalmOS [5].

Previous attempts to implement cryptography on the PalmOS can be characterized as application-centric. The cryptographic algorithms are built as part of larger applications and their maintainance is tight to that of the application. Furthermore, algorithms implemented in an application cannot be reused by another application. It is our view that cryptographic operations are a requirement for the majority of applications handling sensitive, private data on a PalmOS-based device and thus, they should be implemented as a library accessible from all applications. Thus, all applications can benefit from the existence of a correctly-implemented and high-performance library. PalmOS 5 headed for this option by providing a "Cryptography Provider Manager" in the operating system. However, currently it supports only two algorithms, namely RC4 for encryption and SHA-1 for hashing. In the next sections we present CryptoPalm, an attempt to provide a unified cryptographic library supporting all the popular algorithms under a common API for the PalmOS.

## 3 The CryptoPalm Library

The CryptoPalm is a complete cryptographic library for the PalmOS operating system. The CryptoPalm library provides implementations for the following algorithms:

- RSA public key algorithm. The implementation is compatible with the IEEE P1363 standard.
- symmetric key algorithms: DES, and two implementations of AES.
- ECC algorithms: ECDSA, compatible with the IEEE 1363 standard and supporting operations over primary and binary fields ($GF(p)$ and $GF(2^m)$).
- Hashing algorithms: SHA-1 and MD5.

The CryptoPalm cryptographic library is based on the MIRACL big number library [1]. The MIRACL library is a collection of optimized routines for handling multi-precision arithmetic, with emphasis on cryptographic operations.

The selection of the MIRACL library was driven by the fact that it offers a well-tested set of cryptographic operations and all the necessary primitives for implementing new algorithms. Furthermore, MIRACL has the following advantages for the PalmOS platform:

- There are no available general-purpose cryptographic libraries for the PalmOS. The available port of the SSLeay library to the PalmOS has ceased

[1] MIRACL library is available from `http://indigo.ie/~mscott/`.

development for some years now and the port was made for supporting specific operations.
- MIRACL is known to be one of the top high-performance libraries for cryptographic operations [6].
- The library is developed with portability in mind, which was expected to ease the port to the PalmOS platform.
- The library supports both big and little endian architectures, which is a desirable feature, given that the Motorola M68000 processor family follows a big endian architecture.

The development of CryptoPalm library consisted of the following steps: *i.* Porting the MIRACL library to the PalmOS environment, *ii.* Implementation of public key algorithms compatible with the IEEE P1363 standard, *iii.* Integration of symmetric-key and hashing algorithms, *iv.* Implementation of algorithmic optimizations, and *v.* Development of a static and a shared library for use by other programs.

### 3.1 Port of the MIRACL library to PalmOS

The MIRACL library provides the necessary support for big number arithmetic. It has been actively developed since 1988 and currently version 4.85 is available. The library is available in source code and executable forms for various processors and is free for academic and non-profit use. The library is characterized by its compactness, portability, and efficiency. Various processor families are supported and optimizations in the form of assembly code are provided for specific families.

The Motorola M68000 processor family is not currently supported, so the first step was to port the MIRACL library to this processor family. The first step of porting the library to a new environment is to correctly structure the MIRACL Definitions header file `mirdef.h`. The required changes for supporting the PalmOS environment were:

- Define a word size of 16 bits and big-endian storage.
- Disable support for optimized assembly code, since no assembly code is available for the M68000 processor.
- Disable support for standard I/O and file I/O, since the PalmOS platform neither has standardized input/output functions, e.g. ANSI C `printf()` and `scanf()`, nor a file system.

There are some points that must be taken into account when creating a library in the PalmOS environment. A first point is to decide if the library is built as a static or dynamic one. A static library introduces less overhead and the linker, if it supports it, can integrate into the application only the functions of the library that are called. This results to smaller programs. However, if the library code is updated, all programs that are linked with the previous version of the library must be upgraded too, in order to incorporate the necessary updates. Dynamic libraries can be built and installed only once in a Palm device. Programs that use the library have the extra overhead of opening the

library and calling the necessary functions. The advantages of dynamic libraries are the maintainability and the fact that programs can break the 64 KB barrier (dynamic libraries can be 64 KB each).

Another point of attention is the code model, which specifies the type of jumps within the code. There are three approaches to this option: use absolute addresses for calling functions (large model), use only small addressing (small model) and use a mixed model of absolute and relative addresses (smart model). The second approach results in faster code, since it introduces minimum overhead (one jump instruction). However, code jumps must not exceed 32 KB forward or backward (as the jump instruction takes a 16 bit offset value as a parameter). Therefore, a careful link order must be defined for the use of the small model without hitting the jump limit.

The previous work allowed the creation of a static library version of CryptoPalm. The static library comes as a standalone file, `PalmMir.lib`, that contains all the necessary functionality for implementing cryptographic computations. It can be integrated with third-party code. We also developed a shared library version of CryptoPalm, in order to further assist developers of cryptographic software. Since PalmOS shared libraries must not exceed the 64 KB limit, CryptoPalm is composed of four smaller libraries; one for algebraic operations (including the P1363 RSA Implementation), one for elliptic curve operations (including the P1363 ECDSA Implementation), one containing the symmetric ciphers (DES and AES) and the hash functions (SHA-1 and MD5), and one containing Brian Gladman's AES Code. All four PRC files can be transferred to the the PDA during a HotSync operation and can be manipulated like any other Palm OS Shared library.

### 3.2 Public-key algorithms

The implementation of public key algorithms has been standardized in the IEEE P1363 standard [1]. The standard defines all the necessary steps for implementing a public-key algorithm, from the generation and validation of keys, to the encryption, decryption, signing, and signature verification functions. The MIRACL library provides "wrapper" functions for implementing only the public-key cryptography primitives.

We independently implemented a large set of IEEE P1363 functionality, supporting all cryptographic operations for both RSA (IFEP-RSA, IFDP-RSA) and Elliptic Curves (ECSP-DSA, ECVP-DSA). The implementation was validated for conformance with the standard, using the provided test vectors and checking for correct output.

### 3.3 Secret-key and hashing algorithms

The symmetric-key algorithms implemented in the CryptoPalm library are DES and two versions of AES. The MIRACL library contains only an implementation of AES. As to support DES, we ported Eric Young's software DES implementation. This implementation is considered the fastest one in software currently

freely available [6]. We also ported to the CryptoPalm library Brian Gladman's implementation, which is considered the most optimized software implementation of AES [7].

CryptoPalm provides the MD5 and SHA-1 hashing algorithms [8, 9]. We ported to the PalmOS the reference implementation of MD5 provided in [8]. The SHA-1 implementation is contained in the MIRACL library.

Initial performance measurements suggested that the public-key algorithms did not achieve comparable performance with the ones provided in the literature. We further examined the implementations in order to seek for areas of improvements.

### 3.4 RSA Optimizations

Profiling the RSA implementation revealed that the computation of the modular exponentiation operation $x^y \pmod n$ was the dominant one in time, accounting for over 99.5% of the time. Careful inspection of execution traces revealed four areas of algorithmic optimization [10]:

- Usage of the Chinese Remainder Theorem (CRT).
- Improving the exponentiation computation.
- Improving the modular multiplication computation.
- Improving the multiplication operation.

Usage of the CRT theoretically contributes an improvement by a factor of four. A comparison of methods for exponentiation verified that MIRACL is already using the best available method, that is of an adaptive sliding window of 5 bits. The modular multiplication method used in MIRACL, the Montgomery reduction, is almost optimal, since it is 2.5% slower than the table lookup method [10]. We opted for Montgomery reduction, since it requires less space and allows for the next optimization. The multiplication operation can be enhanced by combining the Karatsuba-Ofman multiplication method with the Montgomery reduction for the modular multiplication. The combination of the two methods offers an asymptotic improvement from $O(n^2)$ to $O(n^{1.58})$, where $n$ is the number of binary digits of the modulo.

In summary, CryptoPalm contains an optimized RSA implementation using the Chinese Remainder Theorem, an adaptive sliding window exponentiation of 5 bits and the combination of Montgomery reduction and Karatsuba-Ofman method for the modular multiplication operation. CryptoPalm with this setup achieved an significant improvement by a factor of 5 compared to the default implementation.

### 3.5 Elliptic Curve optimizations

Profiling the Elliptic Curve Cryptography implementation revealed that the computation of the scalar product of a number with a point of a curve was the most time consuming task, accounting for over 98% of the time. The scalar multiplication can be improved in four areas:

– Improving the multiplication algorithm.
– Improving the doubling and addition operation.
– Improving the modular multiplication algorithm.
– Improving the multiplication method.

The analysis of the profiling revealed that the computations could be improved by using the Brickell method [11]. This applies to both primary and binary fields. The improvement is almost halving the required time for the computation. Furthermore, for computations over prime fields, it is possible to use the combined Karatsuba-Ofman multiplication method and the Montgomery reduction method as before, along with the Brickell method. All combined, we achieved an improvement by an order of magnitude for the computations over a prime field.

## 4 Performance analysis

In this section we present performance diagrams for the algorithms supported by CryptoPalm. All performance measurements were taken in the PalmOS Profiler, a special version of the PalmOS Emulator (POSE) [12]. The Profiler provides detailed timing analysis of application execution with high accuracy. For shake of comparison with other works, we provide diagrams of the performance on the Motorola Dragonball 16 MHz processor. We note that we took measurements for other processors too (Dragonball EZ 20 MHz, VZ 20 MHz, VZ 33 MHz), and on a real device (Handspring Visor Pro having a Motoral Dragonball VZ 33 MHz processor). All measurements indicate that cryptographic operations are processing-bounded (performance improves linearly with the speed of processor). Furthermore, performance on the POSE and the real device are identical, further supporting the confidence on the POSE measurements. All measurements presented are the weighted average of 100 experiments.

Table 1 provides a comparison between the CryptoPalm optimized RSA implementation and the one provided by the pilotSSLeay for encryption and decryption. The two implementations provide almost identical performance; for decryption pilotSSLeay is slighty better, for encryption CryptoPalm is slightly better. This is rather encouraging, since pilotSSLeay implements the time-critical functions in assembly code, while CryptoPalm does not.

**Table 1.** RSA (n=512, e=17) performance on DragonBall 16 MHz

| | pilotSSLeay | CryptoPalm |
|---|---|---|
| Decryption | 7028 ms | 7343 ms |
| Encryption | 1376 ms | 1338 ms |

Table 2 provides a comparison between the CryptoPalm optimized ECC implementation and the Certicom Security Builder SDK 2.1 for PalmOS, as reported in [4]. Clearly, the commercial product is achieving better performance

(4–5 times faster). This is an area of improvement for the CryptoPalm library. We should note however that the commercial product contains highly-optimized code in assembly language, while CryptoPalm is written entirely in ANSI C.

**Table 2.** ECC (160/163 bit) performance on DragonBall 16 MHz

| | pilotSSLeay | CryptoPalm |
|---|---|---|
| Key generation | 597 ms | 3465 ms |
| Signature generation | 776 ms | 3684 ms |
| Signature verification | 2448 ms | 10084 ms |

Table 3 provides a comparison between DES and AES of CryptoPalm. The AES algorithm is considered more secure than the DES algorithm; the results indicate that it is also provides higher performance, so there is no practical reason to prefer DES rather AES.

**Table 3.** DES and AES performance (bytes/sec)

| | DES | AES 128 | AES 192 | AES 256 |
|---|---|---|---|---|
| Dragonball 16 MHz | 4343 | 7547 | 6501 | 5710 |
| Dragonball EZ 16 MHz | 4343 | 7547 | 6501 | 5710 |
| Dragonball EZ 20 MHz | 5305 | 9217 | 7940 | 6975 |
| Dragonball VZ 33 MHz | 8705 | 15123 | 13019 | 11429 |

Table 4 provides a comparison between three AES implementations: the one contained in MIRACL and ported to PalmOS, Brian Gladman's ANSI C code ported in the CryptoPalm library, and AESLib, a commercial product using also Gladman's code along with assembly language optimizations. We note that the performance for AESLib is taken by the manufacturer, so actual performance may slightly differ from the one reported. In any case, our port and AESLib offer comparable performance.

**Table 4.** AES 128 encryption performance on DragonBall 16 MHz (bytes/sec)

| | MIRACL | Gladman | AESLib |
|---|---|---|---|
| Encryption | 7733 | 8316 | 10296 |

Table 5 provides a comparison for various input sizes for the performance of the SHA-1 and the MD5 algorithms, as implemented in CryptoPalm. The authors are not aware of other implementations of the two algorithms for the

PalmOS in order to make a comparison. From the table, it is clear that there is a logarithmic relationship between input size and time to complete a hash operation. We also note that for input sizes smaller than 256 bits the performance is the same in both cases, due to padding (the input is padded to 256 bits and then hashed). Furthermore, it is clear that MD5 is about four times faster than SHA-1.

**Table 5.** SHA-1 and MD5 performance (time, in milliseconds)

| Input size (bits) | 160 | 256 | 512 | 1024 | 2048 | 4096 |
|---|---|---|---|---|---|---|
| SHA-1, DragonBall 16 MHz | 9223 | 9210 | 16991 | 24737 | 40229 | 71212 |
| SHA-1, DragonBall VZ 33 MHz | 4605 | 4598 | 8483 | 12350 | 20085 | 28749 |
| MD5, DragonBall 16 MHz | 2843 | 2843 | 4722 | 6468 | 9944 | 16894 |
| MD5, DragonBall VZ 33 MHz | 1423 | 1423 | 2367 | 3233 | 4963 | 8424 |

## 5 Discussion and Future Work

We have presented CryptoPalm, a high-performance cryptographic library for the PalmOS platform. The library is compatible with the IEEE P1363 standard, a unique characteristic for the specific platform that further raises the confidence for the correctness of the implementation. The library achieves high-performance on the platform; the performance of the algorithms is comparable with other works that focus exclusively on a specific algorithm or family of them. Furthermore, CryptoPalm offers a unified API for accessing all algorithms, which is a highly desirable feature and incorporates a set of algorithms for each fundamental operation. Thus, it allows the developer to choose the algorithm that best matches his needs, without requiring to learn and install another library. CryptoPalm remains an extensible platform, where new cryptographic algorithms can be added.

There are some areas of improvement for the library that we plan to work in the future. One direction is the inclusion of more cryptographic algorithms in the library. Another direction is the further optimization of the code, by implementing computational-intensive parts in assembly language, where possible. Finally, it is desirable to further port natively the library on the PalmOS 5 and 6 platform. The library now can run as-is on these platforms through the PACE compatibility layer provided by the newer versions of the operating system. However, this layer introduces some unavoidable overhead that we would like to overcome. Newer versions of the operating system include a limited cryptographic library by incorporating licensed technology from other companies. It would be interesting to compare the performance of CryptoPalm and the licensed libraries in the new operating systems. Furthermore, newer versions of the PalmOS operating system are not supported by the emulator and the pro-

filer, thus it is necessary to develop a new testbed environment and methodology for performance measurements on this platform.

## References


1. IEEE: The IEEE P1363 Working Group. (http://grouper.ieee.org/groups/1363/. Available, April 25, 2005)
2. Goldberg, I.: pilotSSLeay-2.01. (http://www.isaac.cs.berkeley.edu/pilot/. Available, December 20, 2005)
3. Hudson, T., Young, E.: SSLeay. (http://www2.psy.uq.edu.au/~ftp/Crypto/. Available, December 20, 2005)
4. Daswani, N., Boneh, D.: Experimenting with Electronic Commerce on the PalmPilot. In: Proceedings of Financial Cryptography '99. Volume 1648 of Lecture Notes in Computer Science., Springer-Verlag (1999) 1–16
5. Copera Inc.: AESLib for Palm OS. (http://www.copera.com/AESLib/. Available, December 20, 2005)
6. Dai, W.: Speed comparison of popular crypto algorithms. (http://www.eskimo.com/~weidai/benchmarks.html. Available, December 2, 2004)
7. Gladman, B.: AES Implementation. (http://fp.gladman.plus.com/AES/. Available, April 25, 2005)
8. Rivest, R.: The MD-5 Message-Digest algorithm. (IETF RFC 1321)
9. Federal Information Processing Standards Publication 180-2: Secure Hash Standard (SHA). FIPS PUB 180-2 (2002)
10. Koc, C.: High-Speed RSA Implementation. RSA Laboratories (1994)
11. Brickell, E., Gordon, D., McCurley, K., Wilson, D.: Fast exponentiation with precomputation: Algorithms and lower bounds. In: Advances in Cryptology: Eurocrypt '92. Volume 658 of Lecture Notes in Computer Science., Springer-verlag (1992) 200–207
12. Palm Inc.: Palm OS Emulator (POSE). (http://www.palmos.com/dev/tools/emulator/. Available, December 20, 2005)