Secure Elliptic Curve Generation and Key Establishment on a 802.11 WLAN Embedded Device

Panagiotis Papaioannou[‡], Panagiotis Nastou[*], Yannis Stamatiou[†] and Christos Zaroliagis [§]
[*]*Dept of Mathematics*
University of Aegean
Karlovasi, 83200, Samos, Greece
Email: pnastou@aegean.gr
[†]*Computer Technology Institute*
and Dept of Mathematics
University of Ioannina
Ioannina, 45110, Greece
Email: istamat@uoi.gr
[‡]*Dept of Computer Eng & Informatics*
University of Patras
Rio 26500, Patra, Greece
Email: papajohn@ceid.upatras.gr
[§]*Computer Technology Institute*
and Dept of Computer Eng & Informatics
University of Patras
Rio 26500, Patra, Greece
Email: zaro@ceid.upatras.gr

Abstract

Elliptic Curve Cryptography (ECC) is one of the most promising alternatives to conventional public key cryptography, such as RSA and ElGamal, since it employs keys of smaller sizes for the same level of cryptographic strength. Smaller key sizes imply smaller hardware units for performing the arithmetic operations required by cryptographic protocols and, thus, ECC is an ideal candidate for implementation in embedded systems where the major computational resources (speed and storage) are limited. In this paper we present a port, written in ANSI C for maximum portability, of an open source ECC-based cryptographic library (ECC-LIB) to ATMEL's AT76C520 802.11 WLAN Access Point. One of the major features of this port, not found in similar ports, is that it supports Complex Multiplication (CM) for the construction of Elliptic Curves with good security properties. We present some experimental results that demonstrate that the port is efficient and can lead to generic embedded systems with robust ECC-based cryptographic protocols using cryptographically strong ECCs generated with CM. As an application of the ported library, an EC Diffie-Hellman key exchange protocol is developed as an alternative of the 4-way key handshake protocol of the 802.11 protocol.

1. Introduction

As computing and communication devices are equipped with increasingly versatile wireless connection capabilities, the demand for security increases accordingly and, perhaps, more dramatically. However, most of the wireless devices in the market (PDAs, VoIP phones, portable computers etc.) do not have sufficient resources for the execution of computationally expensive security protocols. In view of the resource limitations of wireless devices, Elliptic Curve Cryptography offers an interesting alternative to the classical public key cryptography protocols such as RSA and ElGamal. One of the main advantages of

ECC is that ECC-based protocols use smaller key sizes than traditional cryptosystems for achieving the same security levels. For instance, an ECC system with a key size of 160 bits is roughly equivalent, in terms of security, to an RSA system with a keysize of 1024 bits. Since the key size is significally smaller, so are requirements in space and memory making ECC an excellent candidate for implementation in devices with limited resources.

One of the most important aspects in the design of an ECC-based cryptosystem is the selection of the underlying elliptic curve. In general, using randomly generated curves is not a secure option and, thus, a better alternative is to create elliptic curves with certain security properties embedded in them through a special construction algorithm. There are three main methods of creating secure elliptic curves: the *point counting method*, the method based on the *constructive Weil descent*, and the *Complex Multiplication* method.

In the applications domain up to now, 802.11-based wireless devices use symmetric cryptographic algorithms to secure their connections. According to the 802.11i standard that determines the security properties of 802.11 WLANs, the suggested algorithm is AES for the encryption and decryption of data. The use of a symmetric algorithm requires a common key to be available in both parties in order to communicate. This common key is derived through a mechanism named 4-way handshake, where an exchange of messages provides both parties with the same key for pairwise communication. However, there still exists the need for a passphrase to have already been distributed to both parties via a perfectly secure connection.

In this paper, we present a port of the generic ECC library (ECC-LIB) introduced in [8], [9] to ATMEL's AT76C520 embedded device. The ECC-LIB is a modular library that enables the generation of ECs of certain good characteristics using a variant of the Complex Multiplication. To the best of our knowledge there is no similar ECC port that can generate secure elliptic curves using the Complex Multiplication method in embedded devices. Most implementations, such as [4] and [15], focus on the acceleration of certain EC operations, such as scalar multiplication. Our experimental results demonstrate that under certain conditions the CM method can be used efficiently for constructing secure elliptic curves and implementing secure protocols in embedded networked systems. In addition, based on the port of the CM variant of the ECC-LIB, an alternative approach to the 4-way key handshake protocol is proposed. The proposed protocol creates the shared secret key needed for the symmetric block ciphers of the 802.11 MAC layer using the Diffie-Hellman key exchange protocol based on an EC generated with the CM variant of the ECC-LIB.

2. Key Management in 802.11 WLAN

The IEEE 802.11 standard for wireless LANs (WLANs) is a significant milestone in the evolution of wireless networking technology. Since one of the most important issues in the area of wireless and mobile communications technology is security, the members of the 802.11i Task Group have paid particular attention to provide WLAN users with a powerful security protocol. To this end, in [1] the operation of a *Robust Security Network (RSN)* is defined in an Extended Service Set (ESS) or an Independent Basic Service Set (IBSS), which is the ad-hoc case. The operation of an RSN is based on the establishment of RSN Associations between Stations which can be based on Pre-Shared Key (PSK) or on IEEE 802.1X AKM (Authentication and Key Management). In an ESS, the Access Point (AP) is the Authenticator, and associated devices are the Supplicants. In general, the 802.1X protocol [2] performs authentication in a layer above the IEEE 802.11 MAC layer. The authentication processing has been removed from the IEEE 802.11 MAC, delegating this function to 802.1X. The 802.11 MAC passes all data packets it receives from higher layers, which means that all 802.1X messages are sent as 802.11 data messages to/from Authenticator, delegating the filtering of any unauthorized traffic to 802.1X.

The 802.1X encapsulates the Extensible Authentication Protocol (EAP), which supports multiple authentication methods such as certificates and public key authentication, into 802 frames (EAP over LAN or EAPoL) with a few extensions to handle unique characteristics of 802 LANs. The Authenticator and a Supplicant exchange 802.11 Open System Authentication Request/Response and Association Request/Response. Upon the completion of the association, both the Supplicant and the Authenticator have generated a Pairwise Master Key (PMK) independently. In the presence of an Authentication Server (AS) the generation of the PMK takes place in the supplicant and the AS while the AS trasmits the PMK to the Authenticator through a secure channel. But in the absence of an AS (which is also the case of an ad-hoc wireless network), a secret key into the Authenticator, which is called Pre Shared Key (PSK), is established and it plays the role of the PMK.

Upon completion of an RSN association between Authenticator and Supplicant, the Authenticator initiates the 4-way Handshake protocol ([1]). This is a Key

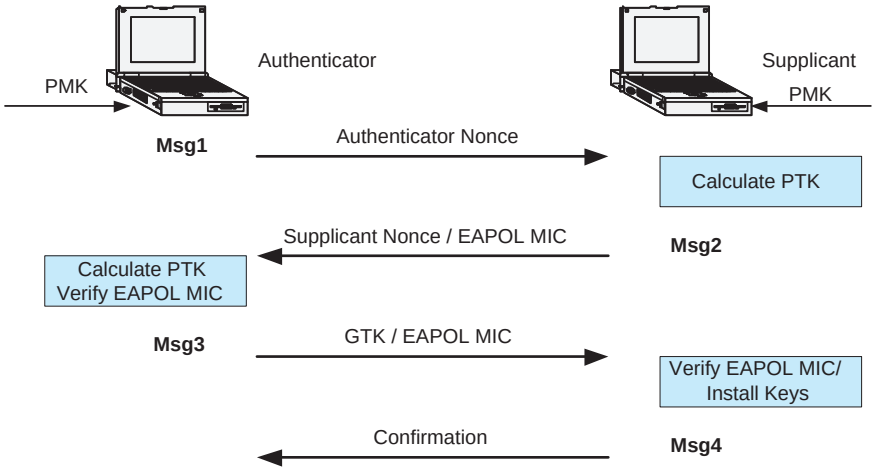

Figure 1. 4-way Handshake Key Management Protocol in 802.11

Management protocol through which the existence of the PMK, as well as that it is current, is confirmed, a unique Pairwise Transient Key (PTK) from the PMK is derived and the unicast encryption and message integrity keys are generated. Those keys are used in the 802.11 MAC Layer symmetric block ciphers (mainly AES) in various modes of operation. In Figure 1 we see the Authenticator and the supplicant exchange four messages as follows: (i) The Authenticator sends a nonce-value to the Supplicant(ANonce), (ii) The Supplicant now has all the attributes to construct the PTK, (iii) The Supplicant sends its own nonce-value (SNonce) to the Authenticator along with a Message Integrity Code (MIC), (iv) The Authenticator generates the PTK and if it is needed the Group Temporal Key (GTK). It sends to Supplicant the GTK and a sequence number along with MIC. The sequence number is the sequence number that will be used in the next multicast or broadcast frame, so that the receiving Supplicant can perform basic replay detection, (vi) The Supplicant sends a confirmation to Authenticator and establishes the PTK and GTK (if it was transmitted by Authenticator).

The Authenticator, whenever a GTK refreshment is necessary, initiates the Group Key Handshake Protocol which consists of two messages where the Supplicant establishes the new GTK created by Authenticator.

3. Elliptic Curves over F_p and the CM method

In this section, we provide a brief introduction to the theory of Elliptic Curves over F_p and to the Complex Multiplication (CM) method. Additional information on elliptic curves and CM method can be found in [3], [5] and [13].

Assuming an odd prime number $p > 3$, the finite field F_p, called a prime field, that consists of the set of integers $\{0, 1, \ldots p-1\}$ and the arithmetic operations addition and multiplication modulo p is defined. An elliptic curve $E(F_p)$ over F_p is the set of points $P = (x, y)$ where $x, y \in F_p$ which satisfy the equation:

$$y^2 = x^3 + ax + b \pmod{p}$$

where $a, b \in F_p$ satisfy $4a^3 + 27b^2 \neq 0$. This set of points along with a point $\mathcal{O}$ called the *point at infinity* and a special addition operation where the point $\mathcal{O}$ plays the role of the identity element define an Abelian Group called the *Elliptic Curve Group*. Based on the addition operation over EC, the multiplication over EC in defined by:

$$Q = k \times P = \underbrace{P + P + \ldots + P}_{k}$$

where $k \in Z$ and Q and P are points of an EC. The order of a Point $P = (x, y)$ is the smallest integer k for which $k \times P = \mathcal{O}$, while the order m of an Elliptic Curve (EC) is the number of points of $E(F_p)$. It holds that $k \leq m$. An $E(F_p)$ is associated with the *curve discriminant* Δ and the j-invariant defined by:

$$\Delta = -16(4a^3 + 27b^2), \text{ with } j = \frac{-1728(4a)^2}{\Delta}$$

Given a j-invariant $j_0 \in F_p$ ($j_0 \neq 0, 1728$), an EC can be easily constructed by setting $a = 3k \ mod \ p$ and $b = 2k \ mod \ p$ where $k = \frac{j_0}{1728 - j_0}$. A second EC, which is called the *twist* of the previous generated EC, is defined by

$$y^2 = x^3 + ac^2x + bc^3$$

where c is any quadratic non-residue in F_p. Upon the completion of the construction of the two ECs, the EC with an order m that satisfies certain conditions ([10], [11], [12], and [14]) ensuring intractability of solving the Discrete Logarithm Problem (DLP) on the EC group is considered *suitable* for a cryptosystem.

The CM method can be used to generate EC of a suitable order m by computing j-variants from which is easy to construct the EC. The method starts with a prime p and then finds the smallest D, which is called the *CM discriminant for the prime* p, along with integer u that satisfies the equations $4p = u^2 + Dv^2, m = p + 1 \pm u$. Then it is checked whether $p + 1 - u$ or $p + 1 + u$ are suitable orders for EC generation. If not, the procedure is repeated, otherwise the next step is based on D to construct Hilbert polynomials ([3] and [5]) and to find their roots. A root which is a j-variant is used for the generation of the EC and its twist. Since only one of the two ECs is suitable, it can be found using Lagrange's theorem by picking points P from each EC at random until a point that satisfies $m \times P \neq \mathcal{O}$ is found. Then the other curve is the right curve.

The CM method is computationally intensive due to the construction of Hilbert polynomials that have huge coefficients and demand high precision floating and complex arithmetic operations. In order to overcome this drawback of the CM method variants of the CM method have been proposed. The variant in [7] uses the construction of Weber polynomial in addition to Hilbert, since Weber polynomials requires less precision leading to much better performance, p is selected at random, u and v are computed using Cornacchia's algorithm [6] and the order m is required to be suitable. A software library, called ECC-LIB based on this variant has been developed and presented in [8] using the framework developed in [9]. This library includes the basic operations in the elliptic curve group, complex and fixed point number arithmetic of high precision and algorithms for the generation of elliptic curves and various algorithms for encryption and digital signature generation. It is a special purpose library that can be used for the development of elliptic curve cryptosystems. It was developed in standard ANSI C for portability reasons and uses the GNU Multiple Precision library (GNUMP) in order to achieve high precision in integer and floating point arithmetic.

4. A port of ECC-LIB to AT76C520 802.11 WLAN AP

In this section, the port of the ECC Library (ECC-LIB) in ATMEL's AT76C520 802.11 WLAN AP is presented. This board is based on an ARM microcontroller, which is an embedded system operating under $\mu CLinux$ Operating System with limited porocessing power and memory resources. In subsection 4.1 the architecture of the AT76C520 device is given while in subsection 4.2 the port of the ECC-LIB to this device is presented along with experimental results.

4.1. The architecture of AT76C520 WLAN AP

The AT76C520 is a device which provides a set of features usually required by routing and gateway applications. The main CPU of the device is a high ARM946 processor running at up to 100MHz. In addition to the main CPU, an ARM7TDMI running at 80MHz is used for the implementation of 802.11a/b/g MAC protocol functions. The AT76C520 can service simultaneously packets exchanged among two Ethernet 10/100T-Base, 80211.a/b/g WLAN networks, interfaces to DSL modems. It can also support standard interfaces like USB, PCI/PCMCIA/Cardbus and UART.

The AT76C520 has internal SRAM of 32-Kbytes which can be used by the processor to avoid transactions with the external memory. The device also supports access to 256Mbytes of external 32-bit SDRAM, 16Mbytes of external SRAM and 16Mbytes of external Flash. The ARM7 processor unit uses 32Kbytes SRAM for both Instruction and Data. The interworking ARM9 processor, which is targeted for routing and bridging functions, uses 8Kbytes Instruction Cache (ICache) and 8Kbytes Data Cache (DCache) while it uses the external SDRAM for instruction or data fetching.

Encryption algorithms and hash functions usually encountered in network applications are implemented in hardware and can be enabled by any of the two ARM processors. Such hardware blocks are the AES unit, which supports the CCM/CTR/CBC modes, the TKIP for WPA support and the IPSec unit with DES/3DES and MD5, SHA-1 capabilities. There is also a WEP unit integrated into the Hardware MAC block that implements the WEP algorithm of IEEE802.11 MAC standard.

The operating system running in AT76C520 embedded device is the μCLinux that stands for Microcontroller Linux. It is a version of the Linux kernel for microcontrollers without Memory Management Unit (MMU). Besides the kernel, μCLinux, which is licensed under the GPL, includes a standard C library (called uClibc) and also contains applications, libraries and tools and supports a root file system. It was first introduced in 1998 and is available for a number of microprocessors, including ARM.

4.2. Porting ECC-LIB to AT76C520

In this section, we present the porting of the Complex Multiplication method variant, as presented in [9] and developed in [7], to the AT76C520 WLAN AP. The Complex Multiplication method generates an EC of a suitable order and also computes the parameters a, b that determine the EC.

The Complex Multiplication method takes as input a discriminant D and the following basic steps are performed:

1) It constructs the Weber/Hilbert polynomials using D.
2) It picks a prime p at random and using the Cornacchia algorithm [6] the equation $4p = u^2 + Dv^2$ is solved finding two integers u and v. If there is no solution a new prime must be chosen.
3) The order of the elliptic curve m shall be either $m = p + 1 - u$ or $m = p + 1 + u$. If one of them is suitable, the method proceeds to step 4 otherwise it proceeds to step 1. An m is suitable if it satisfies ([7]) the following, security related, conditions:
   - $m \neq p$
   - $\forall k$, $1 \leq k \leq 20$, $p^k \not\equiv 1 \ (mod\ m)$.
   - m must have a sufficiently large prime factor (greater than 2^{160}).
4) It computes the roots of the constructed Weber polynomial which are the j-invariants. Based on the roots two elliptic curves are generated.
5) Since only one of the curves has the required order m, we can find the particular one using a simple procedure that is based on Lagrange's theorem where for any EC point, it should hold that $mP = O$. Specifically, points P on each elliptic curve are picked repeatedly, until a point for which $mP \neq O$ is found. Then, the other curve is the suitable curve.

As it is mentioned in Section 3 the ECC-LIB uses the GNU Multiple Precision library (GNUMP) in order to achieve high precision in integer and floating point arithmetic. Since one of the major drawbacks of the embedded systems is the limited memory, it was mandatory to scrutinize the ECC-LIB code so as to port only the modules of ECC-LIB and the GNUMP that are needed for the CM method. The final executable is 250Kb long, which is acceptable for the AT76C520 device. The code was downloaded to the device and a number of experiments were conducted in order to verify the stability and robustness of the code and to measure the performance of the CM method.The experiments consisted of the following: (i) Generation of an elliptic curve using the Complex Multiplication method, and (ii) Generation of a private and a public key pair.

Table 1. Time (in secs) for EC construction with $D = 39$ using Hilbert and Weber polynomials of degree $h = 4$.

	size of p	T_{EC}	T_1	T_2	T_3	T_4
Hilbert	175	122,89	47,73	47,95	5,73	21,46
	192	129,98	47,73	55,10	4,45	22,73
	size of p	T_{EC}	T_1	T_2	T_3	T_4
Weber	175	111,71	29,03	57,06	5,20	20,40
	192	128,13	29,03	71,00	3,86	24,20

Let T_{EC} be the time for the construction of a suitable EC. According to the procedure described in the beginning of this section, we have $T_{EC} = T_1 + T_2 + T_3 + T_4$ where T_1 is the time to construct Weber of Hilbert polynomial, T_2 is the time for finding a prime p and a suitable m, T_3 is the time for computing the roots of the constructed polynomial and the parameters a and b of the ECs with order $p - 1 + u$ and $p + 1 + u$ and T_4 is the time needed to decide which of the two generated EC is the suitable one. The number of primes that were tried in order to find a solution (u, v) using Cornarchia 's algorithm is denoted by $\#p$ while the number of orders m that were examined until to find a suitable one is denoted as $\#m$.

The experiment of the generation of an elliptic curve with CM method using Hilbert and Weber polynomial was performed 1000 times. The obtained timing results are presented in Tables 1, 2 and 3. In each table, for certain values of D and polynomial degree h, we present the total time for the construction of an EC and its constituents T_1, T_2, T_3 and T_4 for various sizes of primes p. In general, the time T_1 for the construction of a Weber polynomial is less than the time needed for the construction of a Hilbert polynomial and as the degree increases the differences in time requirements become larger.

The time T_2 for finding a prime p and a suitable m seems to be greater in the case of Weber polynomials in most of the runs because, as it is presented in Tables 4, 5 and 6, $\#p$ and $\#m$ have greater values in the runs where the construction of a Weber polynomial was used. This is due to the fact that a prime p is selected at random, leading to a nondeterministic behaviour. Moreover, the time T_3 for the computation of roots of Weber polynomial and the parameters a and b of the two ECs with orders $p + 1 - u$ and $p + 1 + u$ is slightly less than T_3 for the computation of

Table 2. Time (in secs) for EC construction with $D = 40$ using Hilbert and Weber polynomials of degree $h = 2$.

	size of p	T_{EC}	T_1	T_2	T_3	T_4
Hilbert	161	68,89	28,38	20,41	0,43	19,66
	175	73,01	28,24	25,49	0,47	18,79
	192	83,75	28,18	29,41	0,50	25,64
	224	119,92	28,21	61,80	0,63	29,27
	size of p	T_{EC}	T_1	T_2	T_3	T_4
Weber	161	67,26	19,39	28,49	0,41	18,95
	175	70,63	19,35	29,93	0,43	20,91
	192	72,30	19,33	30,57	0,62	21,75
	224	98,16	19,36	50,41	0,63	27,75

Table 3. Time (in secs) for EC construction with $D = 88$ using Hilbert and Weber polynomials of degree $h = 2$.

	size of p	T_{EC}	T_1	T_2	T_3	T_4
Hilbert	175	72,31	27,96	20,06	0,46	23,83
	192	77,25	27,89	32,02	0,54	16,78
	224	110,81	27,88	51,05	0,64	31,22
	size of p	T_{EC}	T_1	T_2	T_3	T_4
Weber	175	68,84	21,04	23,97	0,43	23,39
	192	86,23	21,00	34,10	0,67	30,45
	224	102,81	21,00	56,50	0,55	24,75

Table 4. $\#p$ and $\#m$ for $D = 39$ and $h = 4$ and various sizes of p.

size of p	$\#m$		$\#p$	
	Hilbert	Weber	Hilbert	Weber
175	8,84	10,36	74,32	89,46
192	9,56	11,52	77,16	94,08

roots of a Hilbert polynomial. Since the examination of the suitability of the generated ECs is based on picking points from an EC at random, the time T_4 needed to decide which of the two ECs is the suitable one was not always better in the case where Weber polynomial was constructed. Finally, the total time T_{EC} for the EC construction is always better when the Weber polynomial is used.

The second experiment was the generation of a base point and private/public keys. In Table 7 both T_b, the time needed to generate a base point, and T_{pp}, the time for the generation of a pair of private and public keys, increase as the size of prime p and thus the size of the finite field increases.

Table 5. $\#p$ and $\#m$ for $D = 40$ and $h = 2$ and various sizes of p.

size of p	$\#m$		$\#p$	
	Hilbert	Weber	Hilbert	Weber
161	7,25	9,92	28,67	39,96
175	8,01	8,49	30,82	34,18
192	7,92	8,13	32,78	32,8
224	12,01	10,01	50,54	41,2

Table 6. $\#p$ and $\#m$ for $D = 88$ and $h = 2$ and various sizes of p.

size of p	$\#m$		$\#p$	
	Hilbert	Weber	Hilbert	Weber
175	5,96	7,12	22,96	26,96
192	7,92	7,54	31,84	30,44
224	9,42	9,82	36,34	40,24

Table 7. Time (in secs) for a base point generation (T_b) and private and public key construction (T_{pp}).

size of p	T_b	T_{pp}
161	10,41	0,77
175	11,43	0,91
192	13,67	1,03
224	15,59	1,37

5. An EC Diffie-Hellman Key Exchange Protocol in 802.11

The advantages of ECC have led to the deployment of EC cryptography in high end computers and, recently, to resource limited embedded systems. In most of these ECC-based embedded cryptosystems, elliptic curve protocols employed random or standard NIST elliptic curves. To the best of our knowledge there is no generic library that generates secure elliptic curves using the Complex Multiplication method.

In [4], the deployment of a software library that implements the Elliptic Curve Digital Signature Algorithm (*ECDSA*) in an embedded device, which is a wireless Access Point based on an the ARM processor core running at 80MHz. In this implementation the parameters of the used elliptic curves and the base point as well are chosen randomly, which does not guarantee the security of the resulting curve. The Jacobian coordinates are used for the representation of a base point in order to get faster point doubling times. It is a customized library and it does not use any generic library for arithmetic operations. Timing

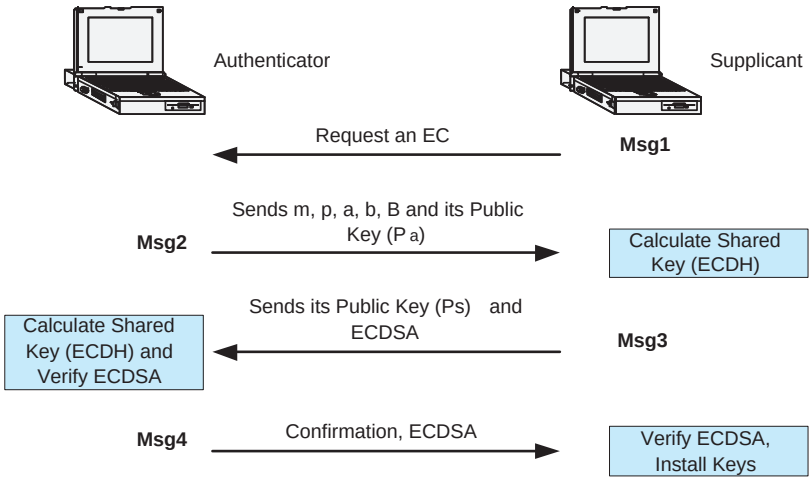

Figure 2. An EC Diffie-Hellman Key Management Protocol in 802.11

results were given only for key size of 160 bit.

An interesting implementation appears in [15] where the generation of a specific NIST approved EC defined over the binary field F_{2^m} of order 233 was tested using an ARM7 microcontroller. The authors also focused on techniques that optimize the *ECDSA* for this specific curve. Since the most time consuming operation is scalar multiplication, this operation was optimized using the Karatsuba multiplication algorithm and a reduction algorithm. A large part of this implementation is written in assembly.

The previously deployed ECC-LIB was applied for the implementation of an EC Diffie-Hellman Key Exchange Protocol that imitates the operation of the 802.11 4-way handshake protocol presented in section 2. The symmetric block ciphers used in 802.11 MAC layer uses the PTK created through the 4-way handshake protocol. The idea is simply to construct a protocol based on Elliptic Curve cryptography that will create PTK in a more secure way.

A simple client-server application was written in standard ANSI C. The server part of the application runs on the AT76C520 device while the client part runs on a laptop. Since the EC generation and the construction of a pair of public and private key using the ECC-LIB requires much time, as it was shown experimentally in section 4.2, a scenario was considered where the authenticator creates an EC with certain characteristics and a pair of private K_a and public P_a keys during its initialization. According to this approach, the PMK is the discriminant D and the size of prime p that the authenticator needs for the EC generation using the CM method.

After a successful association between the authenticator and the supplicant, the latter requests from an 802.11 authenticator an elliptic curve as it is shown in Figure 2. The authenticator sends to the supplicant the EC parameters a, b and m, the prime number p, the coordinates of the base point B and its public key $P_a = K_a \times B$ (Msg2). The supplicant then creates its own set of private and public keys (K_s, P_s) ($P_s = K_s \times B$) based on the received EC and transmits P_s back to the authenticator to complete the Diffie-Hellman protocol (Msg3). The shared key of the two devices is then calculated on the two devices ($PTK_a = K_a \times P_s$ and $PTK_s = K_s \times P_a$). In order to assure both the authenticator and the supplicant that $PTK_s = PTK_a$, the supplicant sends to the authenticator along with the public key, a ECDSA signature using PTK_s. The authenticator verifies the received public key, installs the constructed PTK and sends to the supplicant its public key ECDSA signature using PTK_a (Msg4). Finally, the supplicant verifies that the received signature is the correct one and installs the constructed PTK.

After the completion of the connection, the authenticator can start the generation of either a new EC with new D and, possibly, new size for p or it selects, at random, a new base point. In this way, the authenticator can refresh the shared key used by symmetric block ciphers in 802.11 MAC layer for encryption and decryption. The above protocol was tested using an ethernet connection between a laptop (supplicant) and the AT76C520 embedded device.

6. Conclusions

In this paper we have presented a port of an open source ECC-based cryptographic library to ATMEL's

AT76C520 802.11 WLAN Access Point, showing that it is possible to implement a full ECC-based library that also includes Complex Multiplication to resource limited devices. Our measurements, also, demonstrate that the port is reasonably efficient and compact, given the fact that all the code was written exclusively in ANSI C with no parts optimized in assembly language (something that would limit the portability of the code). As an application of the library's capabilities, an EC-based Diffie-Hellman key exchange protocol was also developed as an alternative to the 4-way key handshake protocol of 802.11, which can be easily adapted to application areas where standardization is not mandatory or, even, desired (Mobile Ad Hoc Networks MANET and Network-Centric Warfare).

References

[1] *IEEE Medium Access Control (MAC) Security Enhancements, IEEE Task Group i P802.11i.*

[2] *IEEE Std 802.1X-2001, Port-Based Network Access Control*.

[3] A.O.L. Atkin and F. Morain. Elliptic curves and primality proving. *Mathematics of Computation*, 61:29–67, 1993.

[4] M. Aydos, T. Yank, and C. K. Koc. An high-speed ecc-based wireless authentication protocol on an arm microprocessor. *Annual Computer Security Applications Conference*, 16:401, 2000.

[5] I. Blake, G. Seroussi, and N. Smart. *Elliptic curves in cryptography*. London Mathematical Society Lecture Note Series 265. Cambridge University Press, 1999.

[6] G. Cornacchia. Su di un metodo per la rizoluzione in numeri interi dell' equazione $\sum_{h=0}^{n} c_h x^{n-h} y^h = p$. *Giornale di Matematiche di Battaglini*, 46:33–90, 1908.

[7] E. Konstantinou. *Theory and Applications of EC based Public-Key Cryptosystems*. PhD thesis, Department of Computer Engineering and Informatics, Patras University, June 2005.

[8] E. Konstantinou, Y. Stamatiou, and C. Zaroliagis. A Software Library for Elliptic Curve Cryptography in Algorithms. In Proc. European Symposium on Algorithms – ESA 2002, Lecture Notes in Computer Science Vol. 2461, Springer-Verlag, 2002, pp. 625–637.

[9] E. Konstantinou, Y.C. Stamatiou, and C. Zaroliagis. On the efficient generation of elliptic curves over prime fields. In *CHES*, pages 333–348. Springer-Verlang, 2002.

[10] A.J. Menezes, T. Okamoto, and S.A. Vanstone. Reducing elliptic curve logarithms to a finite field. *IEEE Transactions on Information Theory*, 39:1639 – 1646, 1993.

[11] G.C. Pohlig and M.E. Hellman. An improved algorithm for computing logarithms over gf(p). *IEEE Transactions on Information Theory*, 24:106–110, 1978.

[12] I.A. Semaev. Evaluation of discrete logarithms on some elliptic curves. *Mathematics of Computation*, 67:353–356, 1998.

[13] J.H. Silverman. *The Arithmetic of Elliptic Curves*. GTM 106. Springer-Verlag, 1986.

[14] N.P. Smart. The discrete logarithm problem on elliptic curves of trace one. *Journal of Cryptography*, 12:193–196, 1999.

[15] E. Turan. ECDSA optimizations on ARM processor for a NIST curve over $GF(2^m)$, June 2001.