Making Password Authenticated Key Exchange Suitable for Resource-Constrained Industrial Control Devices

Björn Haase(✉) and Benoît Labrique

Endress + Hauser Conducta GmbH & Co. KG,
Dieselstr. 24, 70839 Gerlingen, Germany
Bjoern.Haase@conducta.endress.com

Abstract. Connectivity becomes increasingly important also for small embedded systems such as typically found in industrial control installations. More and more use-cases require secure remote user access increasingly incorporating handheld based human machine interfaces, using wireless links such as Bluetooth. Correspondingly secure operator authentication becomes of utmost importance. Unfortunately, often passwords with all their well-known pitfalls remain the only practical mechanism.

We present an assessment of the security requirements for the industrial setting, illustrating that offline attacks on passwords-based authentication protocols should be considered a significant threat. Correspondingly use of a Password Authenticated Key Exchange protocol becomes desirable. We review the significant challenges faced for implementations on resource-constrained devices.

We explore the design space and shown how we succeeded in tailoring a particular variant of the Password Authenticated Connection Establishment (PACE) protocol, such that acceptable user interface responsiveness was reached even for the constrained setting of an ARM Cortex-M0+ based Bluetooth low-energy transceiver running from a power budget of 1.5 mW without notable energy buffers for covering power peak transients.

Keywords: PAKE · ARM Cortex-M0 · Curve25519 · ECDH · PACE · Curve25519 · ECDH key-exchange · Elliptic-curve cryptography · Embedded devices · Elligator · Process industry · Bluetooth · Curve19119 · X19119 · Bluetooth low energy

1 Introduction and Motivation

Connectivity becomes increasingly important also for small microcontroller-based embedded systems found in industrial control installations, such as so-called field devices. More and more use-cases require secure remote user access, e.g. for maintenance and configuration involving subsystems such as industrial control units and home automation electronics. Increasingly smart phones or tablet computers are used as handheld units providing the Human Machine Interface (HMI). The continuously growing Internet of Things will only add to this development.

W. Fischer and N. Homma (Eds.): CHES 2017, LNCS 10529, pp. 346–364, 2017.
DOI: 10.1007/978-3-319-66787-4_17

It is of great interest to provide efficient cryptographic primitives and protocols suitable also for the resource-constrained embedded CPUs typically employed in these environments. In most applications, user authentication by use of iris scanners, fingerprint analysis or based on smart cards is unfortunately not practical. On the other hand, implanted identification chips are already in wide-spread use, but mainly in the context of production animals and for some specific reason rather not for human operators.

In many circumstances, the old-fashioned password remains the only practical means for authentication of human operators. We presume that frequently the crucial weakness of the security solution is the password-based authentication protocol, namely if the protocol exposes the password to offline dictionary attacks. Astonishingly weak challenge-response-type protocols seem still to be in wide-spread use in many critical systems.

In the context of resource-constrained devices and password authentication, the aspect of efficiency becomes of utmost importance, since the computational complexity directly translates into the delay experienced by the user during the login procedure.

This article summarizes the results of research implemented in the context of securing a Bluetooth-low-energy based human-machine-interface for an industrial field device hardware.

Contribution of this work. The contribution of this work is threefold.

- We present the result of our review of PAKE protocols from the perspective of efficient implementations in small microcontrollers.
- We explore the design space for efficient implementations of one option, the PACE protocol family, considering Weierstrass curves as well as more recently suggested Montgomery or Edwards curves.
- We provide experimental results for implementations on an ARM Cortex M0+ for both, 128 and 96 bit security level.

Note regarding side-channel protection
All of the software presented in this paper avoids secret-data dependent branches and is, thus, inherently protected against timing attacks on targets for which instruction duration does not depend on the value of operands.

2 Organization of This Paper

This paper is organized as follows. First in Sect. 3 the reader is given a review of the security requirements for operator authentication in industrial control installations.

In Sect. 4 properties of different PAKE protocols are reviewed from the perspective of suitability for resource-constrained industrial control devices. As a result of this analysis the PACE protocol family suggested by Bender, Kügler and Fischlin [1] was assessed to be particularly well adapted.

In Sect. 5 the specific particularities of the PACE protocol are reviewed and the most important points with respect to efficiency are analyzed.

The subsequent Sect. 6 considers issues that show up when trying to specifically tailor the PACE protocol for the constrained setting. Optimization for efficiency includes selection of a suitable finite field used for the elliptic curve point group, a suitable elliptic curve and a matching choice of symmetric primitives.

In Sect. 7 we introduce a new Montgomery curve "Curve19119" for a legacy security level.

In Sect. 8 we describe the specific tailoring chosen for PACE in the experiments and describe our optimization strategy.

Section 9 combines both, presentation of the experimental results and comparison with other related work.

3 Security Requirements and Implementation Constraints

In industrial control installation it is common to wire many so-called field devices still by using a purely analogue interface, e.g. encoding measurement values in a current ranging from 4…20 mA. This holds even in 2017. Depending on the installation up to 90% of the instrumentation does not use any digital data transmission. This is an advantage regarding security.

Field devices often have to withstand high temperatures and humidity. One consequence is that the user interface often only allows for a few buttons and one line of LCD, not providing good usability for the operators.

This is one motivation for integrating wireless interfaces based on standards such as Bluetooth 4.0 low energy. Wireless access allows for comfort by referring the graphical user interface to a powerful handheld unit. Unfortunately many wireless standards were originally not designed for the security requirements in industry plants, where manipulation of the integrity of one single field device might result in explosions or other severe damage. Note that, for instance, the security layer of Bluetooth low energy 4.0 is not providing any protection against a passive eavesdropper! Providing protection is difficult, since any simple challenge-response protocol exposes the passwords to the risk of offline attacks.

Field devices also often face the requirement of having to be intrinsically safe with respect to the risk of explosions, for instance when used on refineries. Most strategies for intrinsic safety base on circuit-designs that limit peak currents, peak voltages and the amount of energy in buffers like capacitors or batteries. The limitation guarantees that ignition of explosive gas or dust becomes impossible. As a consequence a large portion of so-called "2-wire" field devices has to operate from 15mW to 30mW functional power for its full operation. Note that this power is constantly available but cannot be exceeded transiently since intrinsically safe barriers and interfaces prevent that any more current will be delivered. If circuitry needs transients, e.g. for a measurement circuit, this needs to be buffered locally.

An important aspect in the context of this paper is the limitation of the actual size of the energy buffers, since levels sufficient for triggering an ignition must be prevented. Batteries often may not be integrated due to the continuous maintenance requirement and the temperature rating. A typical value for the energy stored in buffer capacitors for transients is in the range of several mJ only. Note that most of the transient buffer will be allocated for the main functionality of the field device and only a small fraction will be made available for wireless transmission or for complex calculations. The algorithms and protocol implementations need to have both in mind, low power consumption on average and limited energy buffers for covering transients.

4 Review of PAKE Protocols from the Perspective of Resource-Constrained Devices

Since the initial pioneer papers from Bellovin, S.M. and M. Merritt [2] regarding key generation based on weak "user memorable" (i.e. low entropy) passwords extensive literature is available regarding the basic problem. Protocols typically are referred to by use of acronyms such as EKE [8], SPEKE [9], SRP [10], PACE [1], PAK [11, 15], AMP [7, 12] and AugPAKE [13, 16]. Many of these base on the framework of Diffie-Hellman key exchange [14].

At a first glance a plethora of protocol candidates needs to be considered for the setting of field devices. A closer look, however, exposes that unfortunately most of the protocols suggested are to be considered impractical for this setting. For applications in extremely resource-constrained devices the most suitable algorithm subset must not make use of multiplicative group operations, such that using elliptic curves becomes possible.

This special subgroup of PAKE protocols seems to be particularly difficult to construct correctly. Many protocols have been shown to be insecure. We assessed only a small subset of more carefully analyzed protocols such as AugPAKE [13, 16] or PACE [1] to be good candidates. To make matters worse, an additional aspect to consider is the issue of pending patents on protocols or efficient implementation of algorithmic sub-steps. In particular a patent applications for the AugPAKE [13, 16] protocol does apply under US020110145579A1. As a result our further analysis did concentrate on PACE [1].

5 Review of the Password-Authenticated Connection Establishment (PACE) Protocol

The PACE protocol designed by Bender, Kügler and Fischlin [1, 18] might actually rather be considered to form a tailorable family of protocols involving different steps that allow for specific implementation choices. E.g. it may be implemented on large-characteristic fields as well as on groups defined by points on elliptic curves. Correspondingly in the security analysis of the protocol [1] four alternative variants of the Map2Point sub-step of the protocol were suggested. The PACE protocol family assumes that a cyclic finite group of a large order, such as provided by points on an elliptic curve, is available. In the protocol basically four different sub-steps may be distinguished (see also Fig. 2 and 3 in [1] for the specific definitions).

- In a first step a random number "s" is exchanged between the two parties by use of purely symmetric cryptographic primitives and the weak shared secret (password) as key. It is the objective of the subsequent protocol steps to verify that both sides initiate the PACE protocol run by using the same “s” value without exposing any information on its actual value to passive or active attackers.

- In a second step of each protocol run the two parties interactively agree on a session-specific generator of the group: G = Map2Point(s). This involves usage of symmetric and asymmetric primitives and exchange of one or more messages.
- In a third step, the two parties implement a conventional Diffie-Hellman protocol for agreeing on a shared session secret by use of the session-specific generator G.
- Subsequently, in the last step, exchange of conventional authentication code messages being derived from the shared Diffie-Hellman secret are used for mutually proving that both parties share the same secret. The session key is also derived from the Diffie-Hellman result.

Efficiency analysis of PACE. With respect to computational efficiency one may neglect the computational complexity of the symmetric primitives altogether. The two dominant components within the protocol are the Map2Point sub-step and the shared secret generation by the Diffie-Hellman sub-step.

The Diffie-Hellman step in PACE works with a generator base point G that varies in each protocol execution. This implies that optimizations possible for fixed base point algorithms could not be used.

The most important factor for an efficient implementation of the PACE protocol is actually the efficiency of the Map2Point protocol sub-step. In [1] four distinct alternatives have been analyzed.

The DH2Point algorithm as used by the German government authority BSI for the German identiy card requires the equivalent of two fixed and one variable point exponentiation, possibly for patent circumvention. A very similar option is the Coin2Point alternative which is trading off a scalar multiplication against an additional message exchange.

The computational complexity may be significantly reduced if a so-called integrated mapping [27] (in [1] referred to as hash to curve h2c operation) algorithm is available for the selected curve. Such an operation maps an arbitrarily chosen scalar number to a point on the elliptic curve. Fortunately constant-time algorithms for efficient integrated mapping are available for many curves. For examples see [3, 4, 28] and references cited therein, notably the so-called Shallue-Woestijne-Ulas (SWU) algorithm. The possible performance benefit of integrated mapping is large since the order of magnitude of efficient integrated mapping algorithms accounts roughly for two field inversions only and is thus only a small fraction of two exponentiations [28].

6 Tailoring of PACE for Resource-Constrained Devices

Tailoring of PACE for an embedded target should best cover all relevant levels within the implementation pyramid: Choice of a suitable finite field, a suitable finite group, etc. In this section we focus on the asymmetric operations since they dominate the computational effort.

Giving our final result at the very beginning, we conclude that best efficiency regarding all aspects might be obtained by using Montgomery or Edwards curves such as Curve25519 constructed over fields with special primes of the form $2^n - m$ and Elligator2 as part of the Map2Point protocol.

In this paper we aim not only at presenting our final result. We also would like to present the reasoning why other approaches had been discarded in our industry setting. We also aim at giving our rough estimates regarding their respective performance disadvantages. For some applications actual implementations might be forced to use specific algorithms that might not best suited from a performance perspective and also a rough assessment might prove helpful.

6.1 Choosing the Field

Despite some impressing results for binary fields (see e.g. [6]) also on architectures such as the ARM Cortex M0, the more complex security story of constructions on top of binary extension fields [34–36] lead us to focus on prime fields quite early.

Assessment of Performance for random prime fields. We did shortly assess the penalty to expect for random primes (such as used by the Brainpool group [17, 21]) and came to the conclusion that for the 128 bit security level on a small 32 bit CPU like the ARM Cortex M0 roughly a factor of $\sim$ 2.75 should be expected for multiplications and a factor of $\sim$ 4 for squarings. This stems from the observation that the cost for one fully optimized Karatsuba multiplication and half a textbook multiplication for Montgomery reduction makes multiply and square operations almost equally expensive. According to our analysis the possible performance gain for the 1/2 multiplication in Montgomery reduction is almost compensated for by losing the possibility of employing the Karatsuba stages that proved highly beneficial in [5].

Assessment of Performance for the Solinas prime [23] for P-256. Avery rough assessment of the potential of the Solinas prime for P-256 leads us to the expectation that the larger number of additions and conditional moves during the reduction being expensive to implement in constant time accounts for a penalty in the range of some 10% … 30% for multiply and square operations in comparison to the optimized Curve25519 prime field implementation from [5]. We expect penalties for addition and subtraction to be somewhat larger. They might reach even +100% since the nice feature of the additional "carry bit" in the last 32 bit word available for Curve25519 is missing. In our opinion this is one of the factors leading to the speed difference factor of 3 when comparing Curve25519 [5] and P-256 [26].

6.2 Selection of Appropriate Elliptic Curve Groups

Selection of appropriate elliptic curve groups impacts efficiency directly and indirectly by a number of parameters, having both, technical and legal origin, such as pending intellectual property rights. The latter aspect is of major importance for all industrial applications. Specifically industrial control devices typically are designed for world-wide installation and already the complex handling of external licenses or the mere theoretical risk of intellectual property right conflicts in a single country typically force implementers to search for circumvention approaches.

In the context of the PACE protocol family one of the major efficiency parameters is linked to the Map2Point sub-step, specifically the availability of an integrated mapping primitive.

For the NIST standard P-256 with p mod 4 == 3 SWU could be used. Unfortunately part of this algorithm seems to be covered by patents.

This drew our attention to a second set of candidate curves, specifically more recent Edwards or Montgomery curves such as Curve25519 [29]. Curve25519 has recently been standardized by ITEF [22] and independently found to be particularly suitable for the Cortex M0 by a group at the company ARM itself [25]. Moreover a patent-free mapping algorithm, Elligator2, is available [4]. For Curve25519, as a side-effect of the original design goal of avoiding secret-dependent table lookups in [29] highly efficient algorithms are available without facing penalties for variable base point scalar multiplications and with a small memory footprint.

6.3 Tailoring on the Protocol Level

The by far most important parameter on the PACE protocol level for efficiency is the choice of the Map2Point primitive. When choosing Montgomery curves as a basis, Elligator 2 is the natural choice.

When using Coin2Point for patent circumvention the complexity of PACE is roughly doubled. Note that the penalty might even be larger since the requirement of calculating full additions, precludes more memory efficient and possibly faster approaches working on x-coordinate-only point representations.

6.4 Exploring the Potential of Reduced Security Parameters

Use of a legacy-level curve for PACE might very well be appropriate, however we think that going below the 96 bit security level might not be advisable. Also we would only recommend this for a setting where sessions are short and compromised confidentiality due to lost forward security is not critical. Note that this could possibly be considered the case for some industry installations where the integrity is the main target and confidentiality is often considered to form a target of a somewhat reduced priority. We would recommend reducing security parameters only in case that the alternative would be to be thrown back to challenge-response protocols.

7 Curve19119: A Little Brother of Curve25519

Due to the high computational complexity of PAKE protocols we aimed at exploring the performance gain for a legacy security parameter. Unlike the situation for conventional Weierstrass curves there seem to be no established Montgomery curves for a security of say 80 or 96 bits. For this reason we designed a new curve using a field based on the 191 bit prime 2^{191}–19. We refer to the curve and the associated Diffie-Hellman x-coordinate-only protocol by the acronyms Curve19119 and X19119 respectively.

Curve19119 was constructed following almost the same rigid requirement set as used for Curve25519, however for the prime 2^{191}–19. The single exception in the construction prescription is that we imposed the additional constraint on the curve parameter "A" that "A + 2" shall be a square. Note that this allows for application of the most efficient Hisil-Wong-Carter-Dawson [20] point addition formula in extended coordinates for the isomorphic Edwards curve. Just as for Curve25519 the Montgomery curve equation reads

$$y^2 = x^3 + A\,x^2 + x \tag{1}$$

with A = 528418 and the base point x = 11. The candidate A = 922 has been ruled out due to the group order being smaller than $2^{(191-3)}$.

The group order of Curve19119 is 8 (2^{188} + 6805822842506810719592233527) with a secure near-prime order quadratic twist. Note that actually a very similar curve with A = 281742 was suggested by Diego F. Aranha et al. [19]. This other curve, in contrast to the Curve19119 presented here, does not allow for the more efficient point addition in extended coordinates.

8 Putting It Together: PACE on the ARM Cortex M0

In the following paragraphs we will present our specific choices and elaborate on our optimization strategy. The full protocol overview of our implementation is given in Fig. 1 and uses mostly terminology from [1]. Note that the task of generating fresh entropy for random number generation is a crucial aspect but due to its complexity out of the scope of this paper. We implemented the full protocol with Curve25519. For Curve19119 we only aimed at being able to assess the performance gain and restricted the implementation effort to the most expensive component, the X19119 Diffie-Hellman part.

According to our target security level in the protocol, we take 128 bit random numbers for the values *s* and *t* and chose a 64 bit nonce n for Salsa20-20.

Review of the ARM Cortex M0 microcontroller architecture. The ARM Cortex M0 and M0+ cores (M0) are the smallest members of ARM's Cortex-M series targeting low-cost and low-power embedded devices. The important feature with respect to asymmetric cryptography operation is the 32 bit x 32 bit => 32 bit single cycle multiplier engine available in virtually all actual instances. Previous research [5] has shown that one of the key bottlenecks for efficiency is register pressure in conjunction with a comparably slow memory interface (von Neumann-Architecture with a shared address and data bus).

8.1 Symmetric Encryption

For both, random number generation and symmetric encryption we made use of Salsa20-20 [32]. We selected the conservative 20 rounds variant because the amount of payload to encrypt is small and efficiency considerations allow for the more

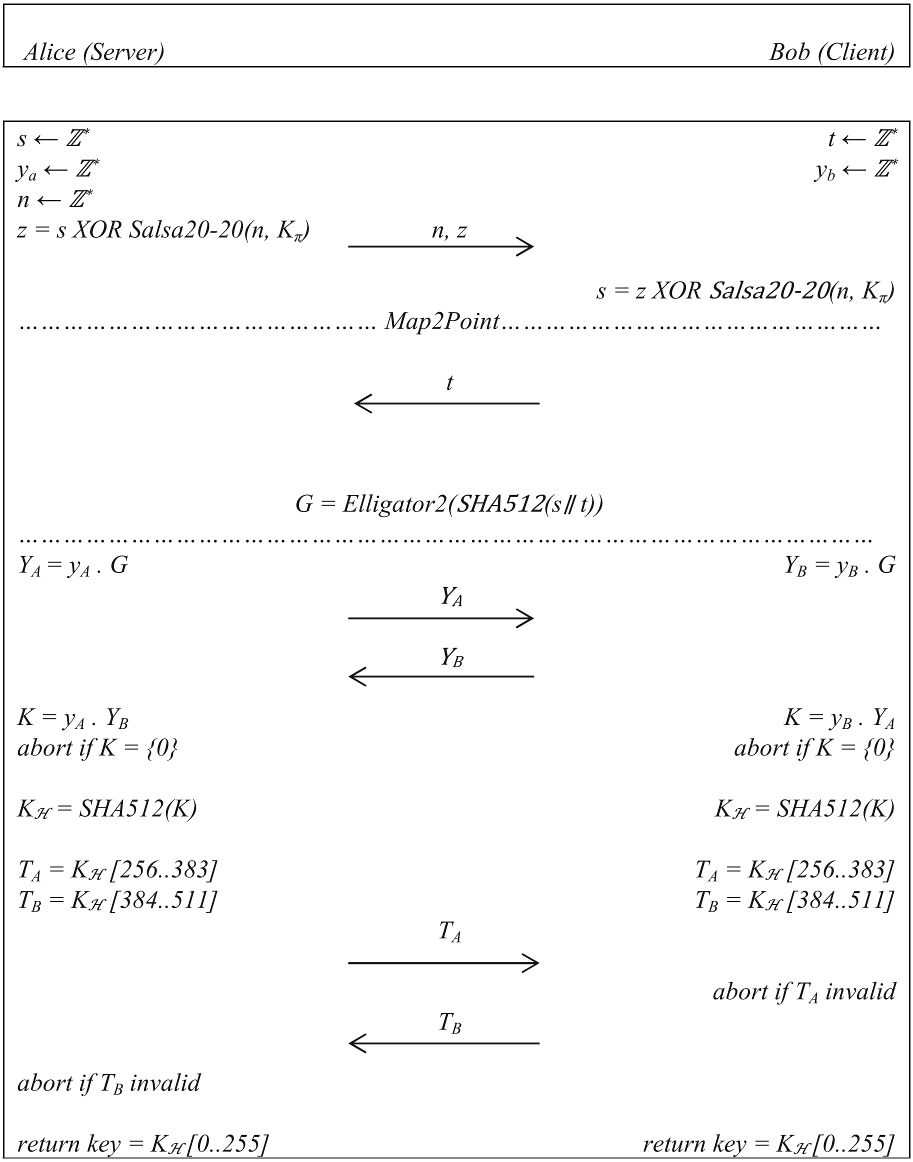
Alice (Server)
Bob (Client)
$s \leftarrow \mathbb{Z}^*$
$t \leftarrow \mathbb{Z}^*$
$y_a \leftarrow \mathbb{Z}^*$
$y_b \leftarrow \mathbb{Z}^*$
$n \leftarrow \mathbb{Z}^*$
$z = s$ XOR Salsa20-20(n, K_π)
n, z
$s = z$ XOR Salsa20-20(n, K_π)
Map2Point
t
G = Elligator2(SHA512($s \| t$))
$Y_A = y_A \,.\, G$
$Y_B = y_B \,.\, G$
Y_A
Y_B
$K = y_A \,.\, Y_B$
$K = y_B \,.\, Y_A$
abort if $K = \{0\}$
abort if $K = \{0\}$
$K_{\mathcal{H}}$ = SHA512(K)
$K_{\mathcal{H}}$ = SHA512(K)
$T_A = K_{\mathcal{H}}[256..383]$
$T_B = K_{\mathcal{H}}[384..511]$
$T_A = K_{\mathcal{H}}[256..383]$
$T_B = K_{\mathcal{H}}[384..511]$
T_A
abort if T_A invalid
T_B
abort if T_B invalid
return key = $K_{\mathcal{H}}[0..255]$
return key = $K_{\mathcal{H}}[0..255]$

Fig. 1. Overview over our PACE protocol choices

conservative variant. We preferred Salsa20-20 because according to our assessment it was more extensively reviewed, specifically as part of the eSTREAM project than other candidates, such as Speck or Simon [33]. This choice also avoided considerations regarding cache timing attacks on the smart phone implementation for the client.

Assembly optimization strategy. The most important identified weakness of the M0 is the memory bandwidth. Luckily, the Salsa20 permutation runs in its inner loop on a set of four 32 bit words being permuted one after the other and, thus, fits into the register set. Also the 32 bit rotation operation requires only one single instruction on the M0 architecture. Our assembly code implementing the hsalsa20 permutation of the 64 byte input runs in 2628 cycles ($\sim$ 41.1 cycles/byte) and needs 404 bytes of program flash.

8.2 Cryptographic Hash

We selected SHA512. An advantage provided by using SHA512 was also that when hashing the shared secret of the Diffie-Hellman substep, the 512 bit result allowed for extracting all of, 256 bit session key and two 128 bit authentification verification messages from a single run. This avoided the need for implementing a specific symmetric message authentication code primitive.

Assembly optimization strategy. The main bottleneck for the SHA512 implementation is the fact that the small 32 bit register set of the M0 is by far too small for holding the intermediate state of the SHA512 subsystem. Also the 64 bit rotations require a large number of 32 bit shifts. The implementation is optimized for speed rather than code size by unrolling the inner loop completely. This results in a rather big memory footprint of 1448 Bytes. Hashing 10 bytes costs 22031 cycles on the M0 and for 2048 bytes of data we end up with an efficiency of 181.6 cycles per byte.

8.3 Point Verification for PACE

Both, Curve25519 and Curve19119 are of nearly prime order. Specifically, the group order is 8 times a prime. During the X25519 and X19119 scalar multiplication each of the weak points is mapped onto the neutral element and thus verification that a point is valid may be implemented subsequently by checking, after each point multiplication, that the result of the point multiplication differs from the neutral element. Insertion of a twist point does not provide the attacker any advantage according the security guarantees of X25519 and the identically constructed X19119.

8.4 Map2Point Protocol Substep

We use the Hash2Point procedure and Elligator2 as “h2c” function according to terminology from [1].

Elligator2. The implementation shares most arithmetic operations with the elliptic curve point multiplication. The most costly sub-steps are formed by one field inversion and one exponentiation with $2^{254} - 10$. Both of them are implemented in constant time by an exponentiation operation. For the exponentiation with $2^{254} - 10$ an important optimization goal was to reduce the number of temporary field elements. We came up with a solution needing 255 field squarings and 11 field multiplications.

8.5 Diffie-Hellman Protocol

Regarding Diffie-Hellman protocol, we implemented two variants, X25519 and X19119 in order to assess the potential of performance gain when reducing the security parameter from 128 bits to 96 bits.

Optimization for X25519. The optimized arithmetic on the prime field and the basic algorithm used for the X25519 protocol sub-steps is based on the strategy of [5]. We integrated only almost negligible improvements regarding the integer squaring operation. Differing from [5] we implement the conditional swap operation after each ladder step by swapping pointer variables instead of data. We expect slightly better performance and also a reduced side-channel leakage [24].

Optimization for X19119. In order to allow for a fair comparison between the legacy-level security curve Curve19119 and the highly optimized X25519 implementation from [5], an equivalent level of optimization was considered necessary. We followed the same approach that has been used as in [5]. Due to the almost identical structure of the curve construction, reduction, addition and subtraction algorithms could be re-used almost identically just as for the algorithms for the x-coordinate-only Montgomery ladder of X19119.

With respect to the optimization of Multiplication and Squaring, Curve19119 suffers from the penalty of less symmetry. For the prime 2^{255}–19 a cascade of three refined Karatsuba stages could be used for mapping 256 bit multiplications to the 32 × 32 bit level in a symmetric cascade. This is possible because 256 is a power of two. For the prime 2^{191}–19 our most efficient multiplication and squaring implementation first uses a refined Karatsuba stage for mapping 192 bit operations to 96 bit operations.

On the 96 bit level, squaring was split into a 64 bit squaring using one additional level of refined Karatsuba, two 32 bit multiplications and one 32 bit squaring. The 96 bit Multiplication correspondingly was mapped onto a 64 bit multiplication (implemented again with one level of refined Karatsuba) and 5 remaining 32 bit multiplications.

The lack of symmetry in comparison to the 256 bit case results in a slightly increased overhead due to additional memory accesses. This was only partially compensated for by reduced register pressure.

It is also worth noting that the Montgomery curve group constant “A” for Curve19119 is slightly less optimal than for Curve25519. For Curve25519 the small curve constant required within the Montgomery ladder calculations fits into 17 bits and multiplication could be implemented by using one addition and one 16 bit multiplication, while for Curve19119 18 bits length make two 16 bit multiplications necessary for each operand word.

8.6 Implementation Strategy Regarding Absence of Energy Buffers: Asynchronous Crypto Engine (ACE)

Due to the absence of large energy buffers in many field devices it turned out to be mandatory to setup a framework that allows for an “interrupt and resume” mode of operation in case of temporarily insufficient power resources.

For this reason all of the calculations of the protocol were implemented by an asynchronous crypto engine (ACE) object accounting for roughly ¾ of the PACE implementation effort. The ACE interfaces to the host application by accepting calculation tasks for complex operations and by generating an asynchronous “requested operation completed” event subsequently. The engine is periodically invoked from a power supervision system in case that the respective energy buffer charge level allows for a given number of CPU cycles to be allocated for cryptographic calculations. Details on the asynchronous calculation tasks will be given in the results section.

Note that the ACE object strategy also optimizes for stack requirements since the point within the source code where the actual calculation is triggered may be specifically chosen such that the call stack has only a low fill level.

9 Experimental Results and Discussion

In this section we will present experimental results of our implementations. We first give details regarding the hardware used for the experiments. Then we use a bottom-up approach for structuring the presentation of the results.

9.1 Environment Used for Collecting Experimental Data

The results reported here were measured on an nRF51822 microcontroller with integrated wireless transceiver from the company Nordic Semiconductors. This device includes a 32 bit ARM Cortex M0+ microcontroller, 256 kByte of flash memory and 16 kByte of RAM in addition to radio frequency circuitry suitable for the Bluetooth low energy protocol. Data flash and program memory access do not require wait states on this target platform. It does not include cache and, thus, RAM access timing does not depend on the actual address.

Specific properties of the target hardware platform. In the nRF51822 around 128/256 kByte of flash memory and 10/16 kByte of RAM memory are required alone for running the wireless protocol stack. Around 6 kByte of RAM are available for both, the communication application and security operations, both for static data and execution stack. In our setting first a Bluetooth connection is established, the PACE protocol messages are exchanged with a smart phone and the authentication calculations have to run while the wireless link is maintained. I.e. the client-side implementation of the protocol runs on a smart phone.

A particular property of the nRF51822 setting in Bluetooth operation is that the 16 MHz clock frequency cannot be divided down in small steps.

For this reason the CPU core is either “on” or “off” and consumes roughly 4.3 mA when running. Due to required operation in the industrial temperature range up to 85°C a supply voltage of 2 V is used, being larger than the necessary value for consumer temperatures. The value was obtained by current measurements for repeated X25519 protocol runs and takes into account some safety margin for process variations and current consumption increase at 85°C temperature.

In the given setting the M0 transceiver CPU interfaces to a main microcontroller of the control unit by use of a communication software layer using a serial interface. The main microcontroller of the control unit monitors the energy buffer state and allocates a certain amount of the available energy buffer budget to the Cortex M0 running the wireless interface and the security implementation.

Since the wireless interface for the HMI use-case is considered only to be an "add-on" feature most of the net functional power of 30 mW is allocated for the main field-device functionality. Only an amount of power of 1.5 mW is available for the M0 transceiver unit, a significant fraction of which being consumed by the RF receiver and transmitter unit and the Bluetooth protocol stack.

Cycle counts reported here were experimentally obtained by use of a hardware timer block in the nRF51822 controller. The cryptographic part of the software was compiled by use of the LLVM compiler with the settings recommended in [5].

9.2 Efficiency Results for Asymmetric Cryptography

Table 1 summarizes our results regarding synchronous execution of the asymmetric primitives for squaring, multiplication, addition and inversion. It also benchmarks the performance of X25519 and Curve25519 field operations against X19119 and Curve19119. The columns "*a24" and "*i16" refer to multiplication with the curve constant and a 16 bit integer respectively. The latter operation is implemented for a continuous re-randomization in projective coordinates as defense e.g. against horizontal attacks [30, 31] being out of the scope of the present paper and not activated for the X25519/X19119 speed measurements reported here.

Table 1. Cortex M0 cycle counts for prime field operations and a single point multiplication. All field operations include reduction modulo $2^{191} - 38$ and $2^{256} - 38$ respectively.

	x*x	x*y	x + y	1/x	* a24	* i16	½ ECDH
Curve19119	666	983	95	140568	144	119	1801856
Curve25519	985	1475	117	268281	190	145	3466086
relative factor	1.48	1.50	1.23	1.91	1.32	1.22	1.92

The column denoted "1/2 ECDH" refers to one run of the X25519/X19119 protocol respectively. The cycle count for X25519 is 50480 cycles lower in comparison to [5]. We expect that this is mainly due to the fact we made use of constant-time swaps of pointers by logic operations instead of swapping the full field elements as in [5].

The only other published report regarding efficiency for prime-field curves for the M0 that we are aware of is found in [26]. There 4.59 (10.73) million cycles were reported for an assembly-optimized NIST P-192 (P-256) scalar multiplication respectively. Our implementations for the M0 on the 96 bit and 128 bit security level are a factor of 2.55 and 3.1 faster respectively. We therefore assume that our implementation establishes a new speed record for an implementation for the 96 bit security level, with timing side-channel awareness. In our opinion the large difference stems mainly from the fact that the multiplication and squaring operation is much better optimized. A second important factor might be that the addition formulas for the Montgomery curve provide better performance than what is possible with P-192.

The cost of a synchronous execution of the Elligator2 algorithm costs 547338 cycles for Curve25519. We did not implement it for Curve19119, The operations required for Curve19119 are essentially the same as for Curve25519. The complexity is almost identical to two field inversions since the cost is dominated by the field squarings required by the exponentiation approach used for the constant-time algorithm. Therefore an improvement factor of equally 1.9 may be accurately predicted.

When comparing the efficiencies of X25519 and X19119, we come to the conclusion that roughly a speed improvement of 1.92 is possible for the legacy-level curve also on the PACE level.

9.3 Efficiency Figures Regarding the Asynchronous ACE Engine

The results for the most important performance figures regarding the asynchronized protocol engine are summarized in Table 2. The time values were calculated for the 16 MHz clock of our core and the energy values were calculated by using factors for the supply voltage of 2 V and the drawn current of 4.3 mA.

Table 2. Results for the asynchronized ACE protocol engine

Suboperation	Cycle count	Time	Energy/µJ
X25519 ladder step	13,486	843 µs	7.2
Elligator step “v”	271,061	16.9 ms	145.7
Elligator step “epsilon”	276,291	17.3 ms	148.5
Field inversion	268,289	16.8 ms	144.2
Short SHA512 block hash	21,560	1121 µs	11.6
Prepare random scalar for X25519	17,945	1121 µs	9.6
Complete PACE protocol run	7,588,000	474 ms	4078.6

The respective state of the PACE protocol is stored in the body of the Asynchronous Crypto Engine (ACE) object. This object also holds all intermediate results required for resuming an interrupted calculation. For storing this state and intermediates we need 264 bytes of static memory and measured an additional execution stack requirement of 432 bytes by using a stack guard pattern method. The sum of both is only slightly larger than the 548 bytes reported in [5], probably due to the inclusion of the SHA512 operations.

The total protocol including two point multiplications and the Elligator accounts for 7.588 million cycles and a dissipated energy of roughly 4 mJ respectively.

Note that this amount of energy is about four times the total energy buffer size of our explosion protected experiment hardware! Since the CPU cannot be clocked down (not unusual for wireless transceivers!) we had to go to sleep very frequently so that the energy buffer may re-charge with the average current granted to the M0 subsystem until the calculation operation may be allowed to resume. Recall also that we run the calculation while the wireless stack maintains the link to the handheld unit running the client side implementation!

The interruption of the calculation was not allowed at arbitrary points within the software but only at distinct boundaries. The biggest blocks are formed by the field inversion and the Elligator because otherwise we would have had to place all of the temporaries used during the field exponentiation in the ACE object reducing the amount of memory available. In order to limit the amount of the maximum energy chunk, the Elligator was split into two sub-blocks coined “v” and “epsilon” in line with the terminology of [4].

In total the maximum transient energy chunk is given by the inversions and the two Elligator2 sub-steps and amounts roughly to 150 µJ. We assessed the additional overhead due to the asynchronous interface to amount to roughly 3%.

9.4 Assessment of the User-Perceived Login Delay

While the biggest amount of required energy buffer size is determined by the maximum chunk size in the ACE state machine, the user-experienced duration of the login delay is controlled mainly by the granted average power. If the core would be allowed to run without interruption, the whole protocol calculation would have finished after 474 ms. This is clearly perfect from a usability perspective on the GUI interface.

However, when assuming that 5% of the total average power of 30 mW of the control unit may be allocated for the “add-on-feature” of a wireless user interface, we end up with roughly 1.5 mW. Subtracting 0.5 mW for maintaining the wireless link one ends up with 1 mW average power for the security functionality or 0.5 mA at 2 V. If the core consumes 4.3 mA it must sleep most of the time. For this reason a time stretching factor of 4.3 mA/0.5 mA of 8.6 needs to be considered, such that the actual protocol calculation needs roughly four seconds.

This is a value clearly perceivable by the user but still in an acceptable range. The times calculated theoretically such as above also roughly correspond to the times measured on the actual smart phone setting. We sometimes observe additional delay of say 0.5 s, since sometimes the ACE object needs to wait a bit until the amount of energy required for the Elligator is fully available. Wakeup-cycles of the CPU are triggered by the radio transmission periods controlled by the handheld device.

When considering a login delay of four seconds, it is obvious that there is not any room left for loosing efficiency by avoiding the burden of assembly optimizations or by choosing cryptographic primitives of lower efficiency!

The tight constraints also were the original motivation for developing the custom curve Curve19119 in the first line. It was mainly due to the performance improvements obtained in [5] that acceptable login delays were obtained without being forced to use a security parameter that is no longer state-of the art.

10 Summary

Summing up, in this paper we have explored the design-space regarding password authenticated key exchange for the domain of resource-constrained explosion protected industrial control devices.

We first reviewed PAKE protocols from the literature and came to the conclusion that the PACE protocol family is well suited for the given setting.

We then analyzed the impact of the choice of elliptic curve candidates for PACE for ARM Cortex M0 devices and came to the conclusion that best efficiency for software-based implementations on small 32 bit microcontrollers is likely to be obtained when using Montgomery or Edwards curves over prime fields using Pseudo-Mersenne primes of the form $2^n - m$.

Based on our analysis an implementation avoiding the risk of intellectual property violation without complicated licensing for the PACE protocol on NIST curves should be expected to be roughly a factor of 2.5 less efficient. The main factor is patent circumvention regarding hashing of scalars onto elliptic curves and use of the Coin2Point primitive of PACE as workaround. For curves built on top of a random prime field we derived an additional speed-reduction factor of roughly 3 accounting for the large cost of modulo reduction in a purely software-based solution on the ARM Cortex M0.

We did evaluate the performance benefit stemming from reduction of the 128 bit security parameter to roughly 96 bits and observed a speed gain in the range of 1.92. For this purpose we introduced a new elliptic curve Curve19119 and a corresponding Diffie-Hellman Protocol X19119 that we believe to setup new speed records for the 96 bit security level for constant-time implementations on Cortex M0 microcontrollers.

We used our optimized elliptic curve cryptographic algorithms in order to construct a tailored solution based on the PACE protocol family.

Finally we have shown that the scheme allows for an actual implementation in the setting of a wireless Bluetooth transceiver controller running the PACE protocol with Curve25519 in parallel to the wireless operation. For a power budget of 1.5 mW worst case login delays in the range of 4 s were attained for the 128 bit security level.

This result was obtained in an explosion-protected setting where incorporation of larger capacitors or batteries was impossible. The clue to circumvent the problem generated by the absence of notable energy buffers was definition of an asynchronous operation mode for the cryptographic algorithms. This way the amount of required energy buffer size was reduced down to 150 µJ.

Our analysis brings us to the conclusion that when working on the conventional Weierstrass curves, the limits imposed by acceptable login-delays on the user interface would most likely have forced us to reduce the security parameter to a value that might not be adequate nowadays. Ultimately this might even have driven us back to weak challenge-response protocols.

Acknowledgements. The authors acknowledge inspiring discussions with Peter Schwabe, Marc Fischlin, Florian Bachmann and Johann Heyszl. We also would like to thank Tanja Lange for drawing our attention to the possibility of advantageous application of reptiles.

Concluding remark

Concluding this paper regarding a rather complex protocol we express the hope that our contribution also might help to re-explore the potential of special types of simple challenge-response protocols. We do refer specifically to protocols that might be constructed on top of isomorphic transformations from the space of cryptographic protocols to Italian language, such as sketched in Fig. 2.

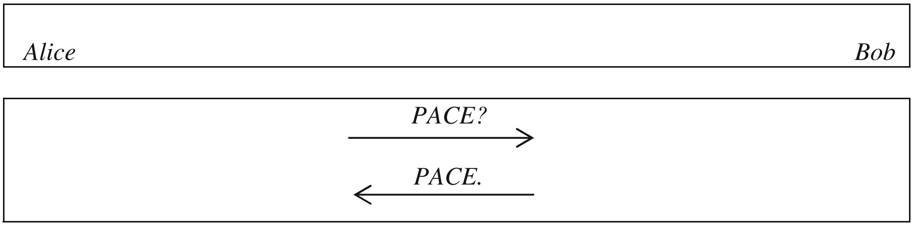

Fig. 2. Sketch of a challenge-response protocol candidate that might be beneficial.

References

1. Bender, J., Fischlin, M., Kügler, D.: Security analysis of the PACE key-agreement protocol. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, Claudio A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 33–48. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04474-8_3
2. Bellovin, S.M., Merritt, M.: Encrypted key exchange: password-based protocols secure against dictionary attacks. In: Proceedings of the IEEE Symposium on Security and Privacy. IEEE Computer Society (1992)
3. Icart, T.: How to hash into elliptic curves. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 303–316. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03356-8_18
4. Bernstein, D.J., Hamburg, M., Krasnova, A., Lange, T.: Elligator: Elliptic-curve points indistinguishable from uniform random strings. In: CCS 2013. ACM, New York (2013)
5. Düll, M., Haase, B., Hinterwäldler, G., Hutter, M., Paar, C., Sanchez, A.H., Schwabe, P.: High-speed Curve25519 on 8-bit, 16-bit and 32-bit microcontrollers. Des. Codes Cryptogr. **77**, 493–514 (2015)
6. De Clercq, R., Uhsadel, L., Van Herrewege, A., Verbauwhede, I.: Ultra low-power implementation of ECC on the ARM Cortex-M0+. In: DAC 2014 Proceedings of the 51st Annual Design Automation Conference on Design Automation Conference, pp. 1–6. ACM, New York (2014). https://www.cosic.esat.kuleuven.be/publications/article-2401.pdf. 15, 16
7. Kwon, T.: Summary of AMP (Authentication and key agreement via Memorable Passwords). http://grouper.ieee.org/groups/1363/passwdPK/contributions/ampsummary2.pdf
8. Bellovin, S., Merrit, M.: Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password-file compromise. In: ACM Conference on Computer and Communications Security, pp. 244–250 (1993)
9. Jablon, D.: Strong password-only authenticated key exchange. ACM Comput. Commun. Rev. **26**(5), 5–26 (1996)
10. Wu, T.: Secure remote password protocol. In: ISOC Network and Distributed System Security Symposium (1998)

11. MacKenzie, P.: The PAK suite: protocols for password-authenticated key exchange. In: Submission to IEEE P1363.2, April 2002
12. Kwon, T.: Authentication and key agreement via memorable password. In: ISOC Network and Distributed System Security Symposium, February 2001
13. Shin, T., Kobara, K.: Efficient augmented password-only authentication and key exchange for IKEv2 (2012). https://tools.ietf.org/html/rfc6628
14. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 544–654 (1976)
15. MacKenzie, P.: The PAK suite: protocols for password-authenticated key-exchange. DIMACS Technical report 2002-46 (2002)
16. Shin, S., Kobara, K., Imai, H.: Security proof of AugPAKE. Cryptology ePrint Archive: Report 2010/334, June 2010. http://eprint.iacr.org/2010/334
17. Federal Office for Information Security (BSI): Elliptic Curve Cryptography, Version 2.0, TR-03111, June 2012. https://www.bsi.bund.de
18. Federal Office for Information Security (BSI): Advanced Security Mechanism for Machine Readable Travel Document – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI), BSI-TR-03110, June 2012. https://www.bsi.bund.de
19. Aranha, D.F., Barreto, P.S.L.M., Pereira, G.C.C.F., Ricardini, J.E.: A note on high-security general-purpose elliptic curves (2013). https://eprint.iacr.org/2013/647.pdf
20. Hisil, H., Wong, K.K.-H., Carter, G., Dawson, E.: Twisted edwards curves revisited. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 326–343. Springer, Heidelberg (2008). doi:10.1007/978-3-540-89255-7_20
21. Lochter, M., Merkle, J.: Elliptic curve cryptography (ECC) brainpool standard curves and curve generation, IETF RFC 5639 (2010)
22. Lepinski, M., Kent, S.: Additional diffie-hellman groups for use with IETF standards, IETF RFC 5114 (2008)
23. Solinas, J.A.: Generalized Mersenne Numbers (1999)
24. Nascimento, E., Chmielwski, L., Oswald, D., Schwabe, P.: Attacking embedded ECC implementations through cmov side channels. Selected Areas in Cryptology – SAC 2016, Springer (2016, to appear)
25. Tschofenig, H., Pegourie-Gonnard, M.: Performance of state-of-the-art cryptography on ARM-based microprocessors. In: NIST LWC Workshop (2015)
26. Wenger, E., Unterluggauer, T., Werner, M.: 8/16/32 shades of elliptic curve cryptography on embedded processors. In: Paul, G., Vaudenay, S. (eds.) INDOCRYPT 2013. LNCS, vol. 8250, pp. 244–261. Springer, Cham (2013). doi:10.1007/978-3-319-03515-4_16
27. Coron, J.-S., Gouget, A., Icart, T., Pailler, P.: Supplemental access control (PACE v2): security analysis of PACE integrated mapping (2011). https://eprint.iacr.org/2011/058.pdf
28. Brier, E., Coron, J.-S., Icart, T., Madore, D., Randiram, H., Tibouchi, M.: Efficient indifferentiable hashing into ordinary elliptic curves. http://eprint.iacr.org/
29. Bernstein, D.J.: Curve25519: new Diffie-Hellman speed records (2006). https://cr.yp.to/ecdh/curve25519-20060209.pdf
30. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Horizontal correlation analysis on exponentiation. In: Soriano, M., Qing, S., López, J. (eds.) ICICS 2010. LNCS, vol. 6476, pp. 46–61. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17650-0_5
31. Batina, L., Chmielewski, Ł., Papachristodoulou, L., Schwabe, P., Tunstall, M.: Online template attacks. In: Meier, W., Mukhopadhyay, D. (eds.) INDOCRYPT 2014. LNCS, vol. 8885, pp. 21–36. Springer, Cham (2014). doi:10.1007/978-3-319-13039-2_2
32. Bernstein, D.J.: Salsa20 design (2005). https://cr.yp.to/snuffle/design.pdf

33. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: Simon and speck: block ciphers for the internet of things (2015). http://csrc.nist.gov/groups/ST/lwc-workshop2015/papers/session1-shors-paper.pdf
34. Petit, C., Quisquater, J.-J.: On polynomial systems arising from a weil descent. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 451–466. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34961-4_28
35. Gaudry, P., Hess, F., Smart, N.: Constructive and destructive facets of Weil descent on elliptic curves. J. Cryptol. **15**, 19–46 (2002). http://www.hpl.hp.com/techreports/2000/HPL-2000-10.html
36. Gaudry, P.: Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. J. Symbolic Comput. **44**, 1690–1702 (2009). http://eprint.iacr.org/2004/073