Energy Efficient ECC Authenticated Key Exchange Protocol for Star Topology Wireless Sensor Networks

Andrzej Chmielowiec, Leszek Klich, and Weronika Woś

Rzeszow University of Technology, Rzeszów, Poland

https://doi.org/10.26636/jtit.2024.1.1389

Abstract — The article proposes a new energy-efficient protocol designed for star topology wireless sensor networks. The protocol has been implemented using ECC, although it can be easily adapted to any algebraic structure, where the discrete logarithm problem is computationally challenging. In addition to the formal description, the authors provide the results of an investigation concerned with the protocol's security properties, conducted by verifying the model using Scyther software. The publication also includes an analysis of the protocol's energy consumption, performed with the use of hardware platforms with ARM microcontrollers.

Keywords — authenticated key exchange, key agreement, secure protocol, STAKE, wireless sensor network

1. Introduction

Since 1976, when Diffie and Hellman introduced the concept of public and private keys to cryptography [1], their work has been a great inspiration for numerous cryptographic algorithms and protocols. The rapid development of network technologies has compelled protocol designers to adopt various approaches while designing secure communication systems. In such cases, the most common approach involves combining the Diffie-Hellman protocol with digital signatures [2] or creating new protocols tailored to the needs of the Internet [3]–[5]. Unfortunately, in the case of many applications, traditional public key infrastructure (PKI)-based protocols cannot be employed due to hardware limitations related to memory size, transmission speed, or computational capabilities. Consequently, the literature describes numerous secure data transmission protocols tailored for specific applications.

The rapid development of the Internet of Things (IoT) has led to a vast number of small devices that are currently connected to the network. Many of these are sensors with relatively limited computational resources – too modest to implement advanced multi-sided communication systems. Often, these types of devices lack efficient cryptographic protection, exposing them to various types of threats. On the other hand, they are frequently used as components of larger infrastructures the disruption of which can have serious consequences for the modern economy.

Numerous examples of cyber attacks and hostile online activities demonstrate that destabilizing the operation of the IoT in a given region may be of interest to organizations engaged in competitive, criminal, terrorist or political activities. It is the need to ensure the security of transmissions even between seemingly insignificant devices that drives the intense development, analysis and verification of cryptographic protocols dedicated to specific applications.

A bulk of publications is devoted to solutions aimed at minimizing energy consumption. This approach is primarily pursued by applications used in devices powered by disposable batteries. For such devices, operations based on public key cryptography always account for a significant portion of the energy consumed. Therefore, minimizing the number of energy-consuming operations and simplifying the protocol are crucial. This enables devices to operate for a longer time on a single cell. Another impetus influencing the desire to create energy-efficient protocols is the global trend towards energy conservation. This trend causes customers to pay more attention to the energy certificates of devices and opt for those that offer lower energy consumption.

Wireless sensor networks (WSNs) are particularly challenging when it comes to ensuring transmission security. In many cases, these are self-organizing ad-hoc networks, where each node can serve as both a client and an access point. In networks of this type, the delivery of flexible protocols for authenticated key agreement poses a significant challenge. Examples in this area include publications by Huang *et al.* [6] and by Tian *et al.* [7], Chilveri and Nagmode [8], as well as Deng and Gao [9]. One of the main research directions is to ensure secure communication between points within the so-called IoT.

Interesting proposals in this domain have been put forward by Li *et al.* [10], Qi and Chen [11], and Srinivas *et al.* [12], Patel [13] and Das *et al.* [14]. On the other hand, articles by Wu *et al.* [15], Wei *et al.* [13], and Ok *et al.* [17] focus on authenticated key exchange in mobile 5G networks. Protocols dedicated to specific applications have been developed as well. An example here is the protocol for authenticated key exchange between two machines, authored by Thammarat and Techapanupreeda [18]. Meanwhile, Tan *et al.* [19], recognizing the shortcomings of PKI, propose an alternative security mechanism for transmission in vehicular networks.

In a large portion of new protocols, there is a clear intention to break away from classic PKI methods. In many cases, practical implementation of this infrastructure proves to be quite problematic. Authenticated key exchange based on identifiers or certificate-less cryptography constitutes an interesting alternative in these scenarios [20]– [22]. Another significant trend is the utilization of protocols based on elliptic curves [8], [21], [23]. Such an approach brings distinct benefits in terms of reducing computational and memory resources required for protocol implementation.
In this article, we introduce a new authenticated key exchange protocol designed for star topology sensor networks. In the subsequent sections, we will provide a formal description of the protocol and showcase the results of its implementation on ARM hardware-based platforms. As part of the formal description, the article will also present the outcomes of procedures verifying the protocol's security. Our analysis was performed using a well-known security model verification tool, Scyther.

2. Related Work

The issue of authenticated key exchange in wireless sensor networks is addressed in numerous publications. In this section, a concise description of the results of previous research in this area will be presented. Special attention will be given to protocols whose security is based on the discrete logarithm problem in a group of points on an elliptic curve. A summary of the essential features of the discussed protocols is presented in Tab. 1. It is worth noting that protocols lacking an authentication mechanism have been omitted. This is due to the fact that these protocols have significantly lower computational complexity than protocols offering authentication, and comparing them would not be meaningful.
In Tab. 1, four primary columns are defined, listing the protocol features subject to benchmarking. The "authentication" column pertains to the authentication method and includes one of three potential values:

- certificate – indicating the use of traditional digital certificates,
- private/public key – signifying the use of cryptographic keys for identity confirmation,
- password – denoting the utilization of a password.

The "topology" column provides information about the network topology for which the protocol has been designed and lists one of the three values:

- any – signifying any topology where each node can communicate with any other node,
- grid – representing a network comprising multiple grids, with direct communication possible between elements within the same grid, but not between elements of different grids,
- star – indicating a network with a distinguished node that any sensor can communicate with, but with the sensors not being able to communicate with each other directly.

The "EC point multiplication" and "EC point scalar product" columns denote, respectively, the number of point multiplication and scalar product operations in a single protocol run.
It should be noted that with proper implementation, point multiplication is approximately 30% faster than scalar multiplication. This is because for a field size of N bits, point multiplication requires an average of $13.3 \cdot N$ multiplications, while scalar product requires an average of $20 \cdot N$ multiplications. This calculation is based on the assumption of using Jacobian coordinates to represent points.

One of the proposals involves protocols introducing authenticated key exchange into the DSS standard, as proposed by Harn *et al.* [2]. The authors suggest three protocols based on operations in the multiplicative group of a finite field, but they can easily be extended to a group of points on an elliptic curve as well. From the point of view of this article, the interactive two-round authenticated key exchange protocol is the most interesting aspect. In its certificate-less version, this protocol is limited to star topology networks. Another protocol providing authenticated key exchange is introduced by Yao and Zhao [3]. It is also based on operations in the multiplicative group of a finite field, and like the previous one, it can successfully utilize elliptic curves. The protocol is designed for any network topology, but assumes the use of certificates, leading to an increased computational overhead.

A protocol dedicated to WSN has been developed by Huang *et al.* [6] and improved by Tian *et al.* [7] for authenticated key exchange between a sensor and a network security manager. This protocol should be considered a tool for communication in a star topology network, as it essentially allows the agreement of keys only between distinguished points of the network (security manager) and a sensor. Additionally, the protocol requires certificate verification, thus increasing computational complexity.

A completely different approach is offered by the protocol proposed by Deng and Gao [9]. They present a method for authenticated key exchange between nodes in a grid. It is characterized by the absence of certificates. In exchange, it features a central registration point which, by generating suitably crafted partial keys, enables nodes to mutually authenticate. The protocol is characterized by very high energy efficiency levels.

IoT is another area that requires efficient methods of authenticated and secure communication. Li *et al.* [10] present an interesting proposal in this regard. The goal of their protocol is to establish a connection with a specified group of N devices. The protocol is characterized by a very high constant, making its use cost-effective only when N exceeds 5.
A highly efficient key agreement protocol has been proposed by Qi and Chen in [11]. It is designed to establish a secure connection between a user and one of the sensors in the network. The gateway node (GWN) plays a special role here, as it controls the proper course of the protocol. It should be noted that the authors do not provide a formal verification of

Tab. 1. Comparison of protocol properties and their computational efficiency.

Protocol	Authentication	Topology	EC point multiplication $R = [k]P$	EC point scalar product $R = [k_1]P + [k_2]Q$
Harn *et al.* [2] (two-round)	Priv./pub. key	Star	6	2
DIKE – Yao and Zhao [3]	Certificate	Any	6	2
Huang *et al.* [6], Tian *et al.* [7]	Priv./pub. key	Star	6	2
Deng and Gao [9]	Priv./pub. key	Grid	8	0
Qi and Chen [11]	Priv./pub. key	Star	6	0
AAS-IoTSG – Srinivas *et al.* [12]	Priv./pub. key	Grid	6	0
Patel [13]	Password	Star	4	0
LACKA-IoT – Das *et al.* [14]	Priv./pub. key	Any	14	0
Wei *et al.* [16]	Priv./pub. key	Star	11	0
SA-KMP – Tan *et al.* [19]	Priv./pub. key	Any	3	5
Lu *et al.* [21]	Priv./pub. key	Any	5	4
CL-PKI (proposed)	Priv./pub. key	Star	4	2
STAKE (proposed)	Priv./pub. key	Star	6	0

the security properties of the presented solution. However, from the analysis of the protocol's execution, it is apparent that the freshness of keys is guaranteed by the control of non-cryptographic timestamps. The fact that the initiating party (user) does not have to respond to messages generated by the other party can be a starting point for potential attacks. Nevertheless, if the protocol's parties accept this level of threat, the protocol is a very fast alternative to other solutions.

Srinivas *et al.* [12] introduce an intriguing proposal called AAS-IOTSG. This protocol is designed to create secure communication between sensors and service providers. It assumes the existence of a higher-level entity, i.e. a trust anchor (TA) which participates in generating keys for each network user during registration. With this architecture, it becomes possible to securely establish connections between nodes (sensors and service providers). The only inconvenience of this protocol is that the TA has the knowledge of the private keys of individual nodes. In many applications, however, this may not be a problem, especially since a computationally efficient protocol is offered in return.

On the other hand, Patel [13] presents a computationally efficient protocol dedicated to sensors with very limited resources. By replacing strong authentication mechanisms with password-based authentication, it becomes possible to reduce the number of costly cryptographic operations.

Das *et al.* [14] propose a different approach by replacing traditional certificates with signed public keys. This allows the transfer of the capabilities of a classical public key infrastructure to the sensor network. While this involves the need for many costly cryptographic operations, it provides flexibility in establishing connections between nodes.

Authors of protocols designed for 5G networks face entirely different challenges. For example, Wei *et al.* [16] aimed to propose anonymity for the authentication service in a roaming system. The solution presented in the mentioned publication was created based on any group in which the discrete logarithm problem is computationally difficult (e.g., in a group of points on an elliptic curve). However, achieving anonymity necessitates performing many operations in the chosen group, thus resulting in high computational complexity of the proposed protocol.

On the other hand, Ok *et al.* [17] present a protocol for exchanging keys between the SIM card and a telecommunications service provider. The protocol is presented using a multiplicative group of a finite field, but it can easily be generalized to a form that allows the use of any group. Unfortunately, a significant drawback of this proposal is the lack of authentication, making it less resistant to various types of threats.

Another noteworthy protocol is the proposal presented by Tan *et al.* [19]. The authors introduce a very specific application that involves organizing secure transmissions between vehicles in a defined area. The goal of this solution is to eliminate certificates while maintaining flexibility of communication between vehicles. The protocol distinguishes three types of nodes: regional transport authority (RTA), road side unit (RSU), and vehicles. The main algorithms used by the protocol are ECDSA and EC-Schnorr. Unfortunately, the flexibility of the protocol comes at the cost of a relatively large number of cryptographic operations required. However, its primary advantage is the absence of certificates.

An older proposal presented in the article by Lu *et al.* [21] follows a somewhat similar approach. Although it does not

focus on a specific application of the protocol, it utilizes the concept of a trust center (TC) to generate data allowing for the direct authentication of users within the network.

3. Protocol Development

In the course of the protocol development phase, the following assumptions have been adopted:

– server and sensors operate in a star network topology, with the server being the central node,
– sensors do not need to communicate directly with each other,
– sensors do not act as routing points for transmitting packets to other sensors,
– before joining the network, a new sensor can be securely connected to and configured with the server. It has been decided that authenticated key agreement will be relied upon, using asymmetric methods of elliptic curve cryptography (ECC).

The security of systems based on elliptic curves is rooted in the computational difficulty of the discrete logarithm problem in the group of its points. Therefore, all algorithms and protocols utilize the point multiplication operation at some stage. This is the most computationally intensive operation which practically determines the speed of operation of the protocol and its energy consumption. Hence, minimizing the number of point multiplications is crucial in terms of energy and time efficiency. This is particularly important for small devices that usually have limited resources. Therefore, the following subsections present a proposal for a new energy-efficient protocol for authenticated key exchange – star topology authenticated key exchange (STAKE).

In our considerations, we assume that E is an elliptic curve dedicated to cryptographic applications, and $G \in E$ is one of its points whose order is known and equal to n. We also assume that H is a cryptographic hash function.

It is worth noting at this point that the entirety of the protocol has been designed in such a way that it allows for the application of a different group than an elliptic curve. In the general case, E can be regarded as any group in which a specific element G of order n has been distinguished. In this case, the security of the protocol will rely on the computational difficulty of the discrete logarithm problem within group E.

3.1. STAKE Protocol Setup

During the protocol setup phase, we will utilize assumption number 4, enabling a secure data exchange between the server and the sensor before connecting it to the network. Let us consider that based on group E, a symmetric cipher $C_e(R) = [e]R$ is realized, where $R \in \langle G \rangle \subset E$ is the plaintext and $e \in \mathbb{Z}$ is the encryption key. The decryption key for such a cipher is the number $d \in \mathbb{Z}$ that satisfies the relationship $ed = 1 \bmod n$. It can be easily shown that if the greatest common divisor $(e, n) = 1$, then using the extended Euclidean algorithm, we can efficiently find the number d with the mentioned property. Therefore, the decryption process can be expressed by:

$$C_d\big(C_e(R)\big) = [de]R = [1]R = R. \tag{1}$$

It should be noted at this point that the introduction of the above cipher into the protocol is purely a technical task, aiming to facilitate the development and verification of the formal security model.

Let us assume that server s has generated a random symmetric key $e \in \mathbb{Z}$ such that $(e, n) = 1$. Additionally, the server has also generated two pairs of asymmetric keys:

1) server key pair $(k_S, P_S) \in (\mathbb{Z} \times E)$,
2) sensor key pair $(k_M, P_M) \in (\mathbb{Z} \times E)$,

where $P_S = [k_S]G$ and $P_M = [k_M]G$. The public keys P_S and P_M also serve as identifiers for each party in the protocol. It should be noted that the server's key pair can be the same for every new sensor. Therefore, the server does not need to store a separate key pair for each sensor operating within the network.

Subsequently, the server encrypts the public keys using algorithm C_e. This results in values $K_S = [e]P_S$ and $K_M = [e]P_M$. Now, the server prepares configurations for each party in the protocol:

1) configuration for the server S: (k_S, P_S, P_M, K_M),
2) configuration for the sensor M: (k_M, P_M, K_S).

After preparing the configurations, the sensor's configuration is securely uploaded to the device.

3.2. STAKE Protocol Execution

After securely storing the appropriate keys on server s and sensor M, the protocol can be executed. Below is the communication process between the server and the sensor. It enables a secure, authenticated exchange of session keys for a symmetric cipher.

1) $S \mapsto M$: the server generates a random key n_S, calculates $Q_{1,S} = [n_S]K_M$ and sends $Q_{1,S}$ to the sensor.
2) $M \mapsto S$: the sensor receives the point $Q_{1,S}$, generates a random key n_M, calculates $Q_{1,M} = [n_M]K_S$, $Q_{2,S} = [n_M]Q_{1,S}$ and sends them back to the server.
3) $S \mapsto M$: The server receives points $Q_{1,M}$ and $Q_{2,S}$, calculates $Q_{2,M} = [n_S]Q_{1,M}$, and sends it to the sensor.
4) S: The server calculates the point $Q_{3,S} = [k_S]Q_{2,S}$ and then generates the session key $K = H(Q_{3,S})$.
5) M: The sensor calculates the point $Q_{3,M} = [k_M]Q_{2,M}$ and then generates the session key $K = H(Q_{3,M})$.

A correctly conducted protocol guarantees that the values of $Q_{3,S}$ and $Q_{3,M}$ are identical, as confirmed by the following algebraic equation:

$$\begin{aligned} Q_{3,S} &= [k_S]Q_{2,S} = [k_S\, n_M]Q_{1,S} = [k_S\, n_M\, n_S]K_M \\ &= [k_S\, n_M\, n_S\, ek_M]G = [k_M\, n_S\, n_M\, ek_S]G \\ &= [k_M\, n_S\, n_M]K_S = [k_M\, n_S]Q_{1,M} = [k_M]Q_{2,M} \\ &= Q_{3,M}. \end{aligned}$$

This means that by executing the protocol, both parties have agreed upon a valid session key.

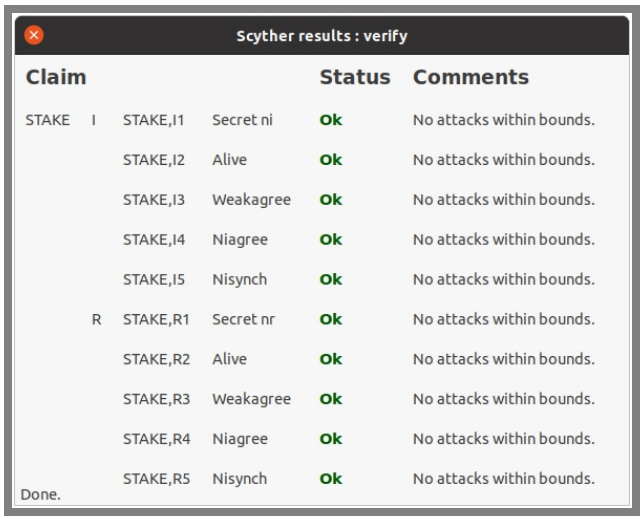

Fig. 1. Scyther software window displaying protocol verification results.

4. Verification of Protocol Security Using Model Checking

Verification of a protocol involves checking whether the protocol satisfies all the specified security properties. Model checking is one of the formal methods used for protocol verification. It considers a substantial but finite number of protocol behaviors and utilizes sophisticated algorithms to search for possible attacks on a properly prepared protocol model.

Verification of the protocol's security was performed using Scyther software. Scyther is a tool implemented under the direction of Cremers [24], [25] for the formal analysis of security protocols under the assumption of perfect cryptography, where all cryptographic functions are considered perfect – an adversary learns nothing from encrypted messages unless he possesses the decryption key.

The tool can be used to find problems resulting from the way the protocol is designed. While this problem is generally undecidable, in practice, many protocols can either be proven to be correct or potential attacks can be detected. The investigation methodology is based on article [26]. A detailed description of the basic model of Scyther is provided in [27]. This book also contains the specification of security properties and algorithms. Fundamental information about protocols is also included in [28].

Scyther takes, as input, a description of the security protocol, including the specification of intended security properties, referred to as security statements, and evaluates them. According to the convention, protocol description files have an .spdl (security protocol description language) extension.

The input language of Scyther is loosely based on a syntax similar to that of C/Java. The main purpose of the language is to describe protocols defined by a set of roles, where roles are defined by sequences of events, with a majority thereof representing data transmission or reception.

4.1. STAKE Protocol in Security Protocol Description Language

To describe the STAKE protocol using an SPDL notation, it is necessary to define two hash functions and an auxiliary protocol that specifies the behavior of these functions. These functions are named ECC and iECC. They represent the operation of point multiplication (ECC) and the inverse of this operation (iECC). These functions can be treated as keyed hash functions. It means that determining the inverse is only possible when the cryptographic key is known. Therefore, we introduce an auxiliary protocol called RemovePrivate. It formalizes the algebraic relationship between curve points, which takes the form of $[x^{-1}][yx]P = [x^{-1}xy]P = [y]P$. The following code formalizes the behavior of point multiplication as a keyed hash function in the SPDL language:

```
01. hashfunction ECC;
02. hashfunction iECC;
03.
04. protocol @RemovePrivate(T) {
05.  role T {
06.   var P: Ticket;
07.   var x: Ticket;
08.   var y: Ticket;
09.   recv_!1(T,T, iECC(ECC(ECC(P,x), y), x));
10.   send_!2(T,T, ECC(P,y));
11.  }
12. }
```

The declared hash functions and auxiliary protocol allow the definition of the STAKE protocol in the following form:

```
13. protocol STAKE(I,R) {
14.  role I {
15.   fresh ni: Nonce;
16.   var nr: Nonce;
17.   send_1(I,R, ECC({R}k(I,R), ni));
18.   recv_2(R,I, ECC(ECC({R}k(I,R), ni), nr),
       → ECC({I}k(I,R), nr));
19.   send_3(R,I, ECC(ECC({R}k(I,R), nr), ni),
20.   claim(I, Secret, ni);
21.   claim(I, Alive);
22.   claim(I, Weakagree);
23.   claim(I, Niagree);
24.   claim(I, Nisynch);
25.  }
26.  role R {
27.   fresh nr: Nonce;
28.   var ni: Nonce;
29.   recv_1(I,R, ECC({R}k(I,R), ni));
30.   send_2(R,I, ECC(ECC({R}k(I,R), ni), nr),
      → ECC({I}k(I,R), nr));
31.   recv_3(R,I, ECC(ECC({R}k(I,R), nr), ni),
32.   claim(R, Secret, nr);
33.   claim(R, Alive);
34.   claim(R, Weakagree);
35.   claim(R, Niagree);
36.   claim(R, Nisynch);
37.  }
38. }
```

The above protocol defines two roles. The first one is role I (initiator, played, in our case, by the server), and the second one is role R (responder, played by the sensor). Each use of the $ECC(X, y)$ hash function should be understood as determining the scalar multiplication $[y]X$. On the other hand, the $\{I\}k(I, R)$ and $\{R\}k(I, R)$ notations denote the server encrypted public key $K_S = [e]P_S$ and the sensor encrypted public key $K_M = [e]P_M$, respectively. The use of such notations follows directly from the assumption made at the beginning of the protocol's description, stating that the public keys of the server and the sensor serve also as their identifiers. The fresh ni and nr values are equivalents keys n_S and n_M keys generated during the protocol execution phase.
Five last lines of each described role contain information about the properties of the protocol to be examined. In this case, five properties specified in [26] are verified. These include the following:

– secret of ni and nr (session keys of the server n_S and sensor n_M remain secret during protocol execution),
– alive – both sides of the protocol are present and active during protocol execution,
– weakagree – only the initiator of the protocol believes that it was talking to the responder,
– nisynch – non-injective synchronization,
– niagree – non-injective agreement on messages.

It is important to emphasize that, in accordance with the assumptions made in [24], [25], Scyther software allows to verify the aforementioned properties only. Consequently, there are no other security properties that the program could test. This means that the above analysis is comprehensive and exhaustive in terms of the utilization of the Scyther tool.

4.2. STAKE Protocol Verification Results

After specifying the expected properties of the protocol, the verification was carried out with the following parameters:

- maximum number of runs 20,
- maximum number of patterns per claim 50.

The results clearly indicate that the model verification did not reveal any vulnerabilities to attacks within the specified scope. Figure 1 shows a window verifying the individual properties of the STAKE protocol.

5. Protocol Time and Energy Efficiency Measurements

Measuring the energy consumed by a microcontroller during cryptographic operations presents several challenges. One of them is ensuring a sufficiently high sampling frequency of the measurements of voltage and current supplied to the microcontroller. Therefore, for research purposes, a device equipped with a sensor and software for data analysis and visualization was developed. Voltage and current readings are obtained using a 12-bit INA219 ADC chip, with its block diagram shown in Fig. 2. In addition to voltage and current, the

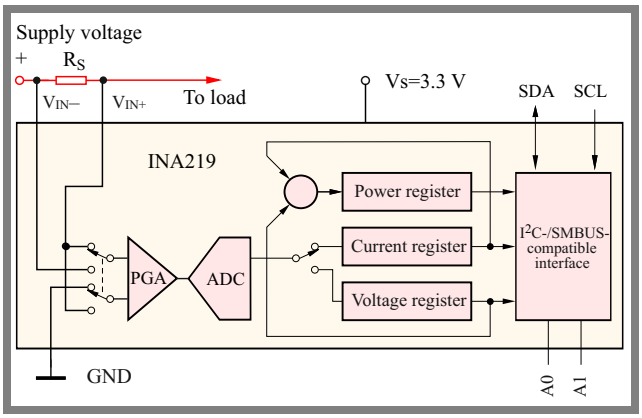

Fig. 2. INA219 integrated circuit block diagram.

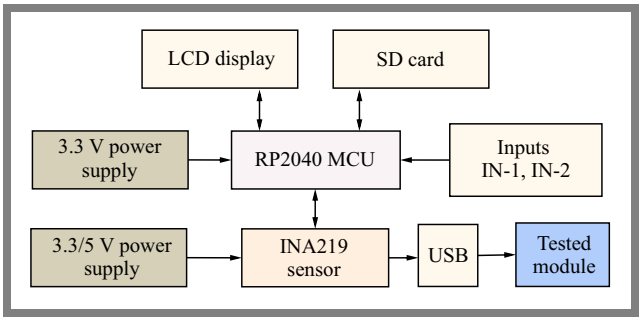

Fig. 3. Block diagram of the measurement device.

Tab. 2. Tested microcontrollers and their specifications.

MCU	Core	Clock [MHz]	RAM [KB]	Flash [KB]
STM32L053	Cortex M0	32	8	64
STM32F401	Cortex M4	84	96	512
STM32F207	Cortex M3	120	128	1024
STM32F429	Cortex M4	180	256	2048

measurement device records two digital signals originating from the microcontroller. Based on these signals, it is possible to determine what is currently being processed by the investigated microcontroller and which tested functionalities the collected data pertains to. Data acquisition and initial processing take place on an RP2040 microcontroller. The block diagram of the entire measurement device is presented in Fig. 3. The current value is determined by measuring the voltage across a 0.1 Ω shunt resistor, up to 3.2 A. The measurement can be triggered by a button or a digital signal originating from the tested circuit. This enables the measurement of the current characteristics of specific code segments with 12-bit resolution and a sampling frequency of 1880 Hz. Such a repetition is entirely sufficient for estimating the amount of energy consumed by each of the tested microcontrollers.
Four 32-bit microcontrollers based on the ARM architecture were subjected to testing: STM32L053, STM32F401, STM32F207, and STM32F429. A detailed comparison of their computational parameters is provided in Tab. 2. The STAKE protocol presented in this article has been compared to a certificate-less public key infrastructure protocol (CL-

PKI). This specific name is used to describe an ECDH key agreement protocol with digital signature-based authentication (ECDSA). Due to the topology of the network, the protocol eliminates the need for certificate and CRL verification. Therefore, it can be assumed that this version of the CL-PKI protocol is the least computationally demanding mechanism based on key agreement and digital signature algorithms.

The preparation phase of the CL-PKI protocol involves generating key pairs for the server $(k_S, P_S) \in \mathbb{Z} \times E$ and the sensor $(k_M, P_M) \in \mathbb{Z} \times E$. These keys are securely stored in the following manner:

– configuration for server S: (k_S, P_S, P_M),

– configuration for sensor M: (k_M, P_M, P_S).

After configuring both sides, it is possible to execute the authenticated key exchange protocol. The protocol proceeds as follows:

- $S \mapsto M$ – the server generates a random key n_S, calculates $Q_S = [n_S]G$ and digital signature $s_S =$ ECDSA-SIG(Q_S, k_S) and sends (Q_S, s_S) to the sensor,
- $M \mapsto S$ – the sensor receives (Q_S, s_S), generates a random key n_M, calculates $Q_M = [n_M]G$ and digital signature $s_M =$ ECDSA-SIG(Q_M, k_M) and sends (Q_M, s_M) to the server,
- S – the server receives (Q_M, s_M), verifies signature s_M using the sensor's public key P_M, calculates point $R_S = [n_S]Q_M$ and then generates session key $K = H(R_S)$,
- M – the sensor verifies signature s_S using the server's public key P_S, calculates point $R_M = [k_M]Q_S$ and then generates session key $K = H(R_M)$.

The correct execution of the protocol ensures that the values of R_S and R_M are equal, leading to the agreement of identical keys.

All cryptographic primitives implemented for testing purposes were based on the FIPS 186-4 standard [29], utilizing a defined P-192 curve. This approach enables a direct comparison between the STAKE and CL-PKI protocols, both in terms of execution time and energy consumption. The parameters for the P-192 curve are:

```
p    = 6277101735386680763835789423207666416083908700390324961279
n    = 6277101735386680763835789423176059013767194773182842284081
SEED = 3045ae6f c8422f64 ed579528 d38120ea e12196d5
c    = 3099d2bb bfcb2538 542dcd5f b078b6ef 5f3d6fe2 c745de65
b    = 64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1
G_x  = 188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012
G_y  = 07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811
```

The source code is implemented under an MIT license in an open repository at https://github.com/achmie/stake. The version available in the repository is intended for PC computers. Nevertheless, after removing debugging prints and adding appropriate measurement instructions, it is possible to replicate the experiments. It should be noted that the authors intentionally failed to include a version of the software with measurement instructions, since these are specific to a particular evaluation board and dedicated measurement devices. The software testing the functionality of the protocols was built using the Arduino environment with default settings. The compilation was performed using the g++ compiler from the GCC 12.2 package, with the -Os (code size optimization) and gnu++17 (C++ language standard version) options.

Tab. 3. Average execution times for STAKE and CL-PKI protocols.

MCU	STAKE	CL-PKI	Speedup
	Time [ms]	Time [ms]	Ratio
STM32L053	882	1022	13.7%
STM32F401	104	118	11.8%
STM32F207	90	110	18.2%
STM32F429	50	76	34.2%

Tab. 4. Average energy consumption for STAKE and CL-PKI protocols.

MCU	STAKE	CL-PKI	Saving
	Energy [mJ]	Energy [mJ]	Ratio
STM32L053	242	280	13.6%
STM32F401	37	42	11.9%
STM32F207	104	130	20.0%
STM32F429	45	71	36.6%

Figures 4–7 present current consumption over time for the tested microcontrollers. Each of these figures contains two graphs with identical variable ranges, allowing for a comparison of STAKE and CL-PKI protocol profiles. As the measurements aim to reflect the energy consumed by each protocol, the transmission part has been omitted, and only the energy required for cryptographic computations has been taken into consideration. Additionally, the graphs separate the current consumption for each side of the protocol. The red line represents current consumption on the server side, while the blue line represents current consumption on the sensor side. It is also assumed that the microcontroller enters a sleep state while waiting for the other party's response. The presented graphs clearly show that the STAKE protocol is significantly faster than the CL-PKI protocol. Table 3 presents averaged execution times for the two compared protocols. The obtained results demonstrate that the proposed STAKE protocol is approximately 12% to even 34% faster on the tested processors. The main reason for this outcome is that STAKE only performs point multiplication operations, while CL-PKI requires that the ECDSA signature be generated and verified.

The presented graphs show relatively stable current consumption levels during the execution of cryptographic operations. Therefore, one can anticipate that a shorter protocol execu-

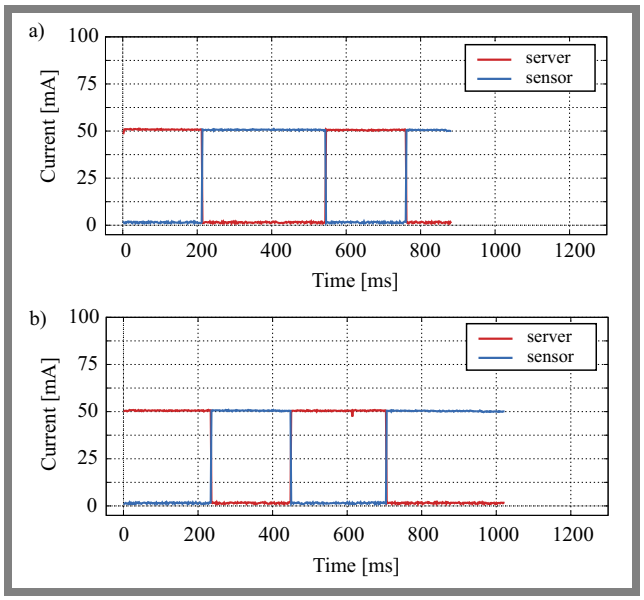

Fig. 4. Current consumption and execution time of: a) STAKE and b) CL-PKI protocols on the STM32L053 microcontroller.

tion time will also result in lower energy consumption. The averaged energy consumption has been determined for both protocols and each of the tested microcontrollers.

The experimental results are presented in Tab. 4. It can be observed that the savings ratio calculated in the last microcontroller column is very close to the values of computational acceleration presented in Tab. 3. Hence, it can be concluded that the proposed STAKE protocol is approximately 12% to 37% more energy-efficient compared to the CL-PKI protocol, for the tested microcontrollers. Therefore, one may conclude that the implementation of the STAKE protocol can be highly beneficial in applications with battery-powered sensors.

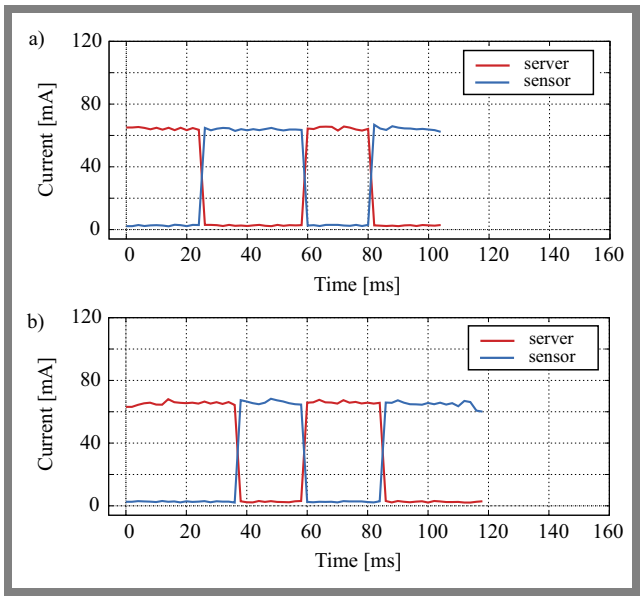

Fig. 5. Current consumption and execution time of: a) STAKE and b) CL-PKI protocols on the STM32F401.

6. Conclusions

The article presents a new authenticated key exchange protocol, referred to as STAKE. This protocol is designed specifically for star-shaped network topologies. In particular, it can be used to ensure secure communication between nodes/sensors and a distinguished gateway node. Based on the verification of the SPDL model using Scyther software, it has been demonstrated that the protocol possesses all necessary security features.

Simultaneously, protocol implementation tests and comparisons with fundamental PKI mechanisms have revealed faster operation and lower energy consumption. It should be emphasized that the comparability of the STAKE and CL-PKI

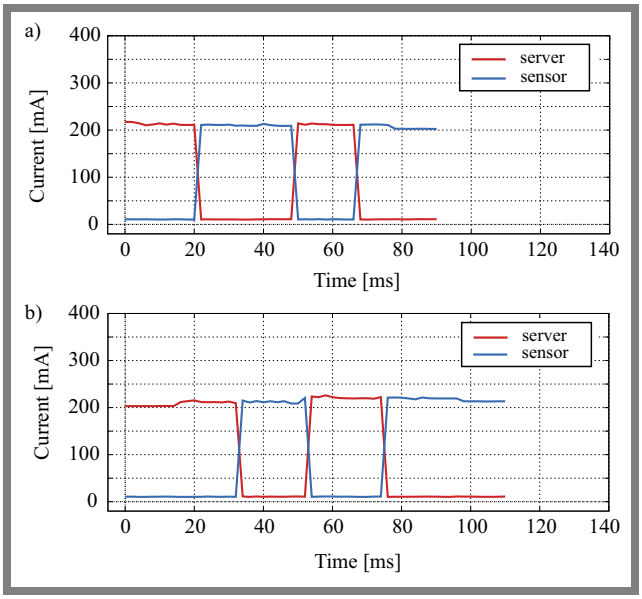

Fig. 6. Current consumption and execution time of: a) STAKE and b) CL-PKI protocols on the STM32F207 MCU.

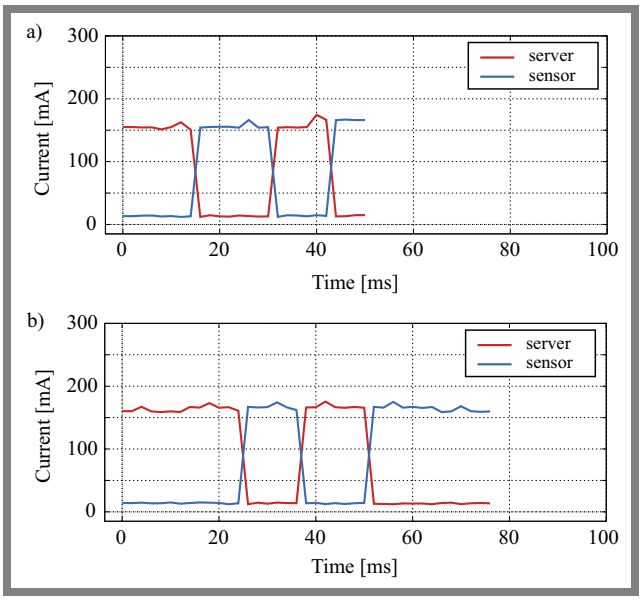

Fig. 7. Current consumption and execution time of: a) STAKE and b) CL-PKI protocols on the STM32F429.

protocols was achieved by employing the same cryptographic primitives operations on a standard FIPS 186-4 elliptic curve P-192.

The conducted experiments have shown that depending on the processor used, energy savings of up to 37% can be achieved. Therefore, this protocol may be used primarily in battery-powered WSNs. If we compare STAKE to other protocols presented in Table 1, it is clear that it falls into the category of protocols with the lowest computational requirements, similar to those typical of the solutions proposed by Qi and Chen [11] and Srinivas *et al.* [12]. In terms of performance, it only fails to match the protocol proposed by Patel [13] in which authentication is performed using a password.

Additionally, the STAKE protocol is simple to implement and does not require extensive cryptographic libraries. This is a clear advantage, especially if one plans to use it in very small microcontrollers. It is also important to consider certain characteristics of the protocols described in [11] and [12] which may rule them out in the case of specific applications. Therefore, STAKE can be an interesting alternative.

References

[1] W. Diffie and M. Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976 (https://doi.org/10.1109/TIT.1976.1055638).

[2] L. Harn, M. Mehta, and W.-J. Hsin, "Integrating Diffie-Hellman Key Exchange into the Digital Signature Algorithm (DSA)", *IEEE Communications Letters*, vol. 8, no. 3, pp. 198–200, 2004 (https://doi.org/10.1109/LCOMM.2004.825705).

[3] A.C.-C. Yao and Y. Zhao, "Privacy-preserving Authenticated Key-exchange over Internet", *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 1, pp. 125–140, 2014 (https://doi.org/10.1109/TIFS.2013.2293457).

[4] T. Yi, M. Shi, and W. Shang, "Personalized Two Party Key Exchange Protocol", in: *2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS)*, Las Vegas, USA, pp. 575–579, 2015 (https://doi.org/10.1109/ICIS.2015.7166659).

[5] A.S. Rawat and M. Deshmukh, "Efficient Extended Diffie-Hellman Key Exchange Protocol", in: *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, New Delhi, India, pp. 447–451, IEEE, 2019 (https://ieeexplore.ieee.org/document/8940607).

[6] Q. Huang, J. Cukier, H. Kobayashi, B. Liu, and J. Zhang, "Fast Authenticated Key Establishment Protocols for Self-organizing Sensor Networks", in: *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*, San Diego, USA, pp. 141–150, 2003 (https://doi.org/10.1145/941350.941371).

[7] X. Tian, D.S. Wong, and R.W. Zhu, "Analysis and Improvement of an Authenticated Key Exchange Protocol for Sensor Networks", *IEEE Communications Letters*, vol. 9, no. 11, pp. 970–972, 2005 (https://doi.org/10.1109/LCOMM.2005.11006).

[8] P.G. Chilveri and M.S. Nagmode, "A Novel Node Authentication Protocol Connected with ECC for Heterogeneous Network", *Wireless Networks*, vol. 26, pp. 4999–5012, 2020 (https://doi.org/10.1007/s11276-020-02358-4).

[9] L. Deng and R. Gao, "Certificateless Two-party Authenticated Key Agreement Scheme for Smart Grid", *Information Sciences*, vol. 543, pp. 143–156, 2021 (https://doi.org/10.1016/j.ins.2020.07.025).

[10] Z. Li, Z. Yang, P. Szalachowski, and J. Zhou, "Building Low-Interactivity Multifactor Authenticated Key Exchange for Industrial Internet of Things", *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 844–859, 2021 (https://doi.org/10.1109/JIOT.2020.3008773).

[11] M. Qi and J. Chen, "Secure Authenticated Key Exchange for WSNs in IoT Applications", *The Journal of Supercomputing*, vol. 65, no. 3, pp. 1–14, 2021 (https://doi.org/10.1007/s11227-021-03836-y).

[12] J. Srinivas, A.K. Das, X. Li, M.K. Khan, and M. Jo, "Designing Anonymous Signature-based Authenticated Key Exchange Scheme for Internet of Things-enabled Smart Grid Systems", *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4425–4436, 2021 (https://doi.org/10.1109/TII.2020.3011849).

[13] C. Patel and N. Doshi, "Secure Lightweight Key Exchange Using ECC for User-gateway Paradigm", *IEEE Transactions on Computers*, vol. 70, no. 11, pp. 1789–1803, 2021 (https://doi.org/10.1109/TC.2020.3026027).

[14] A.K. Das, M. Wazid, A.R. Yannam, J.J.P. Rodrigues, and Y. Park, "Provably Secure ECC-based Device Access Control and Key Agreement Protocol for IoT Environment", *IEEE Access*, vol. 7, pp. 55382–55397, 2019 (https://doi.org/10.1109/ACCESS.2019.2912998).

[15] T.-Y. Wu *et al.*, "An Authenticated Key Exchange Protocol for Multi-server Architecture in 5G Networks", *IEEE Access*, vol. 8, pp. 28096–28108, 2020 (https://doi.org/10.1109/ACCESS.2020.2969986).

[16] F. Wei, P. Vijayakumar, Q. Jiang, and R. Zhang, "A Mobile Intelligent Terminal Based Anonymous Authenticated Key Exchange Protocol for Roaming Service in Global Mobility Networks", *IEEE Transactions on Sustainable Computing*, vol. 5, no. 2, pp. 268–278, 2020 (https://doi.org/10.1109/TSUSC.2018.2817657).

[17] K. Ok, V. Coskun, C. Cevikbas, and B. Ozdenizci, "Design of a Key Exchange Protocol Between SIM Card and Service Provider", in: *2015 23rd Telecommunications Forum (TELFOR)*, pp. 281–284, IEEE, 2015 (https://doi.org/10.1109/TELFOR.2015.7377465).

[18] C. Thammarat and C. Techapanupreeda, "A Secure Authentication and Key Exchange Protocol for M2M Communication", in: *2021 9th International Electrical Engineering Congress (IEECON)*, Pattaya, Thailand, pp. 456–459, 2021 (https://doi.org/10.1109/iEECON51072.2021.9440355).

[19] H. Tan *et al.*, "A Secure and Authenticated Key Management Protocol (SA-KMP) for Vehicular Networks", *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9570–9584, 2016 (https://doi.org/10.1109/TVT.2016.2621354).

[20] L. Xiaoyong and Z. Hui, "Identity-based Authenticated Key Exchange Protocols", in: *2010 International Conference on Educational and Information Technology*, Chongqing, China, vol. 3, pp. 85–87, 2010 (https://doi.org/10.1109/ICEIT.2010.5608417).

[21] E.-H. Lu, W.-T. Ko, and H.K.-C. Chang, "A Novel ECC Protocol for Key Exchanging in Non-PKI Networks", in: *Proceedings of 2011 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*, Harbin, China, vol. 1, pp. 822–825, 2011 (https://doi.org/10.1109/CSQRWC.2011.6037077).

[22] S. Jarecki, M. Jubur, H. Krawczyk, N. Saxena, and M. Shirvanian, "Two-factor Password-authenticated Key Exchange with End-to-end Security", *ACM Transactions on Privacy and Security (TOPS)*, vol. 24, no. 3, pp. 1–37, 2021 (https://doi.org/10.1145/3446807).

[23] A. Abusukhon, Z. Mohammad, and A. Al-Thaher, "Efficient and Secure Key Exchange Protocol Based on Elliptic Curve and Security Models", in: *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, Amman, Jordan, pp. 73–78, 2019 (https://doi.org/10.1109/JEEIT.2019.8717496).

[24] C. Cremers, *Scyther: Semantics and Verification of Security Protocols*, PhD thesis, Mathematics and Computer Science, Eindhoven University of Technology, 2006 (https://doi.org/10.6100/IR614943).

[25] C. Cremers, "The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols", *Lecture Notes in Computer Science*, vol. 5123, pp. 414–418, Springer, 2008 (https://doi.org/10.1007/978-3-540-70545-1_38).

[26] G. Lowe, "A Hierarchy of Authentication Specifications", in: *Proceedings 10th Computer Security Foundations Workshop*, Rockport, USA, pp. 31–43, 1997 (https://doi.org/10.1109/CSFW.1997.596782).

[27] C. Cremers and S. Mauw, *Operational Semantics and Verification of Security Protocols*, Springer-Verlag, 188 p., 2012 (ISBN: 9783540786351) (https://doi.org/10.1007/978-3-540-78636-8).

[28] L. Dong and K. Chen, *Cryptographic Protocol. Security Analysis Based on Trusted Freshness*, Springer-Verlag, 384 p., 2012 (ISBN: 9783642240720) (https://doi.org/10.1007/978-3-642-24073-7).

[29] National Institute of Standards and Technology (NIST), "Digital Signature Standard (DSS)", *Federal Information Processing Standards Publication*, FIPS PUB 186-4, 2013 (https://doi.org/10.6028/NIST.FIPS.186-4).

Andrzej Chmielowiec, Ph.D., Assistant Professor
Department of Computerization and Robotization of Industrial Processes, Faculty of Mechanics and Technology
https://orcid.org/0000-0001-6629-0029
E-mail: achmie@prz.edu.pl
Rzeszow University of Technology, Rzeszów, Poland
https://www.prz.edu.pl

Leszek Klich, Ph.D., Assistant Professor
Department of Computerization and Robotization of Industrial Processes, Faculty of Mechanics and Technology
https://orcid.org/0000-0001-6099-2417
E-mail: l.klich@prz.edu.pl
Rzeszow University of Technology, Rzeszów, Poland
https://www.prz.edu.pl

Weronika Woś, Ph.D., Assistant Professor
Department of Computerization and Robotization of Industrial Processes, Faculty of Mechanics and Technology
https://orcid.org/0000-0003-4709-4369
E-mail: weronikawos@prz.edu.pl
Rzeszow University of Technology, Rzeszów, Poland
https://www.prz.edu.pl