INFORMATICA, 2020, Vol. 31, No. 2, 277–298
© 2020 *Vilnius University*
DOI: https://doi.org/10.15388/20-INFOR415

Authenticated Key Agreement Protocol Based on Provable Secure Cryptographic Functions

Ausrys KILCIAUSKAS[1], Gintaras BUTKUS[2], Eligijus SAKALAUSKAS[3,*]

[1] *Department of Applied Informatics, Vytautas Magnus University, Lithuania*
[2] *Department of Informatics, Kauno kolegija / University of Applied Science, Lithuania*
[3] *Department of Applied Mathematics, Kaunas University of Technology, Lithuania*
e-mail: eligijus.sakalauskas@ktu.lt

Received: January 2020; accepted: March 2020

Abstract. The vulnerable part of communications between user and server is the poor authentication level at the user's side. For example, in e-banking systems for user authentication are used passwords that can be lost or swindled by a person maliciously impersonating bank.

To increase the security of e-banking system users should be supplied by the elements of public key infrastructure (PKI) but not necessary to the extent of standard requirements which are too complicated for ordinary users.

In this paper, we propose two versions of authenticated key agreement protocol (AKAP) which can be simply realized on the user's side. AKAP is a collection of cryptographic functions having provable security properties.

It is proved that AKAP1 is secure against active adversary under discrete logarithm assumption when formulated certain conditions hold. AKAP2 provides user's anonymity against eavesdropping adversary. The partial security of AKAP2 is investigated which relies on the security of asymmetric encryption function.

Key words: cryptography, identification, key agreement protocol, asymmetric encryption, e-signature.

1. Introduction

The vulnerable part of communications between user and server is the poor authentication level on the user's side. For example, in e-banking systems for user authentication are used passwords that can be lost or swindled by a person maliciously impersonating bank.

Nowadays appeared Smart-Id identification using smart phones has some advantages compared with the bank's supplied passwords table to the user, but nevertheless it is a temporary measure to mitigate the increasing number of attacks to e-banking system.

Despite the fact that public key infrastructure (PKI) and certificates based identification exists for the 5-10 years, newly appeared Smart-Id identification becomes more popular. The reason is the complexity of PKI in traditional public key settings and the

*Corresponding author.

key escrow problem in ID-based public key settings. In this connection the alternative certificate-based signature is proposed as an attractive public key setting, which reduces the complexity of PKI and resolves the key escrow problem (Tseng *et al.*, 2019).

Authors proposed a new and efficient certificate-based signature (CBS) scheme from lattices. Under the short integer solution (SIS) assumption from lattices, the proposed CBS scheme is shown to be existential unforgeability against adaptive chosen message attacks.

The other alternative is certificateless signature that has become a widely studied paradigm. This paradigm has a lack of key escrow problem and certificate management problem. But the problem of this primitive was non-resistance to catastrophic damage caused by key exposure. New results in this field are presented in (Mei *et al.*, 2019).

Oulined above perspective solutions are in the investigation and developing stage so far.

The one of currently available solutions can be the cryptographic chip implemented in the user's smartphone or in his credit card. This cryptographic chip could be supplied by the bank to the user with public key cryptosystem (PKC) parameters and supporting software. This software can be used to more secure authentication and communication session creation using authenticated key agreement protocol (AKAP).

In this case smart phone can provide much more functions to the customer. For example, it can be used as e-purse for off-line payments (Muleravicius *et al.*, 2019).

Then user may not communicate with bank for any money transfer. It is enough to communicate with bank for withdrawal and deposit money to and from e-purse respectively using AKAP.

Moreover, AKAP can be combined together with biometric identification methods which popularity is growing nowadays but not so rapid as desirable.

In general, the user has significantly less computing power than server and therefore AKAP realization should need as small computation resources as possible.

We will consider two legal parties communication with each other, namely user Alice and Bank and an adversary. We assume that in all cases adversary has public keys of both parties and system parameters (SP) of a used cryptographic system. We consider the following type of attacks:

- Eavesdropping attacks: the adversary can eavesdrop on the legal communications between parties and can obtain the transcript of several interactions between them. As a consequence, the adversary can decrypt secret messages or compromise the secret key.
- Active attacks: the adversary uses the interaction to try and learn something that will let it impersonate Alice to the Bank and the Bank to Alice. Suppose Alice runs an identification protocol with the Bank over the internet. An active adversary controls the channel and can block or inject messages at will. The adversary waits for Alice to run the identification protocol with the Bank and relays all protocol messages from one side to the other. Once the identification protocol completes successfully, the adversary sends requests to the Bank that appear to be originating from Alice. The bank honors these requests, thinking that they came from Alice. In effect, the adversary uses Alice to authenticate to the Bank and then "hijacks" the session to send his own messages to

the Bank. As a consequence of these attacks, the adversary can decrypt secret messages exchanged between parties or compromise their secret keys.

Active attacks are more powerful than eavesdropping attacks. They come up when Alice tries to login from a local infected computer. The malware infecting the computer could display a fake login screen and fool Alice into interacting with it, thus mounting an active attack.

One of the very "popular" kinds of attack is a Man-in-the-Middle (MiM) attack. The HTTPS protocol is vulnerable to this kind of attack (Callegati *et al.*, 2009). An attacker capable of eavesdropping on traffic is also able to inject its own messages. The protocol completely falls apart in the presence of an active adversary who controls the network. The main reason is the lack of authentication. Alice sets up a shared secret, but she has no idea with whom the secret is shared. The same holds for the Bank. An active attacker can abuse this to expose all traffic between Alice and the Bank. This attack works against any key exchange protocol that does not include authentication. Moreover, neither KAP, nor identification protocols alone are secure against the MiM attack (Boneh and Shoup, 2020).

In 2015.03.17 Euronews made a report that on-line banking might be full of holes like Swiss Emmental cheese, http://www.euronews.com/2015/03/17/internet-banking-a-hacker-s-ideal-target/.

The reasons of this situation which has not significantly changed so far are outlined above, therefore, the measures must be implemented to protect the user especially against active adversary attacks.

To realize secure AKAP it is required to have a combination of several cryptographic primitives: key agreement protocol, identification protocol, digital signature, and asymmetric encryption.

To provide secure communications between Alice and the Bank it is required that Alice prove to the Bank her identity and that the Bank prove to Alice its identity. One party proving one's identity is named a Prover – P and the other party verifying this proof is a Verifier – V. Hence, to create secure communications both parties should be both P and V to each other. This kind of identification is called mutual identification.

Secure identification protocols are based on the interaction between the P and V. They use a technique called challenge-response identification (Just, 2011) together with other protocols including key agreement protocol (KAP) thus yielding authenticated key agreement protocol (AKAP).

The aim of this paper is to present integrated AKAP between two parties: user Alice and the Bank using well known cryptographic primitives with provable security. AKAP should have the following properties:

- Secure mutual authentication between Alice and the Bank and session key agreement.
- Effective realization especially on the user's side.
- Alice's anonymity against eavesdropping and active adversary.

Security analysis of proposed AKAP is presented referencing to security assumptions of cryptographic primitives used in our construction.

Effective realization means that computations and communications should be minimized. It is also desirable that the number of system parameters should be minimized as well. The number of these parameters depends on the selection of suitable cryptographic protocols. Several cryptographic protocols and schemes are having the same system parameters such as ElGamal cryptosystem (ElGamal, 1985) together with the same private and public keys:

- Diffie–Hellman key Agreement protocol (DH-KAP),
- ElGamal encryption (ElG-Enc),
- ElGamal signature (ElG-Sig),
- Schnorr identification protocol (S-Id),
- Schnorr signature (S-Sig).

These protocols are realized using the same discrete exponent functions dexp() in multiplicative cyclic groups of finite order. Some of them can be realized in elliptic curve groups. We will consider numerical groups, where operations are performed modulo large prime number p.

Two protocols AKAP1 and AKAP2 are considered. AKAP1 is a simpler protocol that does not provide user's anonymity. AKAP2 provides user's anonymity by adding additional encryption in the first communication round.

In the list above we have two signature schemes, namely ElGamal and Schnorr. We present here some analyses allowing us to choose a unique scheme better matching our requirements. The signature scheme we use as an additional authentication means from the Bank's side. It is an optional measure since the Bank has a qualified e-signature certificate and can be authenticated by the user's browser and during the execution of SSL/TLS protocol.

ElGamal signature scheme (ElGamal, 1985) is based on the discrete exponent function.

The original paper did not include a hash function as a system parameter. The message m was used directly in the algorithm instead of $\mathrm{H}(m)$. This enables an attack called an existential forgery, as described in the paper of Pointcheval and Stern (Pointcheval and Stern, 2000).

ElGamal signature scheme (ElGamal, 1985) is vulnerable to the Bleichenbacher attack (Bleichenbacher, 1996).

This attack is avoided by using groups G_q of prime order q. The main drawback of ElGamal signature is that it has considerable long keys.

Due to these considerations, we choose the Schnorr signature in our construction. It is a new variant of the ElGamal signature which overcomes the drawbacks, namely: a long signature size and Bleichenbacher attack.

Schnorr identification and signatures (Schnorr, 1990, 1991) constitute one of the most fundamental public-key cryptosystems.

Pointcheval and Stern (1996, 2000) have shown that it is provably secure, assuming the hardness of the discrete logarithm (DL) problem in the Random Oracle Model (Bellare and Rogaway, 1993; Neven *et al.*, 2009; Seurin, 2012).

Schnorr identification protocol is based on the exchange of challenge-response conversations between prover P and verifier V when P is seeking to prove to V some parameters associated with his/her identity. In our case the prover is Alice and the verifier is the Bank. The process of proof is based on the exchange of messages between P and V and is called conversation. In Schnorr identification protocol conversation consists of three rounds:

1. P computes a commitment and sends it to V.
2. V generates a challenge and sends it to P.
3. P computes a response and sends it to V.

Both P and V are actively involved in the conversation, and the timing and ordering of the messages are critical. The active adversary playing the role of a prover must generate the first message before it sees the challenge generated by V.

To achieve the security of the AKE protocol against the active adversary, one must carefully intertwine the processes of identification and anonymous key exchange. The adversary actively impersonates a legitimate verifier. For example, the adversary may clone a banking site and wait for a user being a prover P to visit the site. When it occurs P runs the identification protocol with the adversary. As a result, the adversary repeatedly interacts with P on the behalf of verifier V and sends the prover arbitrary messages of its choice. After several such interactions, the adversary turns around and attempts to authenticate himself as the prover to a legitimate verifier V. Identification protocol is secure against active attacks if the adversary still cannot fool the legitimate verifier V.

In this paper we define security assumptions and provide security proof of AKAP1 against an active adversary. Security proof is based on transforming S-Id to AKAP1 which represents the so called sigma protocol (Boneh and Shoup, 2020). Unfortunately, the similar security proof for AKAP2 is not possible since AKAP2, being a more complex protocol, does not satisfy sigma protocol's conditions. But nevertheless, AKAP2 seems to be more secure than AKAP1. Hence, so far the security of AKAP2 can be based only on the security of its cryptographic components listed above.

The other objective of this paper is to try to extend these results to the other conjectured one-way functions (OWF) having some similarity with used here dexp() function. For example, new conjectured OWF based on so called matrix power function (MPF) was proposed earlier in our papers (Sakalauskas *et al.*, 2008, 2017; Sakalauskas and Mihalkovich, 2014, 2017; Sakalauskas, 2018). In Sakalauskas and Mihalkovich (2018) it is proved that inversion of MPF corresponds to NP-complete problem. This proof was based on the result presented in Sakalauskas (2012).

The structure of the paper is the following. To be self-contained, in Section 2 we present some mathematical background and describe cryptographic protocols and functions used in our construction. In Section 3 we present AKAP1 and AKAP2 description. Section 3 is dedicated to security analysis. In Section 4 conclusions and a look to the future work are presented.

2. Preliminaries

We are dealing with a cyclic group G_q of prime order q with generator g. In our case G_q is a subgroup of the cyclic multiplicative group of integers $Z_p^* = \{1, 2, \ldots, p-1\}$ where p is prime and multiplication is performed modulo p. Prime p is of n bit length, where n is a security parameter.

Since q is a prime factor of $p-1$, then according to Lagrange's theorem and its consequences all elements of G_q are generators. Then for all g in G_q the following identity holds

$$g^q = 1 \bmod p. \tag{2.1}$$

This identity allows checking if number $g \in Z_p^*$, $g \neq 1$ is a generator in G_q.

Let $g \in G_q$ be a generator and x be an integer $1 \leqslant x \leqslant q-1$, then discrete exponent function dexp() in G_q is defined as follows:

$$d\exp(x) = g^x \bmod p = a, \quad a \in G_q. \tag{2.2}$$

The inverse function to dexp() is a discrete logarithm function $d\log_g(a)$ and is defined as follows:

$$d\log_g(a) = x \bmod (q-1), \tag{2.3}$$

where generator g is a discrete logarithm function's base defined in (2.2).

If g is a generator in G_q then function dexp() is one-to-one and performs the following mapping

$$\text{dexp} : Z_{q-1} \to G_q, \tag{2.4}$$

where $Z_{q-1} = \{0, 1, 2, \ldots, q-1\}$ is a ring of integers with addition and multiplication modulo q. This mapping plays a very important role in security considerations of cryptographic protocols based on dexp() function.

The necessary but not sufficient security assumption for all protocols presented above is discrete logarithm assumption and associated discrete logarithm problem (DLP).

Definition 2.1. Discrete Logarithm Problem – DLP is to find x in (2.2) when g, p and a are given.

Definition 2.2. Discrete logarithm assumption. We say that the discrete logarithm (DL) assumption holds for G_q if the probability to find x in (2.2) when g, p and a are given is negligible.

We will need a notion of one-way function (OWF) which we define in the following non-formal way.

Definition 2.3. Let $F : A \to B$ be a function. Function F is said to be one-way if: 1) for given $x \in A$, it is computationally easy to compute $y = F(x)$, which corresponds to the direct F value computation; 2) for given $y \in B$, it is computationally hard to compute (at least single) $x \in A$ such that $F(x) = y$, which corresponds to the inverse F value computation.

Conjecture 2.4. The discrete exponent function is a candidate OWF.

Indeed, the computation of g^x mod p can be done efficiently even for large numbers commonly referred to as square-and-multiply algorithms. But its inverse value computation corresponds to DLP and is reckoned as hard using classical (non-quantum) computers. But nevertheless, due to Shor (1997) DLP can be solved in polynomial-time using quantum algorithms running on quantum computers.

For example, when p and q are sufficiently large and suitably chosen primes the discrete logarithm problem in the group G_q being a subgroup of Z_{q-1} is believed to be hard to compute. Prime p should be at least 2048-bits, and q should be at least 256-bits.

All cryptographic primitives presented in the introduction are using the same system parameters $\mathrm{SP} = (p, g)$, namely large (secure) prime number p and generator g of group G_q.

To generate random and uniformly distributed parameters for cryptographic protocols we use a special notation. For example, if we uniformly choose a random element r from the set S then we write:

$$r = \mathrm{rand}(S). \tag{2.5}$$

We assume that SP are generated by the Bank. The Bank generates a prime number p of at least 2048 bits length, i.e. $|p| = 2048$. Prime p should be suitably chosen in such a way that $(p-1)$ should have a prime divider q of 256 bit length, i.e. $|q| = 256$. Then the Bank finds a generator g of defined above group G_q.

According to ElGamal cryptosystem, the Bank randomly generates its private key $\mathrm{PrK}_B = y$, where

$$y = \mathrm{rand}(Z_q), \quad y \in Z_q,\ 1 < y < q. \tag{2.6}$$

Then corresponding to its private key the public key $\mathrm{PuK}_B = b$, is computed

$$b = g^y \bmod p, \quad b \in G_q. \tag{2.7}$$

System parameters $\mathrm{SP} = (p, g)$ and the Bank's $\mathrm{PuK}_B = b$ are openly distributed among all the Bank's customers including Alice.

When user Alice opens her account in the Bank, then during the registration phase she receives $\mathrm{SP} = (p, g)$ and the Bank's $\mathrm{PuK}_B = b$.

In addition, there are two opportunities for Alice to complete the registration operation. Either she receives the Bank's generated public and private key pair $\mathrm{PrK}_A = x$, $\mathrm{PuK}_A = a$, for her, where

$$x = \mathrm{rand}(Z_q), \quad x \in Z_q,\ 1 < x < q, \tag{2.8}$$

$$a = g^x \bmod p, \tag{2.9}$$

or she generates this key pair by herself using special certified application software supplied by the Bank. In the latter case Alice keeps secret her $\mathrm{PrK}_A = x$ from everyone (including the Bank).

In both cases all parameters mentioned above are kept in certain storage devices (e.g. USB token, SIM card, Smart phone apps, etc.) together with the certified application program.

Every user including Alice has system parameters $\mathrm{SP} = (p, g)$, the Bank's $\mathrm{PuK}_B = b$, her $\mathrm{PuK}_A = a$ and her $\mathrm{PrK}_A = x$.

In our model the Adversary can know two alternative sorts of information: either system parameters $\mathrm{SP} = (p, g)$, the Bank's $\mathrm{PuK}_B = b$ and user's public key, e.g. $\mathrm{PuK}_A = a$ or he may know only SP and PuK_B. In the latter case, PuK_A is not openly transmitted during AKAP.

To be self-contained we present here a description of protocols and functions used for AKAP construction.

2.1. *Diffie–Hellman Key Agreement Protocol (DH-KAP)*

Let Alice be the initiator of the DH-KAP protocol with the Bank. It is executed in two communications between Alice and the Bank:

1. Alice generates a random secret number $u = \mathrm{rand}(Z_q)$, $1 < u < q - 1$ and using $\mathrm{SP} = (p, g)$ computes a non-secret session parameter
$$k_A = g^u \bmod p. \tag{2.10}$$
Alice sends k_A to the Bank.
2. After receiving Alice's message, the Bank generates a random secret number $v = \mathrm{rand}(Z_q)$, $1 < v < q - 1$ and using $\mathrm{SP} = (p, g)$ computes his session parameter
$$k_B = g^v \bmod p. \tag{2.11}$$
The Bank sends k_B to Alice.

After this open data exchange, Alice and the Bank compute their common agreed secret key $k_{AB} = k_{BA} = k$, $k_{AB} = (k_B)^u \bmod p = (g)^{vu} \bmod p = k_{BA} = (k_A)^v \bmod p = (g)^{uv} \bmod p = k$.

So Alice and the Bank can create a secure channel for encrypted communications between each other.

If $|p| = 2048$ bits and $|q| = 256$ bits, then the maximal number of exponentiation operations from $2 * 2048 = 4096$ is reduced to $2 * 256 = 512$ for each party to compute the agreed key k.

Unfortunately, the discrete logarithm assumption by itself is not enough to ensure that the Diffie–Hellman protocol is secure. The following definition and assumption of Computation Diffie–Hellman (CDH) problems are required.

Definition 2.5. CDH problem in G_q is to compute g^{uv} when g^u and g^v are given.

Definition 2.6. CDH assumption in G_q states that it is infeasible to compute g^{uv} when g^u and g^v are given.

To compromise DH-KAP the eavesdropper has to solve the CDH problem which is stronger than DLP. Some evidence still suggests that this is a reasonable assumption in groups where the DL assumption holds but CDH does not. In DH-KAP, an eavesdropper observes g^u and g^v exchanged as part of the protocol, and the two parties both compute the shared key g^{uv}. A fast means of solving the CDH problem would allow an eavesdropper to violate the privacy of the Diffie–Hellman key exchange by compromising the agreed secret key.

The stronger assumption for the non-ephemeral agreed key is decisional DH, (DDH) assumption (Boneh, 1998).

Definition 2.7. The DDH assumption states that given g^u and g^v for $u = \mathrm{rand}(Z_q)$ and $v = \mathrm{rand}(Z_q)$ the value g^{uv} has the same distribution as any element $w = \mathrm{rand}(Z_q)$, i.e. g^{uv} is computationally indistinguishable from w when p and q are sufficiently large.

We assume that the agreed key in DH-KAP is not ephemeral and is different from session to session. Therefore it is not required to provide forward secrecy of this key. Moreover, in the case of challenge-response protocols parties are communicating in a very restricted time interval. Hence, according to these restrictions security of DH-KAP does not require DDH assumption.

But nevertheless, in AKAP2 we use ElGamal encryption where DDH assumption is required to provide user's anonymity.

DH-KAP is realized in SSL/TLS protocols included in the HTTPS protocol. DH-KAP is vulnerable to an active adversary attack known as a Man-in-the-Middle (MiM) attack (Callegati *et al.*, 2009). This attack is executed in the following way:

1. Alice randomly generates a secret number u in the interval $1 < u < p - 1$. She computes a session parameter $k_A = g^u \bmod p$ and sends k_A to the Bank.
   Then Adversary intercepts k_A and terminates message transmission to the Bank. Adversary impersonating the Bank against Alice randomly generates a secret number z in the interval $1 < z < p - 1$, computes a session parameter $k_{Z1} = g^z \bmod p$ and sends k_{Z1} to Alice. Analogously, Adversary impersonating Alice against the Bank randomly generates a secret number w in the interval $1 < w < p - 1$, computes a session parameter $k_{Z2} = g^w \bmod p$ and sends k_{Z2} to the Bank.
2. Alice presuming that message k_{Z1} is received from the Bank, computes the agreed secret key $k_{AZ} = (k_{Z1})^u \bmod p$.
   Adversary computes the same secret key $k_{ZA} = (k_A)^z \bmod p$.
3. The Bank presuming that k_{Z2} is received from Alice, randomly generates a secret number v in the interval $1 < v < p - 1$. It computes a session parameter $k_B = g^v \bmod p$ and sends k_B to Alice but this message is intercepted by Adversary. The Bank computes the agreed secret key $k_{BZ} = (k_{Z2})^v \bmod p$ as well.

Adversary computes the same secret key $k_{ZB} = (k_B)^w \bmod p$.

Evidently, $k_{AZ} = k_{ZA} = k_1$ and $k_{BZ} = k_{ZB} = k_2$ and hence, Adversary is able to decrypt any messages sent between Alice and the Bank. Moreover, Adversary can send to Alice his own messages encrypted with the key *k1* which can be decrypted by Alice and vice versa. Alice and the Bank do not suspect that Adversary impersonates both of them.

This attack can be prevented using AKAP.

2.2. *ElGamal Encryption*

Let m be a message to be encrypted by Alice and sent to the Bank. To obtain unambiguous encryption m must satisfy the following inequality $1 < m < q$. Encryption is performed using $\mathrm{SP} = (p, g)$ and the Bank's $\mathrm{PuK} = b$. Encryption is executed in the following way.

Alice chooses at random k, $1 < k < q$ and computes

$$e = mb^k \bmod p, \tag{2.12}$$

$$d = g^k \bmod p. \tag{2.13}$$

The ciphertext is $c = (e, d)$ which is sent to the Bank.

For decryption the Bank uses the same system parameters $\mathrm{SP} = (p, g)$ and its private key $\mathrm{PrK} = y$. Then

$$m = ed^{-y} \bmod p. \tag{2.14}$$

To be short we omit the validity proof of this identity. Further we use the following symbolic notation for encryption Enc() and decryption Dec() functions

$$c = (e, d) = \mathrm{Enc}(b, m), \quad m = \mathrm{Dec}(y, c). \tag{2.15}$$

This cipher we denote by the pair (Enc, Dec). The semantic security of ElGamal cipher is based on the following theorem (Tsiounis and Yung, 2006).

Theorem 2.8. *The semantic security of the ElGamal encryption is actually equivalent to the decision Diffie–Hellman* (DDH) *problem.*

2.3. *Schnorr Identification Protocol (S-Id)*

We assume that the Bank has Alice's $\mathrm{PuK} = a$ as her identity. Alice must prove that she knows her $\mathrm{PrK} = x$, corresponding to her $\mathrm{PuK} = a$ associated with her identity. $\mathrm{Prk}_A = x$ is called a witness and corresponding $\mathrm{Puk} = a = g^x \bmod p$ is called a statement. This protocol is initiated by Alice and has the following three communications.

1. Alice generates a random secret number $u = \mathrm{rand}(Z_q)$ and using $\mathrm{SP} = (p, g)$ computes **commitment** l in the following way

$$l = g^u \bmod p, \quad l \in G_p. \tag{2.16}$$

Alice sends l to the Bank.

2. The Bank generates a random **challenge** $h = \text{rand}(Z_q)$ and sends h to Alice.
3. After receiving h Alice computes her **response** r having her private key x together with previously generated secret number u:

$$r = u + xh \bmod q, \quad r \in Z_q. \tag{2.17}$$

After the third communication the Bank verifies if the following identity holds

$$g^r = la^h \bmod p. \tag{2.18}$$

If it is the case, the Bank trusts that Alice proved the knowledge that she possesses a private key PrK $= x$ corresponding to her public key PuK $= a$.

To be short we omit the validity proof of (2.18) identity.

In general, Alice is prover P proving that she knows a secret, namely her private key x, not revealing it and the Bank as a verifier V is either accepting this proof if (2.18) identity holds, or rejecting it otherwise. So SID is called **proof-of-knowledge**.

Proof-of-knowledge must satisfy three properties:

1. **Completeness:** if the statement is true, the honest verifier V, that is one following the protocol properly, will be convinced of this fact by an honest prover P.
2. **Soundness:** if the statement PuK $= a$ is false, no cheating prover P can convince the honest verifier V that he knows the secret, except with some small probability.
3. **Zero-knowledge:** if the statement PuK $= a$ is true, no verifier learns anything other than the fact that the statement is true. In other words, just knowing the statement but not the secret is sufficient to be convinced that the prover knows the secret. This is formalized by showing that every verifier has some *simulator* that, given only the statement to be proved but without any access to the prover, can produce a conversation that “looks like” an interaction between the honest prover and the verifier in question.

An interaction between P and V is performed when P knows PrK $= x$ and V knows PuK $= a$. This interaction we denote by P(x) and V(a) respectively generating a **conversation** $(l, h, r) \in G_q \times Z_q \times Z_q$. This conversation is an accepting conversation for a if (2.18) holds.

Proposition 2.9. *If the challenge space was small then Schnorr’s identification protocol is insecure.*

Comment 2.10. Let cardinality of challenge space Z_q be N, i.e. $|Z_q| = N$. Then, in its impersonation attempt, an adversary could use the simulator to prepare an accepting conversation (l, h, r), send l to V, and then hope that the challenge chosen by V is equal to its prepared challenge h. If so, the adversary could then respond with r, and so make V accept. Thus, Schnorr’s identification protocol is broken with advantage $1/N$; therefore, the challenge space Z_q must be super-poly in order to ensure security. In our case it is $N = 2^q$.

For further security considerations of our AKAP, the following notions should be introduced.

Let Gen be a key generation algorithm with input of certain system parameters SP and outputting private and public key pair (PrK, PuK). Then arbitrary identification protocol Id can be represented by the following triplet: Id = (Gen, P, V).

For example, in S-Id described above input to Gen is SP $= (p, g)$ and its output is a pair of private and public keys (x, a) according to (2.8), (2.9). Then symbolically S-Id = (Gen, P, V). Recall that according to DL assumption Gen is one-way-function (OWF).

The following theorem we present without proof (Boneh and Shoup, 2020).

Theorem 2.11. *Under the one-wayness assumption for Gen, and assuming $|Z_q| = N$ is super-poly, Schnorr's identification protocol is secure against eavesdropping attacks.*

In S-Id the one-wayness assumption for Gen means that the DL assumption is valid.

It is an open question as to whether Schnorr's identification protocol is secure against active attacks. So far there are no known effective, active attacks, but there is also no proof that rules out such an attack under the DL assumption.

Later we present a modification of S-Id, that is proven secure against active attacks under the DL assumption. Some introduction of the following notions is needed to provide this proof (Boneh and Shoup, 2020).

Definition 2.12. Let Id = (Gen, P, V) be an identification protocol. We say that Id is honest verifier zero knowledge, or HVZK for short, if there exists an efficient probabilistic algorithm Sim called a simulator such that for all possible outputs (PrK, PuK) of Gen, the output distribution of Sim on input PuK is identical to the distribution of a transcript of a conversation between P on input (PrK, PuK) and V on input (PuK).

The term "honest verifier" conveys the fact this simulation only works for conversations between P and the actual, "honest" verifier V, and not some arbitrary, "dishonest" verifier, such as may arise in an active attack on the identification protocol.

In our construction we propose mutual identification between Alice and the Bank. When the Bank is taking the role of Verifier we assume that the Bank is Honest Verifier due to the following assumptions. Firstly, we can assume that the Bank can prove its identity to the user more easily since the Bank has a public key certificate which can be recognizable by the user's browser. Secondly, during the identification protocol the Bank is using encrypt and sign procedures to confirm its identity.

Theorem 2.13. *Schnorr's identification protocol is* HVZK.

Proof. Simulator Sim in generating a conversation (l, h, r) and does not need to generate the messages of the conversation in a given order, as in a real conversation between P and V. Sim can generate the messages in reverse order. On input $\text{PuK}_A = a$, Sim computes $r = \text{rand}(Z_q)$, $h = \text{rand}(Z_q)$ and $l = g^r/a^h$. Then Sim outputs the conversation (l, h, r). We must prove that it is an acceptable conversation. It means that the output of Sim on input

$\mathrm{PuK}_A = a$ has the right distribution. The main observation is that in a real interaction, h and a are independent, and are uniformly distributed in Z_q. Moreover, for given h and a, the value l is uniquely determined by the equation $g^r = la^h \bmod p$ since according to (2.4) dexp() function is one-to-one. Then l has the same distribution as the output distribution of the simulator Sim. □

2.4. *Schnorr Signature Scheme (S-Sig)*

Let m be a message in Z_q to be signed by the Bank and sent to Alice. Parties are using cryptographic secure H-function to create and verify the signature on the message digest obtained by this function. For signature creation the Bank uses system parameters $SP = (p, g)$ and the Bank's $\mathrm{PrK} = y$. Let H-function be a mapping H: $G_q \times Z_q \to Z_q$.

The Bank chooses at random z, $1 < z < q$ and computes first component t of his signature:

$$t = g^z \bmod p. \tag{2.19}$$

The Bank computes H-value h and second component s of his signature:

$$h = H(m, t), \tag{2.20}$$

$$s = z + yh \bmod q. \tag{2.21}$$

The Bank's signature on h is $\sigma = (s, t)$. Then the Bank sends m and σ to Alice.

After receiving m and $\sigma = (s, t)$, Alice, according to (2.20), computes h and verifies if

$$g^s = tb^h \bmod p. \tag{2.22}$$

Symbolically we denote this verification function by

$$Ver(b, \sigma, h) \in \{True, False\}. \tag{2.23}$$

This function yields *True* if (2.22) is valid.

Referencing to Seurin (2012), Boneh and Shoup (2020) the following theorem can be formulated.

Theorem 2.14. *If H is modelled as a random oracle and Schnorr's identification scheme is secure against eavesdropping attacks, then Schnorr's signature scheme is also secure against eavesdropping attacks.*

3. AKAP Protocols

We present here two modifications of AKAP, namely AKAP1 and AKAP2 taking three communications between Alice and the Bank. AKAP1 is partially disclosing the user's anonymity by openly sending her PuK_A. In the case of AKAP1, the eavesdropping adversary can see that certain communications with the Bank are performed by the same person using the same PuK.

AKAP2 is providing user's anonymity without disclosing any user's personal information by realizing randomized encryption of the user's PuK during every session.

All parties including the adversary share the common information, namely system parameters SP $=$ (p,g) and the Bank's $\text{PuK}_B = b$. In addition, we also assume that the adversary may know public keys of users. So, in our model adversary knows two alternative sorts of information: either system parameters SP and the Bank's PuK_B or SP, PuK_B and users public keys, (e.g. PuK_A).

When Alice is a prover P then she uses protocol $\text{P}(x, a)$ with input parameters (x, a) and the Bank uses the verification protocol $\text{V}(a)$ respectively. We assume that the Bank is the trusted party and therefore it can prove its identity to users by its signature and PuK_B certificate realized in the lower level protocols such as SSL/TLS. But nevertheless, we supply AKAP1 and AKAP2 by extra identification of the Bank by signing its challenge sent to the user.

AKAP1

Alice and the Bank shares system parameters SP $= (p, g)$, $\text{PuK}_A = a$ and $\text{PuK}_B = b$.

1. Alice chooses a random number $u = \text{rand}(Z_q)$ and computes **commitment** l in the following way

$$l = g^u \bmod p, \quad l \in G_q. \tag{3.1}$$

   Alice sends (l, a) to the Bank.
2. After receiving (l, a) the Bank verifies if the user with his/her public key a is included in its customers' database and belongs to Alice. If it is ok, then the Bank seeks Alice to prove that she knows to correspond her private key x.

   The Bank chooses a random number $v = \text{rand}(Z_q)$ and computes **challenge** h in the following way

$$h = g^v \bmod p, \quad h \in G_q. \tag{3.2}$$

   The Bank signs challenge h using his $\text{PrK}_B = y$ by Schnorr signature scheme obtaining signature σ

$$\sigma = \text{Sig}(y, h) = (s, t). \tag{3.3}$$

   The Bank sends (h, σ) to Alice.
3. Alice verifies the validity of signature σ on challenge h with the Bank's $\text{PuK}_B = b$. If it is ok, Alice computes a secret session key k_{AB} according to Diffie–Hellman key exchange protocol

$$k_{AB} = h^u \bmod p. \tag{3.4}$$

Having her secrets u and x Alice computes the following response

$$r = u + xh + a \bmod (p-1). \tag{3.5}$$

Alice sends (r) to the Bank.

At this stage AKAP1 communications are finished.

After receiving r the Bank verifies if Alice knows her private key x corresponding to her public key a, which is registered in the Bank's database. The verification equation is the following:

$$g^r = la^h g^a \bmod p. \tag{3.6}$$

If the last equation is valid, then the identification procedure is passed successfully. The Bank computes the common session secret key k_{BA} according to Diffie–Hellman key exchange protocol

$$k_{BA} = l^v \bmod p. \tag{3.7}$$

Obviously at this moment parties agreed on their common session key $k = k_{AB} = k_{BA}$ and parties can continue communication using created secure channel with agreed secret key k.

The difference between convenient Schnorr identification protocol and AKAP1 is that there is an additional variable g^a in a verification equation (3.6). This will allow us to prove that S-Id is secure against an active adversary.

The second protocol is AKAP2 providing Alice's anonymity against an eavesdropping adversary. In this case, Alice's Puk $= a$ is encrypted and the adversary cannot distinguish if either the same person or two different persons are communicating with the Bank when he is eavesdropping and analysing any two different communications.

AKAP2

1. Alice chooses a random secret number $u = \text{rand}(Z_q)$ and computes commitment d_A

$$d_A = g^u \bmod p, \quad l \in G_q. \tag{3.8}$$

This commitment is also a partial key for DH-KAP.

To reduce computations Alice uses d_A to encrypt her $\text{PuK}_A = a$ by ElGamal encryption scheme to the recipient, the Bank, by computing

$$e_A = ab^u \bmod p, \tag{3.9}$$

The ciphertext is $c_A = (e_A, d_A)$. In our case d_A plays a triple role: commitment, partial key and second component of ciphertext c_A.

The ciphertext (c_A) is sent to the Bank.

2. After receiving c_A the Bank decrypts c_A using the Bank's $\text{PrK}_B = y$ and obtains Alice's $\text{PuK}_A = a$

$$a = e_A(d_A)^{-y} \bmod p. \tag{3.10}$$

The Bank verifies if the user with his/her public key is included in its customers' database and belongs to Alice. If Yes, then the Bank seeks Alice to prove that she knows her corresponding private key x. Otherwise, protocol is terminated.
The Bank chooses a random secret number $v = \text{rand}(Z_q)$ and computes **challenge** h

$$h = g^v \bmod p, \quad h \in G_q. \tag{3.11}$$

The Bank encrypts h by ElGamal encryption scheme to recipient Alice by choosing a random secret number $z = \text{rand}(Z_q)$ and computing ciphertext $c = (e, d)$

$$e = ha^z \bmod p, \qquad d = g^z \bmod p. \tag{3.12}$$

To confirm its identity the Bank signs component e by choosing a random secret number $w = \text{rand}(Z_q)$ and computing Schnorr signature $\sigma = (s, t)$ using its $\text{PrK}_B = y$

$$t = g^w \bmod p, \qquad s = w + ye \bmod q. \tag{3.13}$$

The Bank sends (c, σ) to Alice.
3. Alice verifies the validity of signature σ on value e with the Bank's $\text{PuK}_B = b$ with verification function $\text{Ver}(b, \sigma, e)$ in (2.23).
If it is the case, then Alice decrypts c using her $\text{PrK}_A = x$ thus obtaining **challenge** h

$$h = ed^{-x} \bmod p. \tag{3.14}$$

Alice computes the common secret session key k_{AB}

$$k_{AB} = h^u \bmod p. \tag{3.15}$$

Then Alice completes AKAP2 by computing her **response** r

$$r = u + xh + a \bmod q. \tag{3.16}$$

Alice sends (r) to the Bank.

At this stage AKAP2 communications are finished.

After receiving r the Bank verifies if Alice is the correct prover. He verifies if the following identity holds

$$g^r = d_A a^h g^a \bmod p. \tag{3.17}$$

If it is the case, then the Bank computes the common session secret key k_{BA}

$$k_{BA} = d_{A^v} \bmod p. \tag{3.18}$$

At this stage parties agreed on the common secret key $k = k_{BA} = k_{AB}$, performed mutual identification and can proceed communications by creating the secret channel.

4. AKAP Protocol Security Analysis

We show that AKAP1 is secure against active attack under the DL assumption by transforming the Schnorr identification protocol to Schnorr Sigma we denoted as AKAP1 protocol. A brief introduction to Sigma protocols is needed. Let X and A be finite sets and R is a binary relation $R \subseteq X \times A$ on $X \times A$. Then referencing to Boneh and Shoup (2020) we have the following definition.

Definition 4.1. Binary relation $R \subseteq X \times A$ is effective if X and A are efficiently recognizable finite sets. Elements of X are called witnesses for elements of A and elements of A are called statements.

Let $X = Z_q$ and $A = G_q$ then $R \subseteq Z_q \times G_q$ and $(x, a) \in R$, when $a = g^x \bmod p$. Then element $\mathrm{PrK}_A = x \in Z_q$ is a witness and element $\mathrm{PuK}_A = a \in G_q$ is a statement.

Lemma 4.1. *Binary relation defined by*

$$R = \{(x.a) \in Z_q \times G_q \mid g^x = a \bmod p\}, \tag{4.1}$$

is an effective binary relation.

Proof. Deciding that x is in Z_q is trivial. Let $a \in Z_p^*$ then to decide if $a \in G_q$ is required to verify the identity (2.1). If it is the case, then $a \in G_q$ since dexp() function is one-to-one and all elements in G_q (except 1) are generators. Then for every statement $a \in G_q$ there exists a unique witness x.

However, we have to find the witness x such that $g^x = a \bmod p$ corresponds to solving the DLP.

AKAP1 is realizing a conversation (l, h, r) where l is a commitment, h-challenge and r-response. □

Definition 4.2. A Sigma protocol for effective relation $R \subseteq Z_q \times G_q$ is a pair of (P, V) protocols satisfying the following conditions:

- P is an interactive protocol algorithm called the prover, which takes as input a witness-statement pair $(x, a) \in R$ and computes $\mathrm{P}(x, a)$.
- V is an interactive protocol algorithm called the verifier, which takes as input a statement $a \in G_q$, computes $\mathrm{V}(a)$ and outputs accept or reject.

P and V interactions are carried out in a similar way as they are presented in Section 2.

– To start the protocol, P computes commitment l and sends it to V;
– Upon receiving P's commitment l, V chooses a challenge h at random from a finite super-poly challenge space C, and sends h to P;
– Upon receiving V's challenge h, P computes a response r, and sends r to V;
– Upon receiving P's response r, V outputs either accept or reject, which must be computed strictly as a function of the statement a and the conversation (l, h, r). In particular, V does not make any random choices other than the selection of the challenge h. All other computations are completely deterministic.

To transform S-Id to AKAP1 we include an input a witness-statement pair $(x, a) \in R$ to compute $\mathrm{P}(x, a)$ for the prover.

We will prove that the AKAP1 protocol satisfies Sigma protocol's conditions. Notice that the prover P in S-IP takes as an input just the witness x, rather than the witness/statement pair (x, a), as formally required in the definition of any Sigma protocol. Therefore the conversation (l, h, r) is changed to the conversation (l, a, h, r).

Sigma protocol must satisfy the following conditions:

Completeness: $\mathrm{V}(a)$ always outputs accept for all $(x, a) \in R$, when $\mathrm{P}(x, a)$ and $\mathrm{V}(a)$ interact with each other.

Soundness: guarantees that no prover P that doesn't know the witness x can succeed in convincing the verifier V.

The following theorem is presented without a proof (Boneh and Shoup, 2020).

Theorem 4.3. S-SP *provides knowledge soundness.*

To proceed we must transform Definition 2.17 of HVZK to the definition of special HVZK (Boneh and Shoup, 2020).

Definition 4.4. Let (P, V) be a Sigma protocol for $R \subseteq X \times A$ with challenge space C. We say that (P, V) is special honest verifier zero knowledge, or special HVZK if there exists an efficient probabilistic algorithm Sim called a simulator that takes as input (a, h), and satisfies the following properties:

- for all inputs (a, h), algorithm Sim always outputs a pair (l, r) such that (l, h, r) is an accepting conversation for a;
- for all (x, a) in R, if anybody computes $h = \mathrm{rand}(Z_q)$ and $(l, r) = \mathrm{Sim}(a, h)$, then (l, h, r) has the same distribution as that of a transcript of a conversation between $\mathrm{P}(x, a)$ and $\mathrm{V}(a)$.

The differences between HVZK and special HVZK are the following: first, the simulator takes the challenge h as an additional input; second, it is required that the simulator produce an accepting conversation even when the statement a does not have a witness x. These two properties are the reason for the introduction of the notion of special HVZK.

Theorem 4.5. AKAP1 *is a special* HVZK.

Proof. Let input to the simulator Sim be (a, h). Then Sim computes $r = \text{rand}(Z_q)$, $a^h \bmod p$, $g^{a \bmod q} \bmod p$ and $l = g^r/(a^h g^{a \bmod q}) \bmod p$. Then computed conversation parameters (l, h, r) are accepting parameters since they have the same distribution as actual conversation of (P, V). □

The following theorem we present without proof is required (Boneh and Shoup, 2020).

Theorem 4.6. *Let* (P, V) *be a Sigma identification protocol for an effective relation R with super-poly challenge space. Assume that* (P, V) *provides knowledge soundness and is special* HVZK. *Furthermore, assume that the key generation algorithm Gen for R is one-way. Then Sigma identification protocol with parameters* (Gen, P, V) *is secure against active attacks.*

Referencing to our considerations above and Theorem 4.6. We can prove the following result.

Theorem 4.7. AKAP1 *is secure against active attacks.*

Proof. In Lemma 4.1 we proved that relation R in (4.1) is an effective binary relation. The challenge space C is super-poly since $|C| = 2^q$. Referencing to Theorem 4.3 AKAP1 provides knowledge soundness and referencing to Theorem 4.5 AKAP1 is HVZK. Under the DL assumption and conjectured one-wayness of dexp() function key generation algorithm Gen for R is one-way. □

Unfortunately, the similar result can not be proved for the AKAP2 protocol. The main reason is that it is not a Sigma protocol since the user's $\text{PuK}_A = a$ is encrypted during the first move of the protocol and can be decrypted only by a designated verifier V which is a Bank. In this case an active adversary has no access to the user's public key. Therefore, the theorems formulated above for Sigma protocols are not valid.

The security of AKAP2 we consider in the context of security of its components.

The user's anonymity protection is based on the security of the ElGamal encryption scheme. According to Theorem 2.12, the compromising of anonymity is equivalent to DDH problem solution. If SP has secure values then DDH assumption holds and anonymity is not compromised. In this case an eavesdropping adversary cannot distinguish any two conversations either they are originated from the same user or from the two different users.

The other characteristic of AKAP2 is that challenge h in AKAP1 is encrypted and signed. This encrypt and sign paradigm avoids the chosen-ciphertext attack and is CCA-secure encryption (Boneh and Shoup, 2020).

5. Discussions and Further Works

Two authenticated key agreement protocols AKAP1 and AKAP2 based on Diffie–Hellman KAP, Schnorr identification, Schnorr signature, and ElGamal encryption are presented.

It is proved that AKAP1 is secure against an active adversary under the discrete logarithm (DL) assumption.

To increase the security of AKAP1 the modified AKAP2 is proposed. Since this protocol does not satisfy sigma protocols conditions, the security proof of AKAP2 is restricted to only two components providing user's anonymity and CCA-secure encryption of verifiers (Bank's) challenge which is used also to agree on the common secret key.

Referencing to these results it is an intriguing idea to construct AKAP based on other similar assumptions instead of classical DL assumption, namely based on NP-complete problems. New conjectured one-way-function based on so called matrix power function (MPF) was proposed earlier in our papers (Sakalauskas *et al.*, 2008, 2017; Sakalauskas and Mihalkovich, 2014, 2017; Sakalauskas, 2018). MPF has some similarities with discrete exponent function. In Sakalauskas and Mihalkovich (2018) it is proved that inversion of MPF corresponds to a NP-complete problem. This proof was based on the result presented in Sakalauskas (2012). So far, the only key agreement protocol and asymmetric encryption scheme were realized using MPF but we think that the other protocols suitable for AKAP construction can be realized as well. Hence we expect that referencing to the results presented in this paper we could construct new AKAP based on MPF and prove its security using a similar methodology to the one presented in this paper.

References

Bellare, M., Rogaway, P. (1993). Random oracles are practical: a paradigm for designing efficient protocols. In: Ashby, V. (Ed.), *ACM CCS 93: 1st Conference on Computer and Communications Security*, pp. 62–73.

Bleichenbacher, D. (1996). Generating ElGamal signatures without knowing the secret key. In: *Advances in Cryptology EUROCRYPT'96*, Zaragoza, Spain, Lecture Notes in Computer Science, Vol. 1070. pp. 10–18.

Boneh, D. (1998). The decision Diffie–Hellman problem. In: *Proceedings of the Third Algorithmic Number Theory Symposium*, *Lecture Notes in Computer Science*, Vol. 1423, pp. 48–63.

Boneh, D., Shoup, V. (2020). *A Graduate Course in Applied Cryptography*. Version 0.5. https://toc.cryptobook.us.

Callegati, F., Cerroni, W., Ramilli, M. (2009). Man-in-the-middle attack to the HTTPS protocol. *IEEE Security & Privacy Magazine*, 7, 78–81.

ElGamal, T. (1985). A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4), 469–472.

Just, M. (2011). Challenge-response identification. In: van Tilborg, H.C.A., Jajodia, S. (Eds.), *Encyclopedia of Cryptography and Security*. Springer, Boston, MA.

Mei, Q., Zhao, Y., Xiong, H. (2019). A new provably secure certificateless signature with revocation in the standard model. *Informatica*, 30(4), 711–728.

Muleravicius, J., Timofejeva, I., Mihalkovich, A., Sakalauskas, E. (2019). Security, trustworthiness and effectivity analysis of an offline E-cash system with observers. *Informatica*, 30(2), 327–348.

Neven, G., Smart, N., Warinschi, B. (2009). Hash function requirements for Schnorr signatures. *Journal of Mathematical Cryptology*, 3(1), 69–87.

Pointcheval, D., Stern, J. (1996). Security proofs for signature schemes. In: Maurer, U.M. (Ed.), *Advances in Cryptology – EUROCRYPT'96*, *Lecture Notes in Computer Science*, Vol. 1070. pp. 387–398.

Pointcheval, D., Stern, J. (2000). Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3), 361–396.

Sakalauskas, E. (2012). The multivariate quadratic power problem over Zn is NP-complete. *Information Technology and Control*, 41(1), 33–39.

Sakalauskas, E. (2018). Enhanced matrix power function for cryptographic primitive construction. *Symmetry*, 10(2), 43.

Sakalauskas, E., Mihalkovich, A. (2014). New asymmetric cipher of non-commuting cryptography class based on matrix power function. *Informatica*, 25(2), 283–298.

Sakalauskas, E., Mihalkovich, A. (2017). Improved asymmetric cipher based on matrix power function resistant to linear algebra attack. *Informatica*, 28(3), 517–524.

Sakalauskas, E., Mihalkovich, A. (2018). MPF problem over modified medial semigroup Is NP-complete. *Symmetry*, 10(11), 571.

Sakalauskas, E., Listopadskis, N., Tvarijonas, P. (2008). Key Agreement Protocol (KAP) Based on Matrix Power Function. *Information Science and Computing, Book 4 Advanced Studies in Software and Knowledge Engineering*. FOI ITHEA, pp. 92–96.

Sakalauskas, E., Mihalkovich, A., Venčkauskas, A. (2017). Improved asymmetric cipher based on matrix power function with provable security. *Symmetry*, 9(1), 9.

Schnorr, C.P. (1990). Efficient identification and signatures for smart cards. In: Brassard, G. (Ed.), *Advances in Cryptology – CRYPTO'89*, *Lecture Notes in Computer Science*, Vol. 435. pp. 239–252.

Schnorr, C.P. (1991). Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3), 161–174.

Seurin, Y. (2012). On the exact security of Schnorr-type signatures in the random oracle model. In: Pointcheval, D., Johansson, T. (Eds.), *Advances in Cryptology – EUROCRYPT 2012*, *Lecture Notes in Computer Science*, Vol. 7237. pp. 554–571.

Shor, P.W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 1997(26), 1484–1509.

Tseng, Y.-M., Tsai, T.-T., Wu, J.-D., Huang, S.-S. (2019). Efficient certificate-based signature with short key and signature sizes from lattices. *Informatica*, 30(3), 595–612.

Tsiounis, Y., Yung, M. (2006). On the security of ElGamal based encryption. In: *Lecture Notes in Computer Science*, Vol. 1431. Springer, Berlin, Heidelberg, pp. 117–134.

A. Kilciauskas, MsD in informatics, distant learning information technologies, in 2017. Work expertise in corporate security, cryptography, blockchain technology.

G. Butkus, MsD in informatics, in 1992. Expertise in computer networks and security. CompTIA Security+ Certified Professional. Cisco Certified Network Associate (CCNA) Routing and Switching. Cisco Certified Network Professional (CCNP) Routing and Switching. Cisco Certificated Academy Instructor (CCAI). Scientific interests are cryptography and blockchain technology.

E. Sakalauskas is a professor at Department of Applied Mathematics, Kaunas University of Technology. His research interests are focused in cryptography. The main research results in this field were published in over 20 papers.