FORMAL SECURITY PROOF FOR A SCHEME ON A TOPOLOGICAL NETWORK

ROBERTO CIVINO*
Department of Information Engineering, Computer Science, and Mathematics
University of L’Aquila
Via Vetoio, 67100 L’Aquila (AQ), Italy

RICCARDO LONGO
Department of Mathematics
University of Trento
Via Sommarive 14, 38123 Povo (TN), Italy

(Communicated by Jens Zumbragel)

ABSTRACT. Key assignment and key maintenance in encrypted networks of resource-limited devices may be a challenging task, due to the permanent need of replacing out-of-service devices with new ones and to the consequent need of updating the key information. Recently, Aragona et al. proposed a new cryptographic scheme, ECTAKS, which provides a solution to this design problem by means of a Diffie-Hellman-like key establishment protocol based on elliptic curves and on a prime field. Even if the authors proved some results related to the security of the scheme, the latter still lacks a formal security analysis. In this paper, we address this issue by providing a security proof for ECTAKS in the setting of computational security, assuming that no adversary can solve the underlying discrete logarithm problems with non-negligible success probability.

1. Introduction

The Elliptic Curve based Topological Authenticated Key Scheme (ECTAKS), recently developed by Aragona et al. [1], represents a solution to the problem of key assignment in resource-constrained encrypted networks comprised of sensors with a limited lifetime, where there is the persistent need of updating the key information once an off-duty node needs replacing. In ECTAKS, being the scheme designed such that the network topology is more relevant than the node identity, the cost of network updates is highly reduced. The parties involved in the communication, following an asymmetrical Diffie-Hellman-like key exchange, may agree on a shared secret, called the Elliptic Curve based Topological Authenticated Key ECTAK, which can be used to produce encrypted messages and digital signatures. Both point-to-point and point-to-multipoint secure sessions are supported by the scheme. At the time of writing though, ECTAKS lacked a formal proof of security. In [1], the authors proved that the linear-algebra conditions that define the parameters of the

2020 Mathematics Subject Classification: 94A60, 94A62, 94C15, 68P25.

Key words and phrases: Key establishment protocols, elliptic curve cryptography, discrete logarithm problem, formal security, key recovery, key indistinguishability.

The authors are members of INdAM-GNSAGA (Italy). This work was partially supported by the Centre of EXcellence on Connected, Geo-Localized and Cybersecure Vehicles (EX-Emerge), funded by Italian Government under CIPE resolution n. 70/2017 (Aug. 7, 2017).

* Corresponding author.
scheme cannot be exploited by an attacker aiming at recovering the private information of a target node, even if the attacker is able to solve an instance of the underlying discrete logarithm problem on elliptic curves. However, a formal security reduction is reported to be an open problem. In this paper, we address this issue by proving a formal security proof for ECTAKS. On the basis of the Computational and Decisional Diffie-Hellman assumptions [2, 3], we develop a security model for ECTAKS and we prove that the scheme is secure against key-recovery and key-indistinguishability attacks, respectively.

The paper is organized as follows: in Section 2 we provide a detailed description of ECTAKS as in [1] and introduce the security assumptions on the discrete logarithm problems over elliptic curves. In Section 3 we introduce the security model for ECTAKS and prove two main results, i.e. Theorem 3.3 and Theorem 3.6, where we respectively show the security of the scheme against key-recovery and key-indistinguishability attacks. Finally, in Section 4 we draw our conclusions.

2. Preliminaries on ECTAKS

In this section we introduce the scheme ECTAKS and the security assumptions on the discrete logarithm problems on which our formal security proofs for ECTAKS are based. The description of the scheme is provided below; for more detailed information the reader is referred to [1]. Previous versions of ECTAKS may be found in [4, 5, 6], where the construction is also considered in relation to other groups for which Diffie-Hellman problems are hard. It is important to point out here that, even if the presented scheme is based on elliptic curves, the security proofs do not rely on particular properties of elliptic curves and so can be easily adapted to arbitrary groups suitable for discrete logarithm problems.

2.1. ECTAKS. Let $p$ be a prime number, and let $\mathbb{F}_p$ be the finite field with $p$ elements. Let $q > 3$ be the power of a prime number, and let $C$ be an elliptic curve over $\mathbb{F}_q$ [7] such that there exists $G \in C$ that generates a subgroup of $C$ of order $p$. The element $G$ is called the base point.

The following definitions are useful: given a vector $k = (\alpha_1, \alpha_2) \in (\mathbb{F}_p)^2$ and the scalars $\beta_1, \beta_2 \in \mathbb{F}_p$ let, as in [1],

$$kG \overset{\text{def}}{=} (\alpha_1G, \alpha_2G) \in C \quad \text{and} \quad k \cdot (\beta_1G, \beta_2G) \overset{\text{def}}{=} \alpha_1\beta_1G + \alpha_2\beta_2G \in C.$$

In the protocol ECTAKS, the sensor network is modeled by means of a graph where each user (i.e. sensor) represents a node, and where two users are entitled to communicate if and only if they are connected in the corresponding graph. More precisely, if an edge from node $i$ to node $j$ exists in the network, then user $i$ is allowed to start a communication session with user $j$. Each node possesses a set of parameters, called Local Configuration Data (LCD), assigned by an external and trusted Certification Authority (CA). The LCD is composed by two secret components, that remain unchanged once generated, and a public component, which is updated every time a new node joins the dynamic network. The formal description of what has been previously mentioned is provided below.

2.1.1. Parameter definition. A directed graph is a pair of sets $(V, E)$, where $V \neq \emptyset$ and $E \subseteq V \times V$. The elements of $E$ are called arrows and, for each arrow $e = (i, j) \in E$, the tail of the arrow is denoted as $t(e) = i$ and the head as $h(e) = j$. For sake of compactness, the arrow $(i, j)$ is sometimes denoted by writing ‘$i \rightarrow j$’.
If $i, j \in V$, then $i$ and $j$ are said to be strongly adjacent in $(V, E)$ if $(i, j) \in E$ and $(j, i) \in E$.

Let $N$ be a positive integer such that $p > N$ and let $V = \{1, 2, \ldots, N\}$. The Authenticated Network Topology is a loop-free symmetric directed graph $\text{ANT} = (V, E)$, i.e., a graph with no cycles of length 1, where $E \subseteq V \times V$ and if $(i, j) \in E$, then $(j, i) \in E$. For each $1 \leq i \leq N$, $\text{ANT}_i = (V_i, E_i)$ is the (non-symmetric and cycle-free) directed subgraph of $\text{ANT}$ such that

$$E_i \overset{\text{def}}{=} \{ e \in E \mid t(e) = i \} \quad \text{and} \quad V_i \overset{\text{def}}{=} \{i\} \cup \{h(e) \mid e \in E_i\}.$$ 

In the point of view of the application, $\text{ANT}_i$ is the subgraph of the users which user $i$ is entitled to communicate to. An example of network topology network is displayed in Fig. 1.

For each node $i \in V$, its assigned Local Configuration Data $\text{LCD}_i = (S_i, P_i)$ is such that

$$S_i \overset{\text{def}}{=} \{k_i, t_i\} \quad \text{and} \quad P_i \overset{\text{def}}{=} \{m_{i \rightarrow j} G\}_{j \in V_i \setminus \{i\}},$$

where $k_i \in (\mathbb{F}_p)^2 \setminus \{0\}$ is called the local key component of the node $i$, $t_i \in (\mathbb{F}_p)^2 \setminus \{0\}$ is called the transmitted key component of the node $i$, and $m_{i \rightarrow j} G \in C^2 \setminus \{0\}$ is called the topology vector of the arrow $(i, j)$. The component $S_i$ represents the private information assigned to node $i$, whereas $P_i$ represents its public information. Their role will be discussed in more detail in the subsequent Section 2.1.2.

2.1.2. Parameter assignment and shared secrets. Once the CA has chosen an arbitrary root node for each connected component of the graph, it assigns the parameters to each node in a sequential way. Starting from the parameters assigned to the root node, the CA computes the parameters for the other nodes of the graph according to certain constraints which will allow each pair of strongly adjacent nodes to compute a shared secret, the so-called Elliptic Curve Topology Authenticated Key (ECTAK).

Let node $i$ be the first root node chosen by the CA. The parameters $k_i$ and $t_i$ are generated randomly from $(\mathbb{F}_p)^2 \setminus \{0\}$ and assigned to the secret component $S_i = \{k_i, t_i\}$ of node $i$. For each node $j$ strongly adjacent to node $i$, two cases need to be distinguished:
S\textsubscript{j} is not defined: in this case the parameter \( m_{i\rightarrow j} \) is generated randomly by the CA, provided that \( k_i \cdot m_{i\rightarrow j} \neq 0 \). The corresponding topology vector \( m_{i\rightarrow j}G \) is appended to \( P_i \). The CA chooses the parameters for node \( j \) proceeding as follows:

- \( k_j \) is randomly selected from the solutions of the linear equation

\[ k_i \cdot m_{i\rightarrow j} = k_j \cdot t_i; \]

- \( m_{j\rightarrow i} \) is randomly selected, provided that \( k_j \cdot m_{j\rightarrow i} \neq 0 \);

the corresponding topology vector \( m_{j\rightarrow i}G \) is appended to \( P_j \);

- \( t_j \) is randomly selected from the solutions of the linear equation

\[ k_j \cdot m_{j\rightarrow i} = k_i \cdot t_j. \]

S\textsubscript{j} is already defined: in this case the topology vectors related to the arrows \((i, j)\) and \((j, i)\) are chosen by the CA as follows:

- \( m_{i\rightarrow j} \) is randomly selected from the solutions of the linear equation

\[ k_i \cdot m_{i\rightarrow j} = k_j \cdot t_i; \]

- \( m_{j\rightarrow i} \) is randomly selected from the solutions of the linear equation

\[ k_j \cdot m_{j\rightarrow i} = k_i \cdot t_j. \]

The parameter assignment is completed when the CA has assigned secret and public components to each node of the graph.

Assuming that node \( i \) and \( j \) are strongly adjacent and that node \( i \) wants to start a session with node \( j \), they can agree on the ephemeral shared secret \( ECTAK_{i\rightarrow j} \) performing the following operations:

- node \( i \) generates a random non-zero element \( \alpha \in \mathbb{F}_p \);
- node \( i \) sends \( C_{i\rightarrow j} \overset{\text{def}}{=} \alpha t_iG \) to node \( j \);
- node \( i \) computes \( ECTAK_{i\rightarrow j} \overset{\text{def}}{=} \alpha k_i \cdot (m_{i\rightarrow j}G) \).

Also node \( j \) can compute

\[ k_j \cdot (\alpha t_iG) = (k_j \cdot \alpha t_i)G = (\alpha k_i \cdot m_{i\rightarrow j})G = \alpha k_i \cdot (m_{i\rightarrow j}G) = ECTAK_{i\rightarrow j}, \]

where the second equality is obtained from Equation (1). Consequently, node \( i \) and node \( j \) have shared the non-zero secret \( ECTAK_{i\rightarrow j} \in C \). Similarly, when node \( j \) wants to start a session, they can agree with node \( i \) on the shared secret

\[ ECTAK_{j\rightarrow i} \overset{\text{def}}{=} \alpha k_j \cdot (m_{j\rightarrow i}G) = k_j \cdot (\alpha t_jG), \]

where \( \alpha \) is again an ephemeral randomly chosen non-zero element in \( \mathbb{F}_p \), this time generated by node \( j \), and the second equality is derived from Equation (2).

Aiming at proving that the previously shown protocol is computationally secure, let us define the \textit{Computational} and \textit{Decisional Diffie-Hellman assumptions} \cite{2, 3} on which the security proofs are based. Next, an algorithm is said to be \textit{efficient} if it runs in (expected) polynomial time in the size of the input, possibly using a random source.
2.2. Assumptions. Let $\alpha, \beta \in \mathbb{F}_p$ be chosen uniformly at random and recall that $G \in C$ is the base point of order $p$. The Computational Diffie-Hellman (CDH) problem consists in constructing an algorithm

$$B_C(G, A = \alpha G, B = \beta G) \rightarrow C$$

that efficiently computes the point $\alpha \beta G \in C$. The correctness probability of $B_C$ is defined as

$$\text{Corr}_{B_C} \overset{\text{def}}{=} \Pr \left[ B_C(G, A, B) = \alpha \beta G \right],$$

where the probability is taken over the random choice of the base point $G$, of $\alpha, \beta$ in $\mathbb{F}_p$, and the random bits possibly consumed by $B_C$ to compute the response.

The following security assumption is derived.

**Definition 2.1 (CDH Assumption)**. No probabilistic polynomial-time algorithm $B_C$ solves the CDH problem with more than negligible correctness probability $\text{Corr}_{B_C}$.

Similarly, let $\alpha, \beta, \gamma \in \mathbb{F}_p$ be chosen at random. The Decisional Diffie-Hellman (DDH) problem consists in constructing an efficient algorithm

$$B_D(G, A = \alpha G, B = \beta G, T) \rightarrow \{0, 1\}$$

to distinguish between the tuples $(G, A, B, \alpha \beta G)$ and $(G, A, B, \gamma G)$, outputting respectively 1 and 0. The advantage of $B_D$ is defined as

$$\text{Adv}_{B_D} \overset{\text{def}}{=} \left| \Pr \left[ B_D(G, A, B, \alpha \beta G) = 1 \right] - \Pr \left[ B_D(G, A, B, \gamma G) = 1 \right] \right|,$$

where the probabilities are taken over the random choice of the base point $G$, of $\alpha, \beta, \gamma$ in $\mathbb{F}_p$, and the random bits possibly consumed by $B_D$ to compute the response.

The following security assumption is derived.

**Definition 2.2 (DDH Assumption)**. No probabilistic polynomial-time algorithm $B_D$ solves the DDH problem with more than negligible advantage $\text{Adv}_{B_D}$.

3. Security proofs

Our main results of this paper, i.e. Theorems 3.3 and 3.6, which are the formal proofs of security of the protocol described in Section 2, are shown in this section. They are both based on a preliminary result, namely Lemma 3.7, which is presented in the body of the section and anticipates the proofs of Theorems 3.3 and 3.6.

In our security model, defined below, the aim of the attacker is to either retrieve the ephemeral secret ECTAK or to distinguish a properly generated secret from a random non-trivial point of the curve. The proofs are based, without loss of generality, on the minimal setting of a simple network with three users. Considering an adversary that wants to attack the key exchange between two nodes, indeed, they gain more information when controlling a node strongly adjacent to both target nodes. Moreover, the key generation process limits the information the attacker can gain if they succeed in controlling extra nodes, therefore the model is general enough for a security analysis.
3.1. Security model. Let us assume that the Certification Authority sets up an ECTAKS instance on a network with topology, shown in Figure 2, defined by a graph \((V,E)\) where
\[
V = \{1,2,3\} \quad \text{and} \quad E = \{(1,2),(2,1),(1,3),(3,1),(2,3),(3,2)\}.
\]

In the attack scenario, an adversary \(A\) seizes the control of node 3, therefore gaining access to \(S_3 = (k_3,t_3)\) and \(P_3 = \{m_{3\to 1}G,m_{3\to 2}G\}\), and tries to attack the key exchange from node 1 to node 2.

3.2. Security with respect to key recovery. Our first result proves that the protocol is secure in terms of key-recovery attacks. The security game is formally defined as follows, where, for sake of compactness, ECTAK\(_{i\to j}\) is denoted by \(E_{i\to j}\).

**Definition 3.1** (Security game). The security game for a key-recovery attack, between the challenger \(C\) and the adversary \(A\), proceeds as follows:

- **Setup:** \(C\) acts as the CA and sets up the ECTAKS protocol on the network of the security model of Section 3.1, and provides \(A\) with the key parameters of node 3, i.e. LCD\(_3\) = \(S_3,P_3\);

- **Phase 1:** \(A\) may request multiple session-key pairs \((C^n_{1\to 2},E^n_{1\to 2})\) generated by node 1 to communicate with node 2;

- **Challenge:** \(C\) computes a challenge session-key pair \((C^n_{1\to 2},E^n_{1\to 2})\), with the trivial constraint \(C^n_{1\to 2} \neq C^n_{1\to 2}\) for each \(i\), and sends \(C^n_{1\to 2}\) to \(A\);

- **Phase 2:** Phase 1 may be repeated, provided that the aforementioned constraint is respected;

- **Guess:** \(A\) outputs a guess \(E'_{1\to 2}\) for the private session-key corresponding to \(C^n_{1\to 2}\). The adversary wins if \(E'_{1\to 2} = E^n_{1\to 2}\), i.e. if the output corresponds to the correct key.

**Definition 3.2** (Security against key recovery). An ECTAKS protocol with security parameter \(\xi\) is secure against key recovery if, for all probabilistic polynomial-time adversaries \(A\) that play the game of Definition 3.1, there exists a negligible function \(\psi\) such that
\[
\Pr [A \text{ wins}] = \Pr [E'_{1\to 2} = E^n_{1\to 2}] \leq \psi(\xi).
\]

The following theorem proves that the protocol is secure against key-recovery attacks under the CDH assumption of Definition 2.1, in the security game defined above.

**Theorem 3.3.** If there exists an adversary that wins the security game of Definition 3.1 with non-negligible probability \(\varepsilon\), then a simulator that solves the CDH problem with non-negligible correctness probability \(\varepsilon\) can be constructed.
3.3. Security with respect to key indistinguishability. Next, we prove instead that the protocol is secure with respect to key indistinguishability under the DDH assumption of Definition 2.2, in the security game defined below.

**Definition 3.4 (Security game).** The security game for a key-indistinguishability attack, between the challenger \( C \) and the adversary \( A \), proceeds as follows:

- **Setup:** \( C \) acts as the CA and sets up the ECTAKS protocol on the network of the security model of Section 3.1, and provides \( A \) with the key parameters of node 3, i.e. \( \text{LCD}_3 = (S_3, P_3) \);
- **Phase 1:** \( A \) may request multiple session-key pairs \( (C_{i \rightarrow 2}^{(i)}, E_{i \rightarrow 2}^{(i)}) \) generated by node 1 to communicate with node 2;
- **Challenge:** \( C \) computes a challenge session-key pair \( (C_{1 \rightarrow 2}^*, E_{1 \rightarrow 2}^*) \), with the trivial constraint \( C_{1 \rightarrow 2}^* \neq C_{1 \rightarrow 2}^{(i)} \) for each \( i \). Then \( C \) flips a random coin \( b \in \{0, 1\} \) and delivers the challenge \( (C_{1 \rightarrow 2}^*, E_b) \) to \( A \), where \( E_1 = E_{1 \rightarrow 2}^* \) and \( E_0 \) is a random element of the subgroup of the elliptic curve \( \mathcal{C} \) generated by the base point \( G \), except the point at infinity \( \mathcal{O} \);
- **Phase 2:** Phase 1 may be repeated, provided that the aforementioned constraint is respected;
- **Guess:** \( A \) outputs a guess \( b' \) of the coin toss \( b \), and wins if \( b' = b \).

**Definition 3.5 (Security against key indistinguishability).** An ECTAKS protocol with security parameter \( \xi \) achieves key indistinguishability if, for all probabilistic polynomial-time adversaries \( A \) that play the game of Definition 3.4, there exists a negligible function \( \psi \) such that:

\[
\Pr[A \text{ wins}] = \Pr[b' = b] \leq \frac{1}{2} + \psi(\xi).
\]

**Theorem 3.6.** If there exists an adversary that wins the security game of Definition 3.4 with non-negligible advantage \( \varepsilon \), then a simulator that solves the DDH Problem with non-negligible advantage \( \varepsilon \) can be constructed.

Theorems 3.3 and 3.6, showing the formal security of ECTAKS, are proved in the next section.

3.4. Proofs. The proofs of Theorems 3.3 and 3.6 are based on simulators which can set up the protocol acting as the CA. In both scenarios the same approach is used, i.e. the one of the following lemma.

**Lemma 3.7.** Given as input a Diffie-Hellman-like challenge tuple \( (G, A = \alpha G, B = \beta G) \), there exists a simulator that can set up the ECTAKS protocol as described in Section 3.1, provide an adversary \( A \) with the key parameters given to node 3, i.e. \( \text{LCD}_3 = (S_3, P_3) \), and output a challenge public session-key \( C_{1 \rightarrow 2}^* \) linked to \( A \) and \( \alpha \beta G \).

**Proof.** In order to construct the simulator, our strategy is to embed the value \( A \) of the Diffie-Hellman challenge into the public session key, and the value \( B \) into the public information of the second node \( P_2 \), so that the corresponding private session-key becomes linked to the value \( \alpha \beta G \). In particular, we implicitly set the value \( \alpha^* \), that node 1 should have chosen to construct \( C_{1 \rightarrow 2}^* = \alpha^* t_1 G \), to be equal to the parameter \( \alpha \) of the DH challenge, so that

\[
C_{1 \rightarrow 2}^* = t_1 A.
\]
Then, choosing uniformly at random \(k_1, t_1, m'_{1 \rightarrow 2} \in \mathbb{F}_p^2 \setminus \{(0,0)\}\), provided that \(k_1 \cdot m'_{1 \rightarrow 2} \neq 0\), and implicitly setting \(m_{1 \rightarrow 2} = m'_{1 \rightarrow 2} \beta\), we have
\[
m_{1 \rightarrow 2}G = m'_{1 \rightarrow 2}B.
\]
Note that if \(\beta \neq 0\), then \(k_1 \cdot m_{1 \rightarrow 2} \neq 0\). Moreover if \(\beta = 0\), then the DH challenge becomes trivial, so our reduction is unaffected. Note also that we can restrict our choice of parameters assuming \(k_{1,2}, t_{1,1} \neq 0\), since this happens with overwhelming probability.

We can continue the setup generating the values for the node 2. Since \(k_2\) should be a solution of the equation \(k_1 \cdot m_{1 \rightarrow 2} = k_2 \cdot t_1\), we can freely choose \(k_{2,2} \in \mathbb{F}_p \setminus \{0\}\) and implicitly set \(k_{2,1} = t_{1,1}^{-1}(k_1 \cdot m'_{1 \rightarrow 2} \beta - k_{2,2} t_{1,2})\), so that we can compute
\[
k_{2,1}G = t_{1,1}^{-1}(k_1 \cdot m'_{1 \rightarrow 2})B - t_{1,1}^{-1}k_{2,2} t_{1,2} G.
\]
We can successively generate \(m_{2 \rightarrow 1} \in \mathbb{F}_p^2 \setminus \{(0,0)\}\) uniformly at random and check that \(k_2 \cdot m_{2 \rightarrow 1} \neq 0\), by verifying that \(m_{2 \rightarrow 1,1}(k_2 G) + k_{2,2} m_{2 \rightarrow 1,2} G \neq O\), where \(O\) is the point at infinity of the curve \(\mathcal{C}\). The parameter \(t_2\) should be a solution of the equation \(k_2 \cdot m_{2 \rightarrow 1} = k_1 \cdot t_2\), so we freely choose \(t_{2,1} \in \mathbb{F}_p \setminus \{0\}\) and implicitly set \(t_{2,2} = k_{1,2}^{-1}(k_2 \cdot m_{2 \rightarrow 1}) - k_{1,1} t_{2,1}\), so that we can compute
\[
t_{2,2}G = k_{1,2}^{-1} m_{2 \rightarrow 1,1}(k_2 G) + k_{1,2}^{-1} k_{2,2} m_{2 \rightarrow 1,2} G - k_{1,2}^{-1} k_{1,1} t_{2,1} G.
\]
Note that we can generate the values for node 3 in the same way they would be generated by the CA, i.e. choosing randomly \(m_{1 \rightarrow 3} \in \mathbb{F}_p^2 \setminus \{(0,0)\}\) in order to have \(k_1 \cdot m_{1 \rightarrow 3} \neq 0\), choosing \(k_3\) randomly from the solutions of \(k_1 \cdot m_{1 \rightarrow 3} = k_3 \cdot t_1\) (note that again we can restrict our choice to have \(k_{3,1} \neq 0\) since this happens with overwhelming probability), then choosing \(m_{3 \rightarrow 1} \in \mathbb{F}_p^2 \setminus \{(0,0)\}\) such that \(k_3 \cdot m_{3 \rightarrow 1} \neq 0\), and finally choosing \(t_3\) randomly from the solutions of \(k_3 \cdot m_{3 \rightarrow 1} = k_1 \cdot t_3\). Finally, we need to generate the values \(m_{3 \rightarrow 2}\) and \(m_{2 \rightarrow 3}\), that should satisfy the equations \(k_3 \cdot m_{3 \rightarrow 2} = k_2 \cdot t_3\) and \(k_2 \cdot m_{2 \rightarrow 3} = k_3 \cdot t_2\) respectively. We start by choosing \(m_{3 \rightarrow 2,2} \in \mathbb{F}_p \setminus \{0\}\) uniformly at random, and then implicitly setting \(m_{3 \rightarrow 2,1} = k_{3,1}^{-1}(k_2 \cdot t_3) - k_{3,2} m_{3 \rightarrow 2,2}\), so that we can compute
\[
m_{3 \rightarrow 2,1}G = k_{3,1}^{-1} t_{3,1}(k_2 G) + k_{3,1}^{-1} k_{2,2} t_{3,2} G - k_{3,1}^{-1} k_{3,2} m_{3 \rightarrow 2,2} G.
\]
Ultimately, we choose \(m_{2 \rightarrow 3,1} \in \mathbb{F}_p \setminus \{0\}\) uniformly at random, then we implicitly set \(m_{2 \rightarrow 3,2} = k_{2,1}^{-1}(k_3 \cdot t_2) - k_{2,1} m_{2 \rightarrow 3,1}\), so that we can compute
\[
m_{2 \rightarrow 3,2}G = k_{2,2}^{-1} k_{3,1} t_{2,1} G + k_{2,2}^{-1} k_{3,2} (t_{2,2} G) - k_{2,2}^{-1} m_{2 \rightarrow 3,1}(k_2 G).
\]
Now that every value required in the setup of ECTAKS as per Section 3.1 has been (implicitly or explicitly) generated, \(k_3\) and \(t_3\), that are explicitly generated, can be passed to the adversary \(\mathcal{A}\). Moreover, also the values \(m_{i \rightarrow j}\) can be delivered to \(\mathcal{A}\), for all \(i, j\), since using Equations (6) to (10) we can compute the values whose coefficients were only implicitly generated.

To conclude the proof, note that the private key corresponding to the challenge public session-key defined in Equation (5) satisfies
\[
E'_{1 \rightarrow 2} = k_2 \cdot (\alpha^* t_1 G) = (k_1 \cdot m_{1 \rightarrow 2}) \alpha^* G = (k_1 \cdot m'_{1 \rightarrow 2})(\alpha \beta G),
\]
thus we have successfully linked the challenge public session-key to the target value of the Diffie-Hellman challenge \(\alpha \beta G\). \(\square\)
Remark 1. Note that, since in the Diffie-Hellman challenge \( \alpha \) and \( \beta \) are generated uniformly at random and are not null in non-trivial challenges, then any value of the form \( x\alpha \) or \( x\beta \) is uniformly distributed when \( x \in \mathbb{F}_p \) is chosen uniformly at random. Moreover, suppose we fix \( y = (y_1, y_2) \in \mathbb{F}_p^2 \backslash \{(0,0)\} \). Then, for every \( z \in \mathbb{F}_p \), if \((x_1, x_2), (\bar{x}_1, \bar{x}_2)\) are any two among the \( p \) non-zero distinct solutions in \( \mathbb{F}_p^2 \) of the equation \( x \cdot y = z \), we have \( x_1 \neq \bar{x}_1 \). This also implies that, provided that both \( z \) and \( y_2 \) are non-zero, selecting uniformly at random a solution \((x_1, x_2) \in \mathbb{F}_p^2 \backslash \{(0,0)\}\) is equivalent to selecting uniformly at random \( x_1 \) and computing \( x_2 = y_2^{-1}(z - x_1 y_1) \).

This means that the simulation shown in the proof of Lemma 3.7 generates the parameters exactly as in a standard execution of the protocol, with the sole exception that some values are restricted to be not null. As already noted, this happens with overwhelming probability also in a standard execution, therefore the procedure described in the proof perfectly simulates the protocol.

We are now ready to prove our main results.

Proof of Theorem 3.3. Let \( \mathcal{A} \) be an adversary that wins the game of Definition 3.1 with non-negligible correctness probability \( \varepsilon \), and let \( \mathcal{C} \) be a CDH challenger that provides us with the input \((G,A,B)\). We set up the ECTAKS protocol and compute \( C_1^i \rightarrow 2 \) as described in the proof of Lemma 3.7. For Phases 1 and 2 of the security game, we choose \( \alpha^{(i)} \in \mathbb{F}_p \backslash \{0\} \) uniformly at random and we can compute the response of \( \mathcal{A}'s \) queries as

\[
\left( C_1^{(i)} \rightarrow 2, E_1^{(i)} \rightarrow 2 \right) = \left( \alpha^{(i)} t_1 G, \alpha^{(i)} (k_1 \cdot m_1^{(i)} \rightarrow 2) B \right).
\]

As noted in Remark 1, the protocol is perfectly simulated, consequently the guess \( E_1^i \rightarrow 2 \) of \( \mathcal{A} \) is equal to \( E_1^i \rightarrow 2 = (k_1 \cdot m_1^i \rightarrow 2)(\alpha \beta G) \) with probability \( \varepsilon \), so we can send back to \( \mathcal{C} \) the value

\[
(k_1 \cdot m_1^i \rightarrow 2)^{-1} E_1^i \rightarrow 2,
\]

and we correctly solve the CDH problem with the same probability \( \varepsilon \). \( \square \)

Proof of Theorem 3.6. Let \( \mathcal{A} \) be an adversary that wins the game of Definition 3.4 with non-negligible advantage \( \varepsilon \), and let \( \mathcal{C} \) be a DDH challenger, that gives us the input \((G,A,B,T)\). We set up the ECTAKS protocol and compute \( C_1^i \rightarrow 2 \) as described in the proof of Lemma 3.7, and simulate Phases 1 and 2 exactly as in the previous proof. For the challenge phase we compute

\[
E_b = (k_1 \cdot m_1^i \rightarrow 2) T.
\]

Note that if \( T = \alpha \beta G \), then \( E_b = E_1^i \rightarrow 2 \) (and we have simulated the case \( b = 1 \)), while if \( T = \gamma G \), since both \( k_1, m_1^i \rightarrow 2 \) have been generated uniformly at random, then also \( k_1 \cdot m_1^i \rightarrow 2 \) is uniform in \( \mathbb{F}_p \backslash \{0\} \), so \( E_b \) is a correct simulation of \( E_0 \). Again, as noted in Remark 1, the protocol is perfectly simulated, so we can send back the guess \( b' \) of \( \mathcal{A} \) to \( \mathcal{C} \) and we obtain

\[
\left| \Pr [b' = 1, T = \alpha \beta G] - \Pr [b' = 1, T = \gamma G] \right| = \varepsilon,
\]

i.e. we solve the DDH problem with the same advantage \( \varepsilon \). \( \square \)
4. Conclusions

In this paper we have studied the formal security of the protocol ECTAKS introduced by Aragona et al. [1]. After describing the scheme and the security assumptions on the underlying discrete logarithm problem over elliptic curves, we have introduced two security models: the first is related to computational security with respect to key-recovery attacks, whereas the second is related to security with respect to key-indistinguishability attacks. In both cases, we have proven how an adversary able to successfully recover private key information can be turned into an efficient algorithm for computational or decisional Diffie-Hellman problems. The proofs presented in this paper can be easily adapted to variations of the protocol, where we consider different groups in which the Diffie-Hellman problems are hard in place of the group of the points of an elliptic curve.

References

[1] R. Aragona, R. Civino, N. Gavioli and M. Pugliese, An authenticated key scheme over elliptic curves for topological networks, preprint, arXiv:2006.02147. To appear in Journal of Discrete Mathematical Sciences & Cryptography
[2] D. Boneh, The decision Diffie-Hellman problem, Algorithmic Number Theory (Portland, OR), Lecture Notes in Comput. Sci., 1423, Springer, (1998), 48–63.
[3] W. Diffie and M. E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory, 22 (1976), 644–654.
[4] S. Marchesani, L. Pomante, M. Pugliese and F. Santucci, Definition and development of a topology-based cryptographic scheme for wireless sensor networks, in Sensor Systems and Software, Springer International Publishing, (2013), 47–64.
[5] S. Marchesani, L. Pomante, F. Santucci and M. Pugliese, A cryptographic scheme for real-world wireless sensor networks applications, in Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems, Association for Computing Machinery, 2013.
[6] M. Pugliese, Managing Security Issues in Advanced Applications of Wireless Sensor Networks, Ph.D thesis, Department of Electrical Engineering and Computer Science, University of L’Aquila, 2008, available at https://mpugliese.webnode.it/_files/200000061-a7608a760b/24.22phd_thesis.pdf.
[7] J. H. Silverman, The Arithmetic of Elliptic Curves, Springer-Verlag, New York, Graduate Texts in Mathematics, 2009.

Received November 2020; revised February 2021.

E-mail address: roberto.civino@univaq.it
E-mail address: riccardolongomath@gmail.com