Contents lists available at ScienceDirect
Discrete Applied Mathematics
journal homepage: www.elsevier.com/locate/dam

A general compiler for password-authenticated group key exchange protocol in the standard model

Fushan Wei[a,b], Neeraj Kumar[c,*], Debiao He[d], Sang-Soo Yeo[e]

[a] *State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China*
[b] *State Key Laboratory of Cryptology, Beijing, China*
[c] *Department of Computer Science and Engineering, Thapar University, Patiala, India*
[d] *State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China*
[e] *Division of Convergence Computer and Media, Mokwon University, Daejeon 302-729, Republic of Korea*

ARTICLE INFO

Article history:
Received 15 June 2015
Received in revised form 29 December 2015
Accepted 6 April 2016
Available online xxxx

Keywords:
Group communications
Password
Authenticated key exchange
Compiler

ABSTRACT

Password-authenticated group key exchange (PGKE) protocols are critical for ensuring secure group communications for mobile devices. Until now, only few PGKE protocols have been proposed. However, literature about group key exchange (GKE) protocols consists of many research proposals in last few years. In this paper, we present a protocol compiler based on smooth projective hash functions. The proposed compiler can transform any GKE protocol into a secure PGKE protocol by adding 2 rounds of communication. We conduct the security of our compiler in the standard model without using various other assumptions. Our compiler is round-efficient in the sense that a constant-round PGKE can be derived from the proposal if the underlying protocol is a constant-round GKE protocol.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

With the rapid development of the mobile communications and the cloud computing technologies, users can outsource and store their personal data to the cloud server to enjoy the ubiquitous services [30,32,24]. Consequently, user authentication and key establishment are important issues for ensuring data confidentiality and access control to protect user's sensitive data from unauthorized access [16,18,17,14,15,11]. Meanwhile, Owing to the rapid development of group-oriented applications such as collaboration works in cloud application, there are great needs for secure group communications. Group key exchange (GKE) protocols can establish a common secret session key for a set of participants. These participants can use the common session key to realize secure group communications. Due to the usefulness of GKE protocols, researchers design a lot of GKE protocols [20,6,8]. However, each participant should have its own public–private key pair, which requires expensive Public Key Infrastructure (PKI) to prove the validity of the public key. In order to avoid the use of PKI, the participants can use passwords for authentication. Password-authenticated group key exchange (PGKE) protocols can establish a common session for a group of people using a low-entropy password. PGKE protocols are convenient and attractive because users only need to remember a password and do not have to use complex PKI. However, PGKE protocols are vulnerable to dictionary attacks [23]. Designing a secure PGKE protocol is incredibly difficult and great caution should be taken when designing these protocols.

* Corresponding author.
E-mail addresses: weifs831020@163.com (F. Wei), neeraj.kumar@thapar.edu (N. Kumar), hedebiao@163.com (D. He), sangsooyeo@gmail.com (S.-S. Yeo).

http://dx.doi.org/10.1016/j.dam.2016.04.007
0166-218X/© 2016 Elsevier B.V. All rights reserved.

In 2002, Bresson et al. defined a formal model for PGKE protocols and then presented a PGKE protocol which is proven to be secure in their model [7]. However, the rounds of their protocol are linear with respect to the number of participants. As a result, their protocol is inefficient in terms of communication. In 2006, Abdalla et al. presented a constant-round PGKE protocol which is proven to be secure under the DDH assumption [3]. Their PGKE protocol is suitable for large groups due to its efficiency in terms of both communication and computation. Abdalla et al. also put forward a constant-round PGKE protocol without random oracles using the notion of smooth projective hash functions [5]. In 2009, Abdalla et al. presented the first UC (universal composability) secure PGKE protocol using idealizing assumptions [4]. Above mentioned protocols are all designed for the scenario in which all the participants share the same password. Recently, researchers considered the imbalanced network environment in which each client shares a different password with a powerful server and all clients want to generate a common secret key with the assistance of the server. Several PGKE protocols are proposed in this scenario [27,31,28].

Until now, only few PGKE protocols are proposed in the literature. However, the research on 2-party password-authenticated key exchange (2PAKE) and GKE protocols are extensive and rich of many results. Researchers begin to pay attention to protocol compilers which can convert any 2PAKE protocol or any GKE protocol to a PGKE protocol. Abdalla et al. presented a protocol compiler that can transform any secure 2PAKE protocol into a secure PGKE protocol based on non-interactive and non-malleable commitment in 2007 [2]. The security of the compiler is conducted in the standard model. In 2008, Wu et al. improved Abdalla et al.'s compiler by taking into account the dynamic property [29]. In 2010, Li et al. designed the first compiler which can transform any secure GKE protocol into a secure PGKE protocol [21]. Their compiler added 4 more rounds of communication to the underlying GKE protocol. The security of their compiler is conducted in the random oracle model. Recently, Hao et al. proposed another compiler which can convert any secure 2PAKE protocol into a secure PGKE protocol [12]. Their compiler preserves the round-efficiency of the underlying 2PAKE protocol. However, their compiler only has heuristic arguments and its security is not conducted in a formal security model.

In this paper, we pursue the line of the work in [21], and further investigate the question of how to obtain a PGKE protocol from a basic GKE protocol. Inspired by Gennaro et al.'s work [10], we use labeled encryption schemes, signature schemes and smooth projective hash functions as our main building blocks to propose a general compiler for PGKE protocols. How to prove the security of our compiler is the most difficult part of our work. Because an honestly generated ciphertext is re-used multiple times and the adversary can also forward the ciphertext to multiple sessions, we need to guarantee that the test keys (the outputs of smooth projective hash function) computed by different instances are independent with each other and the adversary is unable to distinguish the test keys from random ones. We present a technical theorem which involves reusing of the ciphertexts that may be of independent interest. The security proof of the compiler is in the standard model. Moreover, by adding 2 more rounds of communication, our compiler preserves the constant-round property of the underlying GKE protocol.

The organization of the paper is as follows. We recall the security model in the next section. We describe our compiler and prove its security in Sections 3 and 4, respectively. Finally, the paper is concluded in Section 5.

2. Security model

We briefly introduce the formal security model presented in [21] in this section. For more details, refer to [21].

Protocol participants. The participants in a PGKE protocol $\mathcal{P}$ is denoted by a nonempty set $\mathcal{U}$. All the participants share a human-rememberable password pw which is chosen randomly from a dictionary space $\mathcal{D}$. Each $U_i \in \mathcal{U}$ can execute several instances of the protocol $\mathcal{P}$ concurrently. We denote the tth instance of participant U_i by Π_i^t.

Adversarial model. We assume the adversary $\mathcal{A}$ to the protocol $\mathcal{P}$ is a probabilistic polynomial time (PPT) adversary, which controls the communications among the participants. We model the attack abilities of $\mathcal{A}$ using the following oracles:

- *Execute*$(\Pi_1^{t_1}, \ldots, \Pi_n^{t_n})$: This query models $\mathcal{A}$'s passive attack ability. $\mathcal{A}$ eavesdrops on a protocol execution among the instances $\Pi_1^{t_1}, \ldots, \Pi_n^{t_n}$. $\mathcal{A}$ finally gets all the exchanged messages during the execution.
- *Send*(Π_i^t, m): This query enables $\mathcal{A}$ to send a faked message to instance Π_i^t in the name of another participant. This query models $\mathcal{A}$'s active attack ability. The instance Π_i^t will respond according to the description of the protocol and sends back the message it will generate upon receiving m to $\mathcal{A}$.
- *Reveal*(Π_i^t): This query is used to capture the known key attack. Through this query, $\mathcal{A}$ gets the session key of the instance Π_i^t.
- *Test*(Π_i^t): This query does captures $\mathcal{A}$'s real attack ability. It is used to measure the session key security of instance Π_i^t. Upon receiving the query from $\mathcal{A}$, the simulator flips a coin, if the random bit is $b = 1$, then sends the real session key to $\mathcal{A}$. Otherwise, the simulator sends a random key of the same size to $\mathcal{A}$. $\mathcal{A}$ will guess the value of b to win the game.

The definitions of session identifications, partner identifications are still as usual. We omit these definitions for simplicity. For more details, refer to [21]. Let sid_i^t and pid_i^t be the session identification and partner identification of the instance Π_i^t, respectively. We say two instances Π_i^t and $\Pi_{i'}^{t'}$ are partnered if and only if $pid_i^t = pid_{i'}^{t'}$ and $sid_i^t = sid_{i'}^{t'}$.

Freshness An instance Π_i^t is said to be *fresh* if: (1) Π_i^t has accepted; (2) there has been no *Reveal* query to Π_i^t or its partner.