Cryptanalysis of an Efficient Diffie-Hellman Key Agreement Protocol Based on Elliptic Curves

Shengbao Wang, Zhenfu Cao, and Rongxing Lu

Department of Computer Science and Engineering,
Shanghai Jiao Tong University
800 Dongchuan Road, Shanghai 200240, P.R. China
{shengbao-wang, cao-zf, rxlu}@cs.sjtu.edu.cn
http://tdt.sjtu.edu.cn

Abstract. In 2005, Strangio proposed an efficient Diffie-Hellman two-party key agreement protocol based on elliptic curves. In this letter, we reveal a security weakness in Strangio's protocol, whereby we demonstrate that the protocol is insecure against a key-compromise impersonation attack.

Key Words. Key agreement, protocols, elliptic curves, key-compromise impersonation attack.

1 Introduction

Key agreement protocols allow two communicating parties to establish a secret *session key* over an insecure network. The session key may subsequently be used to achieve some cryptographic goal, such as confidentiality or data integrity. As such key agreement protocols are a central piece for ensuring secure communications, and are one of the most commonly used cryptographic protocols. In 1976, Diffie and Hellman [2] proposed the first two-party key agreement protocol in their seminal paper which marks the birth of public key cryptography. If in a key agreement protocol one party is assured that no other party aside from the specially identified party may gain access to the particular established secret key, then the protocol is said to provide *implicit key authentication* (IKA). A key agreement protocol that provides mutual IKA between the two parties is called an *authenticated key agreement* (AK) protocol [1].

In 2005, Strangio [4] proposed an efficient two-pass (1-round) elliptic curve key agreement protocol (ECKE-1). It was claimed that ECKE-1 achieves all the desirable security attributes of an AK protocol, such as known-key security, forward secrecy, unknown key-share resilience, imperfect key control, and particularly, *key-compromise impersonation resilience*, which means that an adversary who obtains the long-term private key of an honest entity, say A, is not able to impersonate another honest entity, say B, to A.

In this letter, however, we show that Strangio's protocol ECKE-1 is actually insecure against key-compromise impersonation attacks. In the following, we first review ECKE-1 [4] in Section 2. And then in Section 3 we present our key-compromise impersonation attack on it. The concluding remark will be followed in Section 4.

2 Review of Strangio's Protocol

Here we briefly review Strangio's protocol ECKE-1 [4]. The protocol is defined in an elliptic curve $E(\mathbb{F}_q)$ over a finite field $\mathbb{F}_q$, where q is a prime power. The number of points on $E(\mathbb{F}_q)$ is denoted by $\#E(\mathbb{F}_q)$. We use the same notations as in [4]:

- P $(= (P.x, P.y))$: the base point of large prime order in $E(\mathbb{F}_q)$

– P_∞: the identity point in $E(\mathbb{F}_q)$
– n: the order of P in $E(\mathbb{F}_q)$
– h: the cofactor of n, i.e., $h = \#E(\mathbb{F}_q)/n$
– (ω_U, W_U): a user's long-term key pair, ω_U, $1 \leq \omega_U \leq n-1$, is the private key and $W_U (= \omega_U P)$ is the public key
– id_U: a user's identity information
– $\mathcal{F}_1, \mathcal{F}_2$: two independent hash functions satisfying $\mathcal{F}_1, \mathcal{F}_2 : \{0,1\}^* \to \mathbb{F}_q$
– $\mathcal{G}$: a key derivation function satisfying $G : \mathbb{F}_q \to \{0,1\}^*$

Assume that A and B are two honest entities who run the protocol correctly and that their public keys are exchanged via certificates. The protocol works in the following steps (cf. Fig. 1):

A		B
$r_A \in_R [1, n-1]$		$r_B \in_R [1, n-1]$
$e_A = \mathcal{F}_1(r_A, \omega_A, id_A)$		$e_B = \mathcal{F}_1(r_B, \omega_B, id_B)$
$Q_A = (r_A + e_A\omega_A)P$		$Q_B = (r_B + e_B\omega_B)P$
	$\xrightarrow{Q_A}$	
	$\xleftarrow{Q_B}$	
$d_A = \omega_A \mathcal{F}_2(Q_A.x, Q_B.x, id_A, id_B)$		$d_B = \omega_B \mathcal{F}_2(Q_A.x, Q_B.x, id_A, id_B)$
$T_A = h((r_A + e_A\omega_A)Q_B + d_A W_B)$		$T_B = h((r_B + e_B\omega_B)Q_A + d_B W_A)$
$sk = \mathcal{G}(T_A.x)$		$sk = \mathcal{G}(T_B.x)$

Fig. 1. Strangio's protocol ECKE-1 [4]

1. A picks a random r_A such that $r_A \in [1, n-1]$ and then computes $e_A = \mathcal{F}_1(r_A, \omega_A, id_A)$. In the same way, B picks a random r_B such that $r_B \in [1, n-1]$ and then computes $e_B = \mathcal{F}_1(r_B, \omega_B, id_B)$;
2. A computes $Q_A = (r_A + e_A\omega_A)P$. Symmetrically, B computes $Q_B = (r_B + e_B\omega_B)P$;
3. If $Q_A = P_\infty$ (resp. $Q_B = P_\infty$), A (resp. B) returns to step 2. Otherwise, A initiates a protocol run with B by sending Q_A to B;
4. B invokes a procedure to perform public-key validation of Q_A, namely to verify that Q_A is actually a point in the group $E(\mathbb{F}_q)$, and aborts the protocol run if the validation fails. Otherwise, B sends Q_B to A as the response message;
5. A invokes a procedure to perform public-key validation of Q_B and aborts the protocol run if the validation fails;
6. A and B compute, respectively, the points T_A and T_B;
7. Both A and B terminate holding the session key sk (see also Fig. 1).

Correctness of the protocol follows from the fact that after any honest execution we have $T_A = T_B$, therefore A and B will both compute the same secret session key sk from $h(r_A r_B + r_B e_A \omega_A + r_A e_B \omega_B + e_A e_B \omega_A \omega_B + d\omega_A \omega_B)P$[1], in which $d = \mathcal{F}_2(Q_A.x, Q_B.x, id_A, id_B)$.

The scalar multiplication by the cofactor h prevents the small-subgroup attack [3]. On the surface, ECKE-1 closely resembles the well-known MQV protocol [3] in that, on the one hand, ECKE-1 follows the same approach (namely by introducing e_A and e_B) to destroy the algebraic structure of the group, on the other hand, the session key generation

[1] Notice that there are in fact some typos in [4], where the author mistakenly writes this term as $h(r_A r_B + r_B e_A \omega_B + r_A e_B \omega_A + e_A e_B \omega_A \omega_B + d_A d_B \omega_A \omega_B)P$.

methods used in the two protocols are very similar. Strangio [4] claimed that ECKE-1, just like the MQV protocol, achieves key-compromise impersonation resilience. Unfortunately, our attack presented in the next section invalidates this security claim. And we note that it is the main difference between the two protocols, i.e. the protocol message in ECKE-1 is in the form of $(r + e\omega)P$ while that in MQV is rP, that makes our attack on ECKE-1 feasible.

3 A Key-Compromise Impersonation Attack on ECKE-1

Consider the scenario in which the long-term private key of an entity is disclosed. Obviously, an adversary (denoted by E) who obtains this key is able to impersonate her to other entities, since the long-term private key is the only key that exactly identifies her. However, it is desired in most circumstances that this disclosure does not enable the adversary to impersonate other entities to her. Accordingly, a *key-compromise impersonation attack* is an attack whereby E, with some entity's long-term private key on hand, masquerades as a different entity and tries to establish a valid session key with the compromised party. Note that key-compromise impersonation attack implies a serious threat since an entity may not even be aware that her computer was corrupted and that an adversary has obtained her long-term private key.

Suppose A's long-term private key (i.e. ω_A) is disclosed. We now show that ECKE-1 is vulnerable to the key-compromise impersonation attack depicted in Fig. 2, in which $E(B)$ indicates that E is impersonating B to A. The detailed attack scenario is as follows:

1. $E(B)$ first picks a random $r_{E(B)}$ such that $r_{E(B)} \in [1, n-1]$ and then computes $Q_{E(B)} = r_{E(B)}P$;
2. A first picks a random r_A such that $r_A \in [1, n-1]$ and then computes $e_A = \mathcal{F}_1(r_A, \omega_A, id_A)$ and subsequently, $Q_A = (r_A + e_A\omega_A)P$;
3. If $Q_A = P_\infty$, A returns to step 2. Otherwise, A initiates a protocol run by sending Q_A, with B being the intended recipient;
4. $E(B)$ intercepts Q_A and sends $Q_{E(B)}$ to A, purportedly as B's response message;
5. A (resp. $E(B)$) invokes a procedure to perform public-key validation of $Q_{E(B)}$ (resp. Q_A) and aborts the protocol run if the validation fails;
6. A and $E(B)$ compute, respectively, the points T_A and $T_{E(B)}$;
7. Both A and $E(B)$ terminate holding the session key sk (see also Fig. 2).

A		$E(B)$
$r_A \in_R [1, n-1]$ $e_A = \mathcal{F}_1(r_A, \omega_A, id_A)$ $Q_A = (r_A + e_A\omega_A)P$		$r_{E(B)} \in_R [1, n-1]$ $Q_{E(B)} = r_{E(B)}P$
	$\xrightarrow{Q_A}$	
	$\xleftarrow{Q_{E(B)}}$	
$d_A = \omega_A\mathcal{F}_2(Q_A.x, Q_{E(B)}.x, id_A, id_B)$ $T_A = h((r_A + e_A\omega_A)Q_{E(B)} + d_A W_B)$ $sk = \mathcal{G}(T_A.x)$		$d_{E(B)} = \omega_A\mathcal{F}_2(Q_A.x, Q_{E(B)}.x, id_A, id_B)$ $T_{E(B)} = h(r_{E(B)}Q_A + d_{E(B)}W_B)$ $sk = \mathcal{G}(T_{E(B)}.x)$

Fig. 2. Key-compromise impersonation attack on ECKE-1

The above attack is successful since the two points computed by A and $E(B)$, i.e. T_A and $T_{E(B)}$, are equal, thus they will both compute the same secret session key sk. We prove this as follows (Obviously, we have $d_{E(B)} = d_A$):

$$\begin{aligned} T_A &= h((r_A + e_A\omega_A)Q_{E(B)} + d_A W_B) \\ &= h((r_A + e_A\omega_A)r_{E(B)}P + d_A W_B) \\ &= h(r_{E(B)}(r_A + e_A\omega_A)P + d_{E(B)}W_B) \\ &= h(r_{E(B)}Q_A + d_{E(B)}W_B) \\ &= T_{E(B)}. \end{aligned}$$

Therefore, when A wants to create a secure communication with any specific entity, E can always intercept the first protocol message Q_A and subsequently impersonate the specific entity to A, until the compromise is detected and the long-term key is revoked. Note that in our attack, E does not necessarily have a valid long-term key pair. In other words, our attack against ECKE-1 is powerful since even an outside adversary is capable of launching it.

4 Conclusion

Key agreement protocols play a central role in secure communications, however, the protocol design is extremely error-prone due to the inherent complexity of this problem. In this letter we have shown that, contrary to the author's security claim in [4], the efficient authenticated key agreement protocol ECKE-1 is insecure against key-compromise impersonation attacks.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 60673079 and 60572155.

References

1. S. Blake-Wilson and A. Menezes, "Authenticated Diffie-Hellman key agreement protocols," in *Proc. Selected Areas in Cryptography 1998*, LNCS vol. 1556, pp. 339-361. Springer-Verlag, 1999.
2. W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory* 22(6), pp. 644 - 654, 1976.
3. L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, "An efficient protocol for authenticated key agreement, Dept. C & Q, Univ. of Waterloo, CORR 98-05, 1998.
4. M. A. Strangio, "Efficient Diffie-Hellmann two-party key agreement protocols based on elliptic curves," in *Proc.* 20^{th} *ACM Symposium on Applied Computing (SAC)*, pp. 324-331, ACM Press, 2005.