J. Math. Crypt. **1** (2007), 47–64 © de Gruyter 2007
DOI 10.1515 / JMC.2007.004

Another look at HMQV

Alfred Menezes

Communicated by Spyros S. Magliveras

Abstract. The HMQV protocols are ‘hashed variants’ of the MQV key agreement protocols. They were introduced at CRYPTO 2005 by Krawczyk, who claimed that the HMQV protocols have very significant advantages over their MQV counterparts: (i) security proofs under reasonable assumptions in the (extended) Canetti-Krawczyk model for key exchange; and (ii) superior performance in some situations.

In this paper we demonstrate that the HMQV protocols are insecure by presenting realistic attacks in the Canetti-Krawczyk model that recover a victim’s static private key. We propose HMQV-1, patched versions of the HMQV protocols that resists our attacks (but do not have any performance advantages over MQV). We also identify some fallacies in the security proofs for HMQV, critique the security model, and raise some questions about the assurances that proofs in this model can provide.

Key words. Cryptography, key agreement protocols, provable security

AMS classification. 94A60, 68P25, 11T71

1 Introduction

The MQV protocols [28] are a family of efficient Diffie-Hellman authenticated protocols that have been widely standardized [2, 3, 19, 20, 37]. The two-pass protocol in this family (which is what the term “the MQV protocol” sometimes refers to) provides implicit key authentication. A three-pass variant adds key confirmation, thus providing explicit key authentication, while a one-pass variant is useful for applications such as email where only the sender may be online. The protocols were designed to meet a variety of security goals even in the presence of active attacks.

The security of the MQV protocols has been intensively studied since 1995. However, until recently there was no attempt to ‘prove’ that the protocols actually meet their security objectives. In a recent paper (see [26] and the expanded version [27]), Krawczyk undertook the ambitious task of trying to prove the security of the MQV protocols in the Canetti-Krawczyk model for key exchange [14, 15, 16]. This model provides a formal security definition by carefully specifying the capabilities and goals of an attacker.

Krawczyk’s analysis of MQV [26, 27] resulted in his concluding that “…MQV falls to a variety of attacks in [the Canetti-Krawczyk] model that invalidate its basic security as well as many of its stated security goals” and “…raises clear concerns about the security of the protocol…” He then proceeded to modify the MQV protocols to produce HMQV, a ‘hashed variant’ that “…provides the same superb performance and functionality of the original protocol but for which all the MQV security goals can be formally proved to hold in the random oracle model under the computational Diffie-Hellman assumption."

The HMQV protocols are no less efficient than their MQV counterparts because the additional hashing performed takes negligible time compared to the group exponentiations. In fact, the HMQV protocols are significantly faster in some cases because they dispense with the potentially expensive public-key validations that are mandated in MQV. An important result stated in [26, 27] is that this performance improvement of HMQV is *proven* not to affect security in any way.

We show in §2 of this paper that the omission of public-key validation in the HMQV protocols is a fatal security weakness. We accomplish this by presenting small-subgroup attacks that let an attacker recover a victim's static private key. Our attacks on the HMQV protocols are more realistic and have much more damaging consequences than any of the proposed attacks on MQV in [26, 27]. Our attacks can be circumvented by requiring that all public keys be validated, leading to new variants of the HMQV protocols that we call Patched HMQV (HMQV-1). As a consequence of reinstating public-key validation, the HMQV-1 protocols no longer have any performance advantages over their MQV counterparts. We also present an unknown-key share attack on the one-pass HMQV-1 protocol which demonstrates a deficiency in the binding between the identities of the communicating parties and the key derivation.

In §3 we identify some flaws in the security proofs given in [27]. We critique some aspects of the Canetti-Krawczyk model, and discuss the difficulty of obtaining security reductions in this model that are meaningful in practice.

In §4 the purported attacks on MQV that were listed in [26, 27] are analyzed and critiqued. The paper concludes with some remarks in §5.

2 The HMQV key agreement protocols

Let G' be a multiplicatively-written abelian group of order N, and let $G = \langle g \rangle$ be a subgroup of G' of prime order q. We call G the *main group* and G' the *supergroup*. The integer $h = N/q$ is called the *cofactor*. We assume that $\gcd(h, q) = 1$, from which it follows that G is the unique subgroup of G' having order q. The fundamental security requirement that determines the suitability of a group for implementing Diffie-Hellman key agreement protocols is that the discrete logarithm problem (DLP) in G should be intractable. Groups arising from finite fields and from elliptic curves are the only ones presently in widespread use.

In finite field systems, G' is the multiplicative group $\mathbb{Z}_p^*$ of a prime field $\mathbb{Z}_p$ and G is the subgroup of order q for some prime divisor q of $p - 1$. As in DSA, the primes p and q are usually chosen so that the bitlength of q is considerably smaller than that of p. This is because the DLP in $\mathbb{Z}_p^*$ can be solved using index-calculus methods whose running times are subexponential in p, whereas the best algorithms known for solving the DLP directly in G have running time $O(\sqrt{q})$. If p and q are selected in this way, then the cofactor h is much larger than q. For example, to achieve an 80-bit security level[1] against discrete logarithm attacks, one might take the bitlengths of p and q to be

[1]A *k-bit security level* means that an attacker is expected to expend at least 2^k effort before it can solve a

1024 and 160, respectively; in this case the cofactor h has bitlength about 864.

In elliptic curve systems, G' is the group of points $E(L)$ on an elliptic curve E defined over a finite field L, and G is a subgroup of $E(L)$ of prime order q.[2] Elliptic curves of (almost) prime order can be easily found, so in practice the cofactor h is very small. For example, the 15 NIST-recommended elliptic curves [18] have $h = 1, 2$ or 4.

2.1 Protocol descriptions

The least non-negative residue of an integer x modulo t is denoted by $\lfloor x \rfloor_t$. The following notation is from [27] and will be used throughout this paper.

$\hat{A}$, $\hat{B}$	Identities of two communicating parties.
a, b	Static private keys of $\hat{A}$ and $\hat{B}$; $a, b \in_R [1, q-1]$.
A, B	Static public keys of $\hat{A}$ and $\hat{B}$; $A = g^a$, $B = g^b$.
x, y	Ephemeral private keys of $\hat{A}$ and $\hat{B}$; $x, y \in_R [1, q-1]$.
X, Y	Ephemeral public keys of $\hat{A}$ and $\hat{B}$; $X = g^x$, $Y = g^y$.
H	A hash function.
$\overline{H}$	An l-bit hash function, where $l = (\lfloor \log_2 q \rfloor + 1)/2$.
MAC	Message authentication code algorithm.
K_m	MAC key.
K	Session key.

The identities $\hat{A}$ and $\hat{B}$ may also include certificates for their respective static public keys.

2.1.1 Description of two-pass HMQV

The parties exchange their identifiers and ephemeral public keys. That is, $\hat{A}$ sends $(\hat{A}, \hat{B}, X)$ to $\hat{B}$, while $\hat{B}$ sends $(\hat{B}, \hat{A}, Y)$ to $\hat{A}$. Then $\hat{A}$ checks[3] that $Y \neq 0$ and computes the *shared secret*

$$\sigma_{\hat{A}} = (YB^e)^{x+da}, \tag{2.1}$$

while $\hat{B}$ checks that $X \neq 0$ and computes

$$\sigma_{\hat{B}} = (XA^d)^{y+eb}, \tag{2.2}$$

where

$$d = \overline{H}(X, \hat{B}) \quad \text{and} \quad e = \overline{H}(Y, \hat{A}). \tag{2.3}$$

Notice that $\sigma_{\hat{A}} = \sigma_{\hat{B}} = g^{(x+da)(y+eb)}$, and so both parties can compute the *session key* $K = H(\sigma_{\hat{A}}) = H(\sigma_{\hat{B}})$.

computational problem or defeat the security objectives of a protocol.

[2]The elliptic curve group law is usually written additively, but in this paper we shall use multiplicative notation for the sake of consistency.

[3]If G' is the multiplicative group of a finite field, then the element 0 in checks such as "$Y \neq 0$" refers to the zero element of the finite field. However, since the HMQV protocols are described for generic cyclic groups G, such checks are not well defined since it is not clear what is meant by the element "0".

2.1.2 Description of three-pass HMQV

Three-pass HMQV adds key confirmation to the two-pass HMQV protocol. After receiving $(\hat{A}, \hat{B}, X)$ from the initiator $\hat{A}$, the responder $\hat{B}$ checks that $X \neq 0$, computes $\sigma_{\hat{B}}$, $K_m = H(\sigma_{\hat{B}}, 0)$, $K = H(\sigma_{\hat{B}}, 1)$, and sends $(\hat{B}, \hat{A}, Y, \text{MAC}_{K_m}(\text{“1”}))$ to $\hat{A}$. Upon receipt of this message, $\hat{A}$ checks that $Y \neq 0$, computes $\sigma_{\hat{A}}$, $K_m = H(\sigma_{\hat{A}}, 0)$, $K = H(\sigma_{\hat{B}}, 1)$. $\hat{A}$ verifies the MAC tag received, and sends $\text{MAC}_{K_m}(\text{“0”})$ to $\hat{B}$. Finally, $\hat{B}$ verifies the latter MAC tag.

2.1.3 Description of one-pass HMQV

In one-pass key agreement protocols, only the sender can contribute an ephemeral public key. In one-pass HMQV, the sender $\hat{A}$ transmits the single message $(\hat{A}, \hat{B}, X)$ to $\hat{B}$. Let $d = \overline{H}(X, \hat{A}, \hat{B})$. The session key computed by $\hat{A}$ is $K = H(B^{x+da})$, while $\hat{B}$ computes the same key (after checking that $X \neq 0$) as $K = H((XA^d)^b)$.

2.2 The insecurity of HMQV

We present ‘small-subgroup attacks’ on all three versions of HMQV described in §2.1. The adversary’s actions in all these attacks are legitimate in that they are allowable within the Canetti-Krawczyk security model (cf. §3.1). The attacks have the most damaging consequences possible since the attacker is able to learn the victim’s static private key.

Our attacks exploit the omission in the HMQV protocols of public-key validation of static and ephemeral public keys. That is, the receiver of an element X does not take any steps to verify that X is indeed an element of order q in the main group G – the only required check is $X \neq 0$. Also, a certification authority who issues a certificate to party $\hat{A}$ for its static public key A does not take any steps to verify that A is an element of order q in G. These omissions are by design rather than by accident. They are desirable because the parties and certification authority do not have to perform costly exponentiations by q to verify that a supergroup element is also in G, and also because some certification authorities may not be configured to do such tests [27]. In [27] proofs are claimed that all three versions of HMQV are secure without these checks. Our attacks demonstrate that the security proofs must be fallacious. Some errors in the proofs will be identified in §3.2.

2.2.1 Attack on one-pass HMQV

The attacker $\hat{A}$, who wishes to learn $\hat{B}$’s static private key, selects an element $\gamma \in G'$ of order t for some small prime divisor $t \neq q$ of N. (Recall that N is the order of the supergroup G'.) Here by ‘small’ we mean that $\hat{A}$ can feasibly perform t group operations and hash function evaluations. The attacker selects $a \in [1, t-1]$, and obtains a certificate for her static public key $A = \gamma^a$. She then selects $x \in [1, t-1]$, computes $X = \gamma^x$, and sends $(\hat{A}, \hat{B}, X)$ to $\hat{B}$. Upon receipt, $\hat{B}$ verifies that $X \neq 0$ and computes the session key $K = H((XA^d)^b) = H((\gamma^{x+da})^b)$ where $d = \overline{H}(X, \hat{A}, \hat{B})$. Next, $\hat{A}$

learns K by issuing a session-key query[4] and computes $\beta = \gamma^{x+da}$. $\hat{A}$ then computes $K' = H(\beta^c)$ for $c = 0, 1, 2, \ldots, t-1$ until $K' = K$. When the latter condition is met, $\hat{A}$ knows that $c = \lfloor b \rfloor_t$. After repeating this procedure for several different primes t, the value b can be easily determined by the Chinese Remainder Theorem.

2.2.2 Feasibility of the attack

We first note that session-key queries are allowable in the Canetti-Krawczyk model. However, in practice $\hat{A}$ may not even have to issue a session-key query. For example, if $\hat{B}$ were to subsequently send $\hat{A}$ an authenticated message $(m, T = \text{MAC}_K(m))$, then $\hat{A}$ could determine $\lfloor b \rfloor_t$ by iteratively computing $K' = H(\beta^c)$ and $T' = \text{MAC}_{K'}(m)$ for $c = 0, 1, 2, \ldots$ until $T' = T$. As before, $\hat{A}$ then deduces that $c = \lfloor b \rfloor_t$.

Is it reasonable to expect that N/q has several small prime factors? In the case of DSA-like parameters, the cofactor $h = (p-1)/q$ is large and thus one can expect it to have several small prime factors if p and q were randomly generated. However, if it happens that $(p-1)/q$ does not have several small prime factors, the extreme case occurring when $(p-1)/q$ is twice a prime, then the attack cannot be mounted.

In the case of elliptic curve groups, we have noted that the cofactor h is very small, and sometimes $h = 1$, so the attack as described above will generally fail. However, $\hat{A}$ has the option of mounting an *invalid-curve attack* [4] which we describe next. Suppose that $G' = E(L)$ where E is an elliptic curve defined over a finite field L. The attacker selects an elliptic curve E' defined over L whose Weierstrass equation $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ differs from E's only in the coefficient a_6, and such that the order of $E'(L)$ contains a prime factor t of the desired size. Then, as observed by Biehl, Meyer and Müller [10], the usual formulas for adding points in $E(L)$ do not involve a_6 and thus are identical to the formulas for adding points in $E'(L)$. If $\hat{A}$ selects order-t points $X, A \in E'(L)$ and $\hat{B}$ does not check whether $X, A \in E(L)$, then $\hat{B}$ will successfully compute $(XA^d)^b \in E'(L)$ and the attack can proceed as before. Unlike in the case of DSA-like groups, the invalid-curve attack with elliptic curves is certain to succeed since there is an abundance of elliptic curves over L with group orders divisible by small primes.

We note that $E'(L)$ is not a supergroup of the main group of $E(L)$. One could perhaps argue that the HMQV specification implicitly required the recipient of an element X to check that it belongs to the supergroup $E(L)$, thus thwarting the attack. However, we maintain that it is reasonable to suppose that this check is *not* likely to be performed since points in $E'(L)$ are represented in the same way as points in $E(L)$,[5] and since the HMQV specification [27] explicitly states that no checks are required other than $X \neq 0$.

[4]A well-designed key agreement protocol should achieve its security goals even if an attacker is able to learn some session keys. This is because a key agreement protocol cannot guarantee that session keys won't be subsequently used in insecure applications (e.g., to encrypt messages with a weak symmetric-key encryption scheme).

[5]For example, if $L = \mathbb{F}_{2^m}$, then a point in $E(L)$ or $E'(L)$ is represented by a pair of bitstrings of length m.

2.2.3 Attack on two-pass HMQV

As in the attack on one-pass HMQV, the attacker $\hat{A}$ selects an element $\gamma \in G'$ of order t, selects a static public key $A = \gamma^a$, and sends an ephemeral public key $X = \gamma^x$ to $\hat{B}$. While $\hat{B}$ is computing the session key, $\hat{A}$ issues a state-reveal query[6] and learns $\hat{B}$'s ephemeral private key y. After $\hat{B}$ has computed $\beta = XA^d$ and the session key $K = H(\beta^{y+eb})$, $\hat{A}$ learns K via a session-key query. Now $\hat{A}$ can obtain $c = \lfloor b \rfloor_t$ by iteratively computing $K' = H(\beta^{y+ec})$ for $c = 0, 1, 2, \ldots$ until $K' = K$.

This attack is not as realistic as the attack on one-pass HMQV since a state-reveal query is required. Nonetheless, it is relevant to our analysis of the claims in [27], which include the author's claim to have *proven* resistance of two-pass HMQV to damage from leakage of ephemeral private keys.[7]

2.2.4 Lattice attack on two-pass HMQV

The attack in §2.2.3 assumes that $\hat{B}$ does not reduce the exponent $y + eb$ prior to computing β^{y+eb}. This assumption is reasonable because the HMQV specification does not require $y + eb$ to be reduced. Moreover, it is not at all clear how $\hat{B}$ should reduce $y + eb$ because the order of β is unknown to $\hat{B}$ (since X and A are not validated as belonging to the main group G of order q). Nonetheless, one can expect that $\hat{B}$ will reduce the exponent $y + eb$ in order to accelerate the computation of β^{y+eb}. So, let us suppose that $\hat{B}$ computes β^s where $s = \lfloor y + eb \rfloor_q$. Then the following small-subgroup attack can be mounted.

The attacker $\hat{A}$ selects $A = \gamma^a$ and $X = \gamma^x$, where $\gamma \in G'$ has order $t = 2^r$ for some small r (e.g., $r = 6$). $\hat{A}$ then learns y and $K = H(\beta^s)$ through state-reveal and session-key queries, respectively. Now, $\hat{A}$ can determine $\lfloor s \rfloor_t$ by exhaustive search, thereby obtaining the r least significant bits of s. After repeating this procedure a few times (e.g., 30 times), $\hat{A}$ can use the lattice attack of Leadbitter and Smart[8] [29] (see also [31]) to find the $f/2$ most significant bits of b; here f is the bitlength of q. The remaining $f/2$ bits of b can then be found in $O(q^{1/4})$ steps using Pollard's lambda method [34].

2.2.5 Attacks on three-pass HMQV

The attacks on three-pass HMQV are similar to the ones on two-pass HMQV in §2.2.3 and §2.2.4 except that a session-key query is not needed. This is because $\hat{A}$ learns MAC_{K_m}("1") from $\hat{B}$'s response, and can use this tag to recognize a correct guess for $\lfloor b \rfloor_t$ or $\lfloor s \rfloor_t$.

[6] State-reveal queries capture the possibility of an attacker learning some secret information such as ephemeral private keys that are specific to a particular session. Such queries do not reveal static private keys.

[7] In [26, 27], the author points out that resistance of Diffie-Hellman protocols to damage from the disclosure of ephemeral private keys is 'a prime security concern'. This is because many applications can be expected to precompute ephemeral pairs (x, X) and (y, Y) in order to improve performance, and these stored pairs become more vulnerable to disclosure.

[8] The Leadbitter-Smart lattice attack does not require complete knowledge of y; it only needs the r least significant bits of y.

2.3 Patched HMQV (HMQV-1)

Our small-subgroup attacks on the HMQV protocols are thwarted if public keys are validated as mandated in MQV [28]. That is, the certification authority should verify that a party's static public key A is indeed an element of order q in the main group G before issuing a certificate. Similarly, the recipient of an ephemeral public key X should verify that X is an element of order q in G before proceeding with the next step of the protocol.

If the cofactor h is small, as is the case with elliptic curve groups, the computational cost of ephemeral public-key validation can be reduced significantly by requiring only that the recipient $\hat{B}$ of X verify that $X \in G'$ and that $(\sigma_{\hat{B}})^h \neq 1$ before computing the session key $H((\sigma_{\hat{B}})^h)$.[9] This process is called 'embedded' public-key validation in [28].

2.4 Unknown-key share attack on one-pass HMQV-1

In an unknown-key share (UKS) attack, first formulated by Diffie, van Oorschot and Wiener [17], an adversary interferes with $\hat{A}$'s and $\hat{B}$'s communications. The result is that $\hat{A}$ and $\hat{B}$ still compute the same session key. However, even though $\hat{A}$ correctly believes the key is shared with $\hat{B}$, $\hat{B}$ mistakenly believes that the key is shared with a third party $\hat{C} \notin \{\hat{A}, \hat{B}\}$.

One-pass HMQV-1 succumbs to the following online UKS attack; such attacks were first mounted by Kaliski on the two-pass MQV protocol [22]. Party $\hat{A}$ transmits the message $(\hat{A}, \hat{B}, X)$ and computes the session key $K = H(B^{x+da})$. The message is intercepted by the adversary $\hat{C}$, who selects an arbitrary integer $u \in [1, q-1]$ and computes $X' = XA^d g^{-u}$, $d' = \overline{H}(X', \hat{C}, \hat{B})$, and $C = g^{u(d')^{-1} \bmod q}$. $\hat{C}$ registers C as her static public key (with $c = u(d')^{-1} \bmod q$ being the corresponding static private key), and then transmits $(\hat{C}, \hat{B}, X')$ to $\hat{B}$. Note that X' and C are valid public keys. Party $\hat{B}$ then computes the shared secret

$$(X'C^{d'})^b = (XA^d g^{-u}(g^{u(d')^{-1}})^{d'})^b = (XA^d g^{-u} g^u)^b = (XA^d)^b = B^{x+da}.$$

Thus $\hat{B}$ computes the same session key K as $\hat{A}$, even though $\hat{B}$ believes that the key is shared with $\hat{C}$.

A similar UKS attack can also be mounted on the variant of one-pass HMQV-1 suggested in [27, Remark 9.1] whereby $\hat{A}$ and $\hat{B}$ compute the shared secret as $(BB^e)^{x+da}$ and $(XA^d)^{b+eb}$ respectively, where $d = \overline{H}(X, \hat{A}, \hat{B})$ and $e = \overline{H}(B, \hat{B}, \hat{A})$. To mount the attack, the adversary $\hat{C}$ intercepts $\hat{A}$'s message $(\hat{A}, \hat{B}, X)$ and replaces it with $(\hat{C}, \hat{B}, X')$ where $X' = (XA^d g^{-u})^{(1+e)(1+e')^{-1}}$, $e' = \overline{H}(B, \hat{B}, \hat{C})$, $d' = \overline{H}(X', \hat{C}, \hat{B})$, and $C = g^{u(d')^{-1}(1+e)(1+e')^{-1}}$. Once again, $\hat{A}$ and $\hat{B}$ compute the same session key even though $\hat{B}$ believes that the key is shared with $\hat{C}$.

These UKS attacks illustrate the unsuitability of one-pass HMQV-1 for use as an authenticated CCA encryption scheme or as a verifier-specific signature scheme as

[9]Thus, contrary to what is stated in [27], it is not the case that ephemeral public-key validation is always as costly as a full exponentiation in G.

described in [27]. The UKS attacks are thwarted if the identifiers $\hat{A}$ and $\hat{B}$ are included in the key derivation function, i.e., $K = H(\sigma, \hat{A}, \hat{B})$.

3 Critique of the formal analysis in [27]

Formal analysis of a cryptographic protocol entails the following steps.

(i) (*security model*) Providing a clear specification of the capabilities of an adversary and how it interacts with honest parties.

(ii) (*security definition*) Providing a clear description of the goals of the adversary.

(iii) (*security proof*) Proving that the only way an adversary can meet its goals is by solving a computational problem that is widely believed to be intractable.

The proof in step (iii) typically takes the form of a *reduction*, whereby one shows how a (hypothetical) adversary who succeeds in its task can be used to solve the hard computational problem with little additional effort.[10]

The declaration that a cryptographic protocol is 'provably secure' should always be met with a healthy dose of skepticism. First, one should verify that the security model and definition are the 'right' ones, i.e., they accurately capture the capabilities of attackers that one could expect in the environments in which the protocol will eventually be deployed. Second, the proof should be checked for correctness. This step is notoriously difficult since the proofs are typically long, complicated, and often poorly written. History has shown that subtle flaws may take years to be discovered. Indeed, Stern [38] has proposed adding a validation step to any security proof:

> Also, the fact that proofs themselves need time to be validated through public discussion was somehow overlooked.

Finally, one should carefully consider what the proof actually means, i.e., what assurances are actually provided when the protocol is deployed. This step involves examining the reasonableness of the stated assumptions, and performing a *concrete security analysis*. The next paragraph elaborates on this last point.

About ten years ago, Bellare and Rogaway (see [6]) developed the notion of "practice-oriented provable security". As a result of their work there was a greater awareness of the need to quantify the aforementioned 'additional effort' expended in a security reduction. The measure of this additional effort, obtained through a concrete (non-asymptotic) security analysis, could then be used to provide concrete recommendations about keylengths. If the additional effort is relatively small, then the reduction is said to be *tight* and the security of the protocol is closely linked to the hardness of the related computational problem. For example, if there is a tight reduction from the DLP in a group G to the breaking of a protocol, then the protocol attains an 80-bit security level if the parameters for G are selected so that the best algorithms known for solving the DLP in G take about 2^{80} operations. On the other hand, a non-tight reduction only

[10]The effort includes the number of times the adversary is invoked; here we are thinking of the adversary as a computer program.

assures us that the protocol has an 80-bit security level if the parameters for G are selected so that the best DLP solvers require significantly more than 2^{80} operations. The latter assurance can be achieved only by using significantly larger group parameters, a direct consequence of which is slower performance.

In the remainder of this section we provide a critique of various aspects of the HMQV security proofs given in [27].

3.1 Security model and definition

In a series of papers, Canetti and Krawczyk [14, 15, 16] carefully developed security models and definitions for key establishment. Their model corrected shortcomings in previous attempts (e.g., [9, 11, 7, 36]), and at present is widely accepted as being the 'right' one.

In the Canetti-Krawczyk model, the attacker controls all communications between honest parties. Through 'corrupt', 'session-key' and 'state-reveal' queries, she is even able to learn the static private keys of some parties, learn session keys, and learn certain secret information that is associated with a particular session. Her goal is to learn something about a session key that she cannot deduce by trivial means (e.g. through a session-key query). The thesis is that the Canetti-Krawczyk model captures all realistic attacks in an open network such as the Internet.

However, one should be careful not to be lulled into a false sense of security since there may be desirable security attributes which are not adequately addressed by the model.

As an example, consider the case of key-compromise impersonation (KCI) attacks. In these attacks, first studied by Just and Vaudenay [21], the adversary learns a party $\hat{A}$'s static private key and then tries to impersonate another honest party $\hat{B}$ to $\hat{A}$. Resistance to KCI attacks is important in situations where an attacker wishes to obtain some information possessed by $\hat{A}$, who is only willing to divulge this information to $\hat{B}$ (and where the attacker is not able to obtain $\hat{B}$'s static private key). The Canetti-Krawczyk model did not cover resistance to KCI attacks, and hence protocols proved secure in the model have to be examined on a case-by-case basis for KCI resistance. For example, the ISO [14] and SIGMA [25] protocols would appear to resist KCI attacks, while the Boyd-Mao-Paterson [12] protocol does not. This deficiency in the Canetti-Krawczyk model has apparently been addressed by Krawczyk [27].

Another deficiency of the Canetti-Krawczyk model is that it does not account for the security of a session for which a state-reveal query has been issued — only the security of *other* sessions is considered. This deficiency was recognized in [27] where the security of HMQV session keys that were derived using compromised ephemeral private keys is analyzed.

One should also be aware that the Canetti-Krawczyk model does not account for most kinds of secret-information leakage, e.g., partial information about the static private key that may be obtainable through side-channel attacks that analyze power consumption [24] or electromagnetic radiation [1]. An example of an attack on MQV (and HMQV-1) that is not considered in the Canetti-Krawczyk model is the side-channel attack of Leadbitter and Smart [29] (cf. §2.2.4) whereby a static private key b can be deduced from the least significant bits of some ephemeral private keys y and the least significant bits of the associated secret values $s = \lfloor y + eb \rfloor_q$.

Krawczyk [27] states that the potentially expensive procedure of public-key validation can be omitted if one is certain that private keys y are never revealed. However, as observed in §2.2.4, a small-subgroup attack that recovers the static private key b can be launched on two-pass and three-pass HMQV if the attacker can learn a small number of the least significant bits of each ephemeral secret y. Hence, omitting public-key validation makes two-pass and three-pass HMQV *more* vulnerable to side-channel attacks since an attacker only needs to obtain the least significant bits of y through a side channel. (The least significant bits of the associated secret values s do not have to be gathered using a side channel, but instead can be deduced using the method described in §2.2.4.) Thus the performance improvement that can be achieved by omitting public-key validation does not seem justifiable, especially in the case of elliptic curve groups where the cost of embedded public-key validation is negligible (cf. §2.3). It is interesting that our conclusion about the performance-security trade-off for public-key validation in two-pass and three-pass HMQV, which is based on concrete cryptanalysis and other practical considerations, is different from the conclusion derived from the proof-driven design methodology of [27].

The state-reveal query is indeed useful for modeling some realistic scenarios such as the loss of ephemeral private keys. However, in the Canetti-Krawczyk model each item that is the result of some intermediate calculation involving secret data must be specified as either being in highly secure memory (whose contents are only accessible via a corrupt query), or belonging to less-secure memory (whose contents are accessible via either a state-reveal query or a corrupt query). This distinction is potentially misleading because implementors may be lulled by a provable security result into thinking that there is no possible risk in performing certain calculations in less-secure memory, and this may increase their exposure to side-channel and other attacks.

3.2 The proofs

The HMQV security proof in [27] has two steps. First, an 'exponential challenge-response' signature scheme XCR is defined and proven secure under the computational Diffie-Hellman (CDH) assumption. Second, the security of XCR (actually a 'dual' version of XCR) is proven to imply the security of HMQV. The reduction in the first step is relatively straightforward, but the second reduction is extremely long and complicated.

In the XCR scheme, a verifier $\hat{A}$ selects $x \in_R [0, q-1]$ and sends the challenge $X = g^x$ and a message m to the signer $\hat{B}$. $\hat{B}$ responds by selecting $y \in_R [0, q-1]$ and sending the signature $(Y = g^y, \sigma = X^{y+eb})$ to $\hat{A}$ where $e = \overline{H}(Y, m)$ and (B, b) is $\hat{B}$'s static key pair. The signature is accepted by $\hat{A}$ provided that $Y \neq 0$ and $\sigma = (YB^e)^x$. XCR signatures are different from ordinary digital signatures. In particular, $\hat{A}$ cannot convince a third party that $\hat{B}$ generated a signature (Y, σ) for message m and challenge X because $\hat{A}$ could have generated this signature herself.

The HCR signature scheme [27] is a slight variant of XCR where the second signature component is hashed; i.e., $\hat{B}$'s HCR signature is $(Y, H(\sigma))$ instead of (Y, σ). The HCR scheme is proven secure under the Gap Diffie-Hellman (GDH) and 'KEA1' assumptions even if the attacker learns the ephemeral secrets y for each Y. The security of HCR is then proven to imply the security of HMQV even if the attacker is able to deduce ephemeral private keys x and y via state-reveal queries.

The security proofs for XCR and HCR both have flaws which arise because neither XCR nor HCR perform any validation of static or ephemeral public keys (other than the check $Y \neq 0$). We identify these flaws next.

3.2.1 Flaw in the XCR proof

In the reduction of CDH to XCR forgeries (Figure 3 of [26] and Figure 4 of [27]), the simulator $\mathcal{C}$ responds to an XCR-forger $\mathcal{F}$'s query (X, m) with $(\overline{Y} = g^s/B^e, \overline{\sigma} = X^s)$ where $s \in_R [0, q-1]$ and $e \in_R \{0,1\}^l$. However, this response is in general *not* a valid one if $X \notin G$. To see this, suppose that the order of X is $t \neq q$. Suppose also that the signer does not reduce $y + eb$ modulo q prior to computing X^{y+eb}. A legitimate response to a challenge (X, m) is of the form $(Y = g^y, \sigma)$ where $\log_X \sigma = \lfloor y + eb \rfloor_t$. But this relation is in general not satisfied by $(\overline{Y}, \overline{\sigma})$ since $y = \lfloor s - eb \rfloor_q$ and $\log_X \overline{\sigma} = \lfloor s \rfloor_t$, and

$$s \not\equiv \lfloor s - eb \rfloor_q + eb \pmod{t}$$

in general. Hence $\mathcal{C}$'s response to $\mathcal{F}$'s query is not correct, and consequently the subsequent behaviour of $\mathcal{F}$ cannot be predicted.

This particular flaw in the XCR proof is eliminated if the XCR protocol insists that the signer reduce $y + eb$ to its least non-negative residue modulo q before computing X^{y+eb}.

3.2.2 Flaw in the HCR proof

In the reduction of GDH to HCR forgeries (Figure 6 of [27]), the simulator $\mathcal{C}$ cannot respond to a challenge (X, m) if $X \notin G$. This is because it needs to decide whether $\mathrm{CDH}(X, B) = (\sigma/X^y)^{1/e}$ for certain $\sigma \in G$, but its DDH-oracle assumes that the inputs X, B are in G – indeed $\mathrm{CDH}(X, B)$ is not even defined if $X \notin G$ and $B \in G$.

3.2.3 Attacks on XCR and HCR

We observe that the XCR and HCR schemes are insecure if the signer does not verify that challenges X are in the main group G and if the attacker can learn ephemeral private keys y.[11]

In the case where $\hat{B}$ does not reduce exponents $y+eb$ prior to computing X^{y+eb}, the attacker would select a challenge $X \in G'$ of small order $t \neq q$. Since $\hat{B}$ responds with (Y, X^{y+eb}) or $(Y, H(X^{y+eb}))$, the attacker knowing y can compute $\lfloor b \rfloor_t$ by exhaustive search.

In the case where $\hat{B}$ first computes $s = \lfloor y + eb \rfloor_q$ and then X^s, the attacker would select several challenges $X \in G'$ of small order $t = 2^r$, and thereafter learn the r least significant bits of s from $\hat{B}$'s responses. If the attacker can also learn the r least

[11]This answers the question posed in Remark 4.5 of [27] about the security of XCR in the face of ephemeral private key revelations.

significant bits of the corresponding numbers y, then she can compute the $f/2$ most significant bits of b using the Leadbitter-Smart lattice attack.

Dan Brown [13] found the following 'large subgroup attack' on XCR in the case where exponents $y + eb$ are reduced modulo q. The attacker $\hat{A}$ selects an element $X \in G'$ of smooth order $t > q$; for DSA-like parameters, Banks and Shparlinski [5] showed that such an element X exists with non-negligible probability. Upon receiving X^s from $\hat{B}$, $\hat{A}$ can use the Pohlig-Hellman algorithm [32] to determine $s' = \lfloor s \rfloor_t$. Since $t > q$, we have $s' = s$. If $\hat{A}$ can also learn y through a state-reveal query, then $\hat{A}$ can easily compute $b = \lfloor (s - y)e^{-1} \rfloor_q$.

These particular attacks on XCR and HCR can all be defeated by performing validation on challenges X.

3.2.4 Security of HMQV-1

As noted in §2.3, the patched HMQV-1 protocols are resistant to our small-subgroup attacks because key validation is performed. Furthermore, the flaws in the HMQV proofs in [27] that were identified above are no longer pertinent. It remains to be seen whether the remainder of the proofs for two-pass HMQV-1 and three-pass HMQV-1 are correct, and whether new reductions can be found that are tight enough to provide meaningful assurances about the security of these protocols.

The UKS attacks described in §2.4 show that the one-pass HMQV-1 protocol must be modified if resistance to such attacks is desired.

3.3 Interpreting the proofs

Previous security proofs for key agreement protocols in the Canetti-Krawczyk model (e.g., see [14, 15, 16]) are all missing a concrete security analysis. This omission is partly because the analysis would be quite tedious since it would have to track several parameters including the number of honest parties, the number of sessions a party can be engaged in, and the number of hash function queries if the random oracle assumption is being invoked. It would appear that these reductions are not at all tight, therefore diminishing the practical value of the security proofs.

The HMQV security proof in [27] does not include a concrete security analysis. However, such an analysis is presented for the security proof of the XCR signature scheme. (Recall that this is the first step of the HMQV security proof.) The XCR security proof takes place in the random oracle model, and the forking lemma [33] is invoked. The result in [27], based on the analysis of the forking lemma in [33], is that the existence of an attacker who works in time at most T, presents at most Q queries to an l-bit hash function $\overline{H}$, and produces a forgery with probability at least ϵ, implies the existence of a CDH solver that runs in expected time

$$T' \leq 2^{17} QT/\epsilon \tag{3.1}$$

provided that $\epsilon \geq 7Q/2^l$.

The following analysis is from [23, Section 5.5]. Suppose for simplicity that $\epsilon \approx 1$. There is no justification for assuming that the number of hash function queries the forger makes is bounded by anything other than the forger's running time T. Thus (3.1) simplifies to $T' \leq 2^{17}T^2$. For the remaining analysis, we take $T' \approx 2^{17}T^2$. Since Pollard's rho method for finding discrete logarithms in the main group G has expected running time $\sqrt{q}$, the security reduction is only meaningful if $T' \approx 2^{17}T^2 \ll \sqrt{q}$. Suppose now that the main group G is an order-q subgroup of the multiplicative group $\mathbb{Z}_p^*$. Then for XCR to be assured an 80-bit security level one would have to select roughly a 354-bit q and a 7000-bit p (see [30]). Such parameter sizes would be too inefficient to be used in practice. On the other hand, if we insist on efficiency and use 1024-bit p and 160-bit q, then the security proof only guarantees that forgers whose running times are less than $2^{31.5}$ do not exist; this is a totally useless level of security.

If we analyze the extremely non-tight HMQV security proof, we find that the assurances it provides are even worse than those that follow from the non-tight XCR proof.

4 The MQV key agreement protocols

Section 3.2 of [27], entitled 'Insecurity of MQV', examined the MQV protocols in the Canetti-Krawczyk model to determine whether the protocols possess the security properties that were informally stated in [28]. Seven weaknesses and explicit attacks were identified to demonstrate that some of the most essential security properties were not satisfied in general. We briefly revisit the seven attacks and explain why they are irrelevant in practice. We begin by reviewing the MQV protocols.

For simplicity, we only consider the MQV protocols in the case where G' is the group of points on an elliptic curve and the cofactor h is small; some details are omitted for the sake of brevity. The (two-pass) MQV protocol as described in [28] differs from HMQV in the following ways.

(i) The certification authority must verify that static public keys are non-identity elements in the main group G.

(ii) The recipient of an ephemeral public key X must verify that $X \in G'$ and $X \neq 1$, check that $\sigma^h \neq 1$, and compute the session key as $K = H(\sigma^h)$.

(iii) The integers d and e are derived from the l least significant bits of the x-coordinates of X and Y respectively.

In the three-pass MQV protocol, the MAC tags transmitted are $\text{MAC}_{K_m}(\text{“2”}, \hat{B}, \hat{A}, Y, X)$ and $\text{MAC}_{K_m}(\text{“3”}, \hat{A}, \hat{B}, X, Y)$.

The seven weaknesses and explicit attacks identified in [26, 27] are the following:

(1) Resistance to basic impersonation attacks. The observation is made that since the calculation of $\sigma = g^{(x+da)(y+eb)}$ in MQV depends on the representation chosen for elements of G, some implementations of MQV may be weak because of a poor selection of representation. For example, if a representation is chosen where the l least significant bits of all x-coordinates of elliptic curve points are constant, then MQV falls to impersonation attacks – an attacker who selects x and sends $(\hat{A}, \hat{B}, g^x/g^{da})$ to $\hat{B}$, can easily compute the resulting session key $K = H((g^x)^{y+eb})$.

Of course, such contrived group representations would never arise in practice (and are certainly not allowed in standards such as [2, 3, 37] that specify the bit representations of group elements). The observation does illustrate that MQV cannot be proven secure for generic prime-order groups, but this can hardly be called an 'attack'.

(2) Prime-order testing and exponentiation performance. This is not an attack, but rather the observation that the validation of ephemeral public keys may require an expensive exponentiation.

(3) Validation of long-term public keys. This too is not an attack, but the observation that validation of static public keys imposes an extra requirement on certification authorities.

HMQV was designed to address the perceived weaknesses in MQV raised in the first three points. That is, HMQV was described using generic prime-order groups (and so it isn't clear whether checks $X \in G'$ have to be performed), and validation of ephemeral and static public keys was omitted. Ironically, these three changes are *precisely* what led to the attacks described in §2.2 on the HMQV protocols.

(4) Resistance to key-compromise impersonation attacks. Both MQV and HMQV are shown to fall to KCI attacks provided that the attacker, in addition to knowing $\hat{A}$'s static private key, can also learn the shared secret $\sigma_{\hat{B}}$ (by issuing a state-reveal query). This observation is used to justify the design principle of deriving the session key by hashing the shared secret (which both MQV[12] and HMQV do), and for advising implementors to carefully protect and delete ephemeral information such as the σ values (i.e., the attacker should not be able to get any σ values via state-reveal queries).

(5) Resistance to disclosure of Diffie-Hellman exponents. Another reason for protecting σ values is that both MQV and HMQV-1 fall to certain attacks if the adversary can obtain one of the ephemeral private keys x or y in addition to σ. No attacks are known on MQV or HMQV-1 if only ephemeral private keys (and not σ values) are divulged.

(6) Resistance to unknown-key share attacks. Kaliski [22] observed that the two-pass MQV protocol is vulnerable to a UKS attack. One countermeasure, which is mandated in NIST SP 800-56A [37], is to include the identities of parties in the key derivation function, i.e., $K = H(\sigma, \hat{A}, \hat{B})$. Another countermeasure that was noted in [28] is to use the three-pass MQV protocol. The attack is thwarted in this case because party $\hat{B}$ would send a tag $\text{MAC}_{K_m}(z, \hat{B}, \hat{C}, Y, X)$, whereas $\hat{A}$ would expect a tag $\text{MAC}_{K_m}(z', \hat{B}, \hat{A}, Y', X')$. The trivial observation made in [26, 27] is that if an adversary could learn the MAC key K_m via a state-reveal query, then she could compute the appropriate tags and complete Kaliski's UKS attack.

[12]MQV standards, including ANSI X9.63 [3] and NIST SP 800-56 [37], mandate the hashing of the shared secret to form the session key. The MQV paper [28] also permits a hash function to be used in key derivation. Page 130 of [28] states that the key derivation function does not have to be preimage resistant — this remark was made under the assumption that the bitlength l of the session key is at most half the bitlength of the group order q, in which case one would expect to try at least 2^l preimages before finding the shared secret $\sigma_{\hat{B}}$ that corresponds to a given session key K.

(7) Perfect forward secrecy. It was observed in [26, 27] that no two-pass key agreement protocol can achieve 'full' forward secrecy. This observation is thus not specific to MQV, and in any case was already well known (e.g. see [8, 35]).

Summary. Of the seven weaknesses and explicit attacks on the MQV protocols reported in [26, 27], the first three are non-attacks and in fact were incorrectly addressed in the HMQV protocols, leading to fatal flaws. Attacks (4), (5) and (6) rely on the adversary's ability to learn secret information (other than ephemeral private keys) through a state-reveal query; if this is a realistic threat in practice, then even HMQV-1 is insecure. Finally, attack (7) is a well-known observation about the limited form of perfect forward secrecy that can be attained in any two-pass protocol. We conclude that the analysis performed in [26, 27] did not uncover any new weaknesses in the MQV protocols. In particular, no attacks are known on the MQV protocols as standardized in NIST SP 800-56A [37].

5 Concluding remarks

The work of Canetti and Krawczyk [14, 15, 16, 25, 26] is widely acclaimed for its development of a formal model and definition of secure key exchange. However, the Canetti-Krawczyk model does have some deficiencies. On the one hand, an attack that is legitimate in the model may not be a realistic threat in practice. On the other hand, the model does not account for many realistic threats and often yields proofs that are lengthy, open to misinterpretation, and of questionable value in practice.

Researchers who use the Canetti-Krawczyk model to analyze the security of key establishment protocols should be careful not to rely exclusively on this analysis, and in particular should not abandon old-fashioned cryptanalysis and prudent security engineering practices. Standards committees and security engineers should not be mesmerized by claims of provable security, nor should they be intimidated by technical proofs.

Above all, the results of the present paper validate the main thesis of [23] that the field of provable security is as much an art as a science.

Acknowledgments. I would like to thank Dan Brown, Darrel Hankerson, Ann Hibner Koblitz, Neal Koblitz, and Scott Vanstone for their valuable comments on earlier drafts of this paper, and Phong Nguyen and Igor Shparlinski for answering my questions about the Hidden Number Problem.

References

[1] D. Agrawal, B. Archambeault, J. Rao and P. Rohatgi, *The EM side-channel(s)*. Cryptographic Hardware and Embedded Systems – CHES 2002, Lecture Notes in Computer Science 2523, pp. 29–45. Springer, Berlin, New York, 2002.

[2] ANSI X9.42, *Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography*. American National Standards Institute, 2003.

[3] ANSI X9.63, *Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography*. American National Standards Institute, 2001.

[4] A. Antipa, D. Brown, A. Menezes, R. Struik and S. Vanstone, *Validation of elliptic curve public keys*. Public Key Cryptography – PKC 2003, Lecture Notes in Computer Science 2567, pp. 211–223. Springer, Berlin, New York, 2003.

[5] W. Banks and I. Shparlinski, *Integers with a large smooth divisor*. Preprint, 2005.

[6] M. Bellare, *Practice-oriented provable-security*. Proceedings of the First International Workshop on Information Security – ISW '97, Lecture Notes in Computer Science 1396, pp. 781–793. Springer, Berlin, New York, 1998.

[7] M. Bellare, R. Canetti and H. Krawczyk, *A modular approach to the design and analysis of authentication and key exchange protocols*. Proceedings of the 30th Annual ACM Symposium on the Theory of Computing, 1998.

[8] M. Bellare, D. Pointcheval and P. Rogaway, *Authenticated key exchange secure against dictionary attacks*. Advances in Cryptology – EUROCRYPT 2000, Lecture Notes in Computer Science 1807, pp. 139–155. Springer, Berlin, New York, 2000.

[9] M. Bellare and P. Rogaway, *Entity authentication and key distribution*. Advances in Cryptology – CRYPTO '93, Lecture Notes in Computer Science 773, pp. 232–249. Springer, Berlin, New York, 1994.

[10] I. Biehl, B. Meyer and V. Müller, *Differential fault analysis on elliptic curve cryptosystems*. Advances in Cryptology – CRYPTO 2000, Lecture Notes in Computer Science 1880, pp. 131–146. Springer, Berlin, New York, 2000.

[11] S. Blake-Wilson, D. Johnson and A. Menezes, *Key agreement protocols and their security analysis*. Proceedings of the Sixth IMA International Conference on Cryptography and Coding, Lecture Notes in Computer Science 1355, pp. 30–45. Springer, Berlin, New York, 1997.

[12] C. Boyd, W. Mao and K. Paterson, *Key agreement using statically keyed authenticators*. Applied Cryptography and Network Security – ACNS 2004, Lecture Notes in Computer Science 3089, pp. 248–262. Springer, Berlin, New York, 2004.

[13] D. Brown, personal communication, July 2005.

[14] R. Canetti and H. Krawczyk, *Analysis of key-exchange protocols and their use for building secure channels*. Advances in Cryptology – EUROCRYPT 2001, Lecture Notes in Computer Science 2045, pp. 453–474. Springer, Berlin, New York, 2001. Full version available at http://eprint.iacr.org/2001/040/.

[15] R. Canetti and H. Krawczyk, *Universally composable notions of key exchange and secure channels*. Advances in Cryptology – EUROCRYPT 2002, Lecture Notes in Computer Science 2332, pp. 337–251. Springer, Berlin, New York, 2002. Full version available at http://eprint.iacr.org/2002/059/.

[16] R. Canetti and H. Krawczyk, *Security analysis of IKE's signature-based key-exchange protocol*. Advances in Cryptology – CRYPTO 2002. Lecture Notes in Computer Science 2442, pp. 1–4. Springer, Berlin, New York, 2002 3.161 Full version available at http://eprint.iacr.org/2002/120/.

[17] W. Diffie, P. van Oorschot and M. Wiener, *Authentication and authenticated key exchanges*. Designs, Codes and Cryptography 2 (1992), pp. 107–125.

[18] FIPS 186-2, *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication 186-2. National Institute of Standards and Technology, 2000.

[19] IEEE Std 1363-2000, *Standard Specifications for Public-Key Cryptography*, 2000.

[20] ISO/IEC 15946-3, *Information Technology – Security Techniques – Cryptographic Techniques Based on Elliptic Curves – Part 3: Key Establishment,* 2002.

[21] M. Just and S. Vaudenay, *Authenticated multi-party key agreement*. Advances in Cryptology – ASIACRYPT '96, Lecture Notes in Computer Science 1163, pp. 36–49. Springer, Berlin, New York, 1996.

[22] B. Kaliski, *An unknown key-share attack on the MQV key agreement protocol*. ACM Transactions on Information and System Security 4 (2001), pp. 275–288.

[23] N. Koblitz and A. Menezes, *Another look at 'provable security'*. Journal of Cryptology, to appear. Available at http://eprint.iacr.org/2004/152/.

[24] P. Kocher, J. Jaffe and B. Jun, *Differential power analysis*. Advances in Cryptology – CRYPTO '99, Lecture Notes in Computer Science 1666, pp. 388–397. Springer, Berlin, New York, 1999.

[25] H. Krawczyk, *SIGMA: The 'SiGn-and-MAc' approach to authenticated Diffie-Hellman and its use in the IKE protocols*. Advances in Cryptology – CRYPTO 2003, Lecture Notes in Computer Science 2729, pp. 400–425. Springer, Berlin, New York, 2003.

[26] H. Krawczyk, *HMQV: A high-performance secure Diffie-Hellman protocol*. Advances in Cryptology – CRYPTO 2005, Lecture Notes in Computer Science 3621, pp. 546–566. Springer, Berlin, New York, 2005.

[27] H. Krawczyk, *HMQV: A high-performance secure Diffie-Hellman protocol*. Full version of [26], available at http://eprint.iacr.org/2005/176/

[28] L. Law, A. Menezes, M. Qu, J. Solinas and S. Vanstone, *An efficient protocol for authenticated key agreement*. Designs, Codes and Cryptography 28 (2003), pp. 119–134.

[29] P. Leadbitter and N. Smart, *Analysis of the insecurity of ECMQV with partially known nonces*. Information Security – ISC 2003, Lecture Notes in Computer Science 2851, pp. 240–251. Springer, Berlin, New York, 2003.

[30] A. Lenstra and E. Verheul, *Selecting cryptographic key sizes*. Journal of Cryptology 14 (2001), pp. 255–293.

[31] P. Nguyen and I. Shparlinski, *The insecurity of the Digital Signature Algorithm with partially known nonces*. Journal of Cryptology 15 (2002), pp. 151–176.

[32] S. Pohlig and M. Hellman, *An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance*. IEEE Transactions on Information Theory 24 (1978), pp. 106–110.

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

[34] J. Pollard, *Monte Carlo methods for index computation mod p*. Mathematics of Computation 32 (1978), pp. 918–924.

[35] P. Rogaway, M. Bellare and D. Boneh, *Evaluation of security level of cryptography: ECMQVS (from SEC 1)*. CRYPTREC report, Information-technology Promotion Agency, Japan, January 2001. Available at http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1069_ks-ecmqv.pdf.

[36] V. Shoup, *On formal models for secure key exchange,* version 4, 1999. Available at http://shoup.net/papers/skey.pdf.

[37] SP 800-56A *Special Publication 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*. National Institute of Standards and Technology, March 2006.

[38] J. Stern, *Why provable security matters*. Advances in Cryptology – EUROCRYPT 2003, Lecture Notes in Computer Science 2656, pp. 449–461. Springer, Berlin, New York, 2003.

Received 13 May, 2006; revised 26 July, 2006

Author information

Alfred Menezes, Department of Combinatorics & Optimization, University of Waterloo, Canada.
Email: ajmeneze@uwaterloo.ca