Security of ECQV-Certified ECDSA Against Passive Adversaries

Daniel R. L. Brown*, Matthew J. Campagna* and Scott A. Vanstone*

December 22, 2009

Abstract

We show that the elliptic curve Qu-Vanstone implicit certificate scheme (ECQV), when composed with the Elliptic Curve Digital Signature Algorithm (ECDSA), is secure against passive adversaries under the combined assumption of the random oracle model and the generic group model. In contrast, we detail an attack on the composition of another implicit certificate scheme, proposed by Pintsov and Vanstone for Optimal Mail Certificates (OMC), and ECDSA. This composition attack forges an implicitly certified ECDSA signature, and is passive in the sense of needing no interaction with the signer or the certification authority. (Pintsov and Vanstone did not propose combining OMC with ECDSA.)

1 Introduction

A certification authority (CA) plays a central role in a secure public-key infrastructure as the root of trust. A CA binds an entity's identity to its public-key by the creation of certificates. These certificates are typically comprised of a data portion, containing an identity element and a public key, and a signature on that data produced by the CA. Assurances on the validity of the certificate is provided by verifying the signature on the data portion using the CA's public key. A valid signature on a certificate bears witness that the certificate has not been modified and the entity provided evidence to the CA that it had possession of the associated private key, such as by means of a signed certificate request, and is the identified entity according to the CA's enrollment policies. The entity's public key is explicitly contained in the certificate, and the signature on the certificate can be explicitly verified.

Implicit certificates are composed of a data portion containing an identity element, and a public key reconstruction portion. The public key of the entity identified is not explicitly contained in the certificate, and the validity of the certificate cannot be explicitly verified by using the CA's public key. The public key is reconstructed from the implicit certificate using the CA's public key. The validity of the certificate is verified by interaction with the entity in which evidence of knowledge of the private key associated with the reconstructed public key is provided. The entity's public key is computed from the certificate and the validity of the certificate is implied by proof of possession of the private key, and so they are called implicit certificates.

The security of an implicit certificate scheme requires that the reconstructed public key can only be successfully used by the entity identified in the certificate. Very much like traditional certificates, the named entity is not authenticated to a second party until they provide evidence that they have

*Certicom Research

possession of the private key. Implicit certificates are validated in the process of authenticating the named entity collapsing both computational results to a single decision point for which purpose the certificate is truly intended. By comparison, traditional certificates provide two decision points, first on the validity of the signature on the certificate, and second does the entity purporting to be the named entity have possession of the private key, and so can be authenticated. Both are equally susceptible to protocol failures due to attempts to authenticate without possession of the private key.

Pintsov and Vanstone [5] proposed an implicit certificate scheme, called the *Optimal Mail Certificate* (OMC) scheme. This scheme was specifically designed to be used with the Pintsov-Vanstone (PV) signature scheme giving partial message recovery. The PV signature scheme has since become standardized IEEE 1363A-2004 and in ISO standards and in X9-approved draft ANS X9.92-1.

Brown, Gallant and Vanstone [3] describe a modified version of the OMC implicit certificate scheme, which has become known as the elliptic curve Qu-Vanstone (ECQV) certificate scheme.[1] They give a definition of security for an implicit certificate scheme, and provide a proof of security for the scheme. Their security definition does not address the arbitrary composition of the scheme with other schemes. Rather, the threat model addresses an attacker who forges an implicit certificate together with the private key. They indicate that it may be possible for an attacker to launch an attack on the OMC scheme composed with some other scheme without explicitly needing the private key, although no specific example was given.

Qu [6] and Vanstone derived such an attack. Qu's description attacked the composition of OMC with Elliptic Curve Digital Signature Algorithm (ECDSA). The attack takes the CA's public key and generates an implicit certificate and an ECDSA signature, such that the signature is valid when verified with the public key reconstructed from the implicit certificate. The attack is generalized slightly here. The associated private key is not produced, so this is not an attack on the implicit certificate scheme under the definition of [3], *per se*.

The OMC implicit certificate scheme is not solely to blame for this attack. It is not clear that this attack works for other signature schemes, such as the PV signature scheme providing partial message recovery originally proposed to be combined with OMC [5]. Furthermore, the security proof for ECQV in [3] also applies to the OMC scheme. As noted above, it is not advertised in [5] that OMC is universally composable, it is made clear in [3] that ECQV might not be universally composable.

The other scheme affected by this attack, ECDSA is not usually advertised as being universally composable. More precisely, most treatments of ECDSA assume standard key generation, not the key generation assisted by CA an implicit certificates. When used in the standard manner, ECDSA is generally considered secure: nobody has published a forgery attack against ECDSA meeting the standard Goldwasser-Micali-Rivest definition of a forgery. Indeed, ECDSA also has security proofs [1, 2, 4] under various reasonable assumptions.

Therefore, the composition of two "provably secure" schemes, namely original OMC and ECDSA, results in an insecure scheme. This situation may be viewed as a specific limitation of the security definition for implicit certificates given in [3], or it may be viewed as a broader limitation of provable security, or it may be viewed as a need to formulate all security definitions according to the recently defined universal composability. Regardless, the caution in [3], that the security proof does not carry over to arbitrary composition, has to a degree borne out, so proved well worth heeding.

[1] This scheme is specified in draft Standard for Efficient Cryptography 4.

1.1 Our contribution

We show that the composition of ECQV with ECDSA is secure against a passive adversary under the combination of the random oracle model and the generic group model. We build to this result by revisiting the original OMC scheme. We reconstruct this attack on the composition of OMC with ECDSA, and demonstrate why this attack fails on the composition of ECQV with ECDSA.

1.2 Organization of this paper

The remainder of this paper is organized as follows. In Section 2, we provide a background into OMC, ECQV and ECDSA. In Section 3, we describe the attack on the composition of OMC with ECDSA and indicate why this attack fails on the composition of ECQV with ECDSA. In Section 4, we state and prove our main result that the composition of ECQV with ECDSA is secure against a passive adversary. Note that this proof assumes the random oracle model (for the hash function) and the generic group model for the elliptic curve group. In Appendix A, we review the generic group model. In Appendix B, we suggest some possible further research.

2 Background

2.1 Optimal Mail Certificates

We describe the original OMC, using the notation from [3]. We work in an abelian group in which discrete logarithm problem is believed to be difficult, such as a large prime order subgroup of an elliptic curve group. We will use additive notation, which is typical for elliptic curve cryptography. Let G be the fixed generator of the group, with prime order n.

A CA will choose a random integer $c \in (0, n)$ for a private key and compute a corresponding public key $C = cG$. If Bob wishes to request an implicit certificate, then he chooses a random integer $r \in (0, n)$, computes $R = rG$, and sends this as a *request* to the CA. Bob and the CA agree on an identity value I for Bob. The CA chooses a random integer $k \in (0, n)$ and computes *public key reconstruction data* value

$$P = R + kG.$$

Next, the CA computes an *implicit signature* value, also known as *private key reconstruction data* value,

$$s = k + H(P, I)c \bmod n,$$

where H is a suitable hash function. The CA sends

$$(P, I, s)$$

to Bob. Bob computes his private key:

$$b = r + s \bmod n.$$

Suppose that Alice and Bob wish to communicate. Alice will have already obtained an authentic copy of the CA's public key C. Next Alice can obtain Bob's implicit certificate (P, I) containing identity value I identifying Bob and the public key reconstruction data P allegedly belonging to Bob. Unlike a traditional certificate, an implicit certificate includes neither a signature from the

CA nor Bob's public key. Alice reconstructs Bob's public key from the implicit certificate (P, I) using the CA's public key C:

$$B = P + H(P, I)C. \tag{1}$$

After reconstructing Bob's public key, Alice proceeds to use B as if it were authentic. She relies on the fact that the CA's contribution to (1) is sufficient to ensure that only Bob knows the private key b. Anybody who demonstrates knowledge of b by virtue of a digital signature or decryption of an asymmetric ciphertext would convince Alice that he is Bob — or so Alice would think.

For completeness, we now prove that B is the public key corresponding to b, which justifies why at least Bob can use the implicit certificate:

$$\begin{aligned} B &= P + H(P, I)C \\ &= R + kG + H(P, I)cG \\ &= rG + (k + H(P, I)c)G \\ &= (r + s)G. \end{aligned}$$

This does not show that other parties cannot compute b. The results of [3] suggest that an adversary will not be able to compute b. However, as we shall see there are attacks possible not requiring computation of b.

2.2 ECQV

ECQV is similar to OMC and differs primarily in the implicit signature value and how the certified entity's keys are (re)constructed. We summarize these differences is Table 1.

	OMC	ECQV
CA Key	$C = cG$	$C = cG$
Request	$R = rG$	$R = rG$
Implicit cert	$P = R + kG$	$P = R + kG$
Implicit sig	$s = k + H(P, I)c$	$s = c + H(P, I)k$
Private key	$b = r + s$	$b = s + H(P, I)r$
Public key	$B = P + H(P, I)C$	$B = C + H(P, I)P$

Table 1: Comparison of the OMC and ECQV

2.3 ECDSA

We review ECDSA in the context of the implicit certificate schemes detailed above. Bob is now the signer with the key pair (b, B). To sign a message M, Bob chooses a random integer t computes $T = tG$. Bob then transforms T to compute an integer $u = \phi(T)$. In an elliptic curve group, this function ϕ is computed by taking the x-coordinate, converting it to an integer, and then reducing the integer modulo n. Next, Bob computes a second component of the signature as

$$v = t^{-1}(H(M) + ub) \bmod n$$

Bob's signature on message M is (u, v). Traditionally the notation for ECDSA would have Bob's signature as (r, s), but to avoid confusion that may result in re-using these variables we have modified the notation for ECDSA.

Alice verifies Bob's ECDSA signature (u, v) by executing the following computations,

$$\begin{aligned} w &= v^{-1} \bmod n, \\ W &= w(H(M)G + uB) \\ y &= \phi(W) \end{aligned} \tag{2}$$

If $y \equiv u \bmod n$ the signature is accepted, otherwise it is rejected. For completeness, we prove that Alice will at least accept Bob's signature.

$$\begin{aligned} W &= w(H(M)G + uB) \\ &= (v^{-1} \bmod n)(H(M)G + ubG) \\ &= (((t^{-1}(H(M) + ub))^{-1} \bmod n)(H(M) + ub)G \\ &= tG \\ &= T, \end{aligned}$$

so therefore, $y = \phi(W) = \phi(T) = u$. Obviously, this does not prove that *only Bob* can generate signatures. Nonetheless, ECDSA is believed to be secure, at least if Bob generated his own private key directly.

3 The Attack

We state the attack as described by Qu [6]. We will then expand the analysis to extend the solution space. The attack affords Eve, with access to the CA's public key C, to choose a message M and identity value I and efficiently find values

$$(P, u, v)$$

such that (u, v) is a valid signature on M by the party identified, say Bob, in the implicit certificate (P, I) purportedly certified by the CA. Eve can do this without interacting with Bob or the CA. To see this directly we take any integer $f \in (0, n)$ and compute,

$$\begin{aligned} P &= -\frac{H(M)}{\phi(fC)}G, \\ u &= \phi(fC), \end{aligned} \tag{3}$$

and

$$v = \frac{H(P, I)\phi(fC)}{f}. \tag{4}$$

To see that (P, I) and (u, v) form a valid OMC-certified ECDSA signature on the message M, first reconstruct the public key as

$$B = P + H(P, I)C = -\frac{H(M)}{\phi(fC)}G + H(P, I)C, \tag{5}$$

and next substitute this, (3) and (4) into the ECDSA verification equation (2) as

$$\begin{aligned} W &= (v^{-1} \bmod n)\,(H(M)G + uB) \\ &= \frac{f}{\phi(fC)H(P,I)}\left(H(M)G + \phi(fC)\left(-\frac{H(M)}{\phi(fC)}G + H(P,I)C\right)\right) \\ &= fC. \end{aligned}$$

Then $u = \phi(W)$, so the ECDSA signature is valid, according to the verification equations, under the reconstructed OMC public key.

Note that the private key b corresponding to the public key reconstructed in (5) is

$$b = -\frac{H(M)}{\phi(fC)} + H(P,I)c,$$

where c is the private key of the CA. In this equation, if b were known by the attacker, then c could easily be solved for. But finding the private key of the CA means being able to solve the ECDLP. So, therefore, in this attack, the private key b is never close to being known. In particular, the attack does not violate the definitions of [3], which only considers attacks in which the adversary gains actual knowledge of a private key.

This attack above is *detectable* in the following sense. Given an alleged OMC-certified ECDSA signature (P, u, v), test if the following equation holds:

$$P = H(M)uG. \tag{6}$$

Such detectability suggests a potential work-around to the attack: if (6) holds, then reject the OMC-certified signature, and otherwise, accept or reject the signature as described earlier.

We can expand the above attack to discover a wider set of forged OMC certified ECDSA signatures. This is accomplished by an attacker Eve with access to the CA's public key C, to choose message M, identity I, and $0 < d, f < n$ and compute (P, u, v);

$$P = dC - \tfrac{H(M)}{\phi(fC)}G$$
$$u = \phi(fC)$$

and,

$$v = \frac{\left(d + H\left(dC - \tfrac{H(M)}{\phi(fC)}G,\quad I\right)\right)\phi(fC)}{f}.$$

We argue that, unlike the original attack, this generalized attack is undetectable. The set of values (P, u, v) produced by our attack is identical to the set of legitimate signatures (P, u, v). Therefore our attack resists the work-arounds mentioned above. An undetectable forgery attack is more damaging because it can only be fixed by a complete replacement of the algorithm.

The special case of our attack with $d = 0$ is a natural one to discover,[2] especially if the approach taken to discover the attack is more by insight and intuition than by systematic analysis. Indeed, when the goal is to find any attack, it is natural to pick the first one that comes to mind. As we noted earlier, this special case of the attack, however, is detectable, so the generalized attack can be considered significant.

[2]The case $d = 0$ is what Qu [6] had described.

3.1 Expanded Analysis

The systematic analysis led to the generalized attack and the ability to understand that ECQV certified ECDSA is not susceptible to the same attack. We start this analysis with a review of the attack on the OMC-certified ECDSA with equation (2). Multiply by v, and substitute $B = P + H(P, I)C$ to get

$$vW = H(M)G + uP + uH(P, I)C, \tag{7}$$

where $\phi(W) = u$. To further simplify our notation we set $h = H(M)$ and $q = H(P, I)$ and re-write the above (7) as

$$vW - uP = hG + quC. \tag{8}$$

So for a fixed set of values h, G, C and I we need to find v, W, P such that (8) is true. We can express (8) using row and column vectors:

$$\begin{pmatrix} -u & v \end{pmatrix} \begin{pmatrix} P \\ W \end{pmatrix} = \begin{pmatrix} qu & h \end{pmatrix} \begin{pmatrix} C \\ G \end{pmatrix}.$$

In general, there will exist a 2×2 matrix relating P and W to C and G as follows

$$\begin{pmatrix} P \\ W \end{pmatrix} = \begin{pmatrix} d & e \\ f & g \end{pmatrix} \begin{pmatrix} C \\ G \end{pmatrix}.$$

We can substitute in for (P, W) and drop the (C, G) from the expression and consider

$$\begin{pmatrix} -u & v \end{pmatrix} \begin{pmatrix} d & e \\ f & g \end{pmatrix} = \begin{pmatrix} qu & h \end{pmatrix}. \tag{9}$$

We transpose (9) to express it in terms with matrices acting on column vectors:

$$\begin{pmatrix} d & f \\ e & g \end{pmatrix} \begin{pmatrix} -u \\ v \end{pmatrix} = \begin{pmatrix} qu \\ h \end{pmatrix}.$$

We can consider q as an arbitrary function of d and e, since $q = H(P, I) = H(dC + eG, I)$; which we can express as $q = Q(d, e)$. Similarly u can be considered as a arbitrary function of f and g, say $u = \phi(W) = \phi(fC + gG)$, which we can express as $u = U(f, g)$

A successful forgery needs to solve for the matrix $\left(\begin{smallmatrix} d & f \\ e & g \end{smallmatrix}\right)$ and the free variable v, and satisfy the non-linear equations $q = Q(d, e)$ and $u = U(f, g)$.

We start isolating for v by multiplying by the inverse matrix.

$$\begin{pmatrix} -u \\ v \end{pmatrix} = \frac{1}{dg - ef} \begin{pmatrix} g & -f \\ -e & d \end{pmatrix} \begin{pmatrix} qu \\ h \end{pmatrix} = \frac{1}{dg - ef} \begin{pmatrix} gqu - fh \\ -equ + dh \end{pmatrix}.$$

This gives us four equations in the five unknowns $\{d, e, f, g, v\}$, namely

$$-u = \frac{gqu - fh}{dg - ef}, \tag{10}$$

$$v = \frac{-equ + dh}{dg - ef}, \tag{11}$$

$$u = U(f, g), \tag{12}$$

$$q = Q(d, e). \tag{13}$$

Instead of finding all possible solutions, we consider those where the non-linear complications introduced by the functions in (12) and (13) can be eliminated. If we choose $g = 0$, equation (10) simplifies to $-u = \frac{h}{e}$, with no dependency on f or g. We can express

$$e = -\frac{h}{u} = -\frac{h}{U(f,g)} = \frac{H(M)}{\phi(fC)},$$

and so for $g = 0$, and any given f, we can solve for e. Further, substituting into (11), we see that for any given d and f we can compute

$$v = \frac{(d+q)u}{f}.$$

Now both of our two original equations will be satisfied, and variables d and f are still free.

In summary, the attacker Eve chooses arbitrary $0 < d, f < n$ and computes the undetectable forgery

$$P = dC - \tfrac{H(M)}{\phi(fC)}G,$$
$$u = \phi(fC),$$

and,

$$v = \frac{\left(d + H\left(dC - \tfrac{H(M)}{\phi(fC)}G, \quad I\right)\right)\phi(fC)}{f}.$$

3.2 Failure of the Attack on ECQV

This attack appears to fails when attempted against ECDSA composed with ECQV. To see this, we repeat the systematic analysis to see where it fails. This will, of course, not prove that ECDSA composed with ECQV is secure, but simply prove that a particular attack strategy fails.

Take the main verification equation (2) for ECDSA, multiply by v, and substitute the ECQV public key reconstruction equation $B = C + H(P, I)P$ to get

$$vW = H(M)G + uC + uH(P, I)P.$$

As before, we define $u = \phi(W)$. For a selected message and identity M and I finding (P, u, v) which makes the above equation hold will result in a valid ECDSA signature for ECQV certificate (P, I). Separating the known points C and G from the points for which we wish to solve, namely P and W, we get

$$vW - quP = hG + uC,$$

with the earlier convention $q = H(P, I)$. This results in a similarly derived matrix equation

$$\begin{pmatrix} d & f \\ e & g \end{pmatrix} \begin{pmatrix} -qu \\ v \end{pmatrix} = \begin{pmatrix} u \\ h \end{pmatrix},$$

together with the same pair of non-linear constraints

$$q = Q(d, e),$$
$$u = U(f, g),$$

where Q and U are as before. Once again we take the transpose and isolate for the free variable v.

$$\begin{pmatrix} -qu \\ v \end{pmatrix} = \frac{1}{dg - ef} \begin{pmatrix} gu - fh \\ -eu + dh \end{pmatrix}.$$

Here we encounter an obstacle in the form of the equation defined by the first row of the matrix equation, namely

$$-qu = \frac{gu - fh}{dg - ef}. \tag{14}$$

At this point in the attack on the original scheme, recall that we chose $g = 0$, which allowed us to solve for e in terms of d, f, h, because the equation became linear in e. But here any such choice for d, e, f, g leaves qu on the left hand side; and $qu = Q(d, e)U(f, g)$ is non-linear in each of d, e, f, g. There appears to be no simple trick to move this equation into a special case that is linear in some subset of the variables d, e, f, g. Being non-linear, we have no linear method of solving (14).

For convenience, we place the systems that we attempted to solve for OMC and ECQV composed with ECDSA side-by-side.

OMC	ECQV
$q = Q(d, e)$	$q = Q(d, e)$
$u = U(f, g)$	$u = U(f, g)$
$\begin{pmatrix} -u \\ v \end{pmatrix} = \frac{1}{dg - ef} \begin{pmatrix} gqu - fh \\ -equ + dh \end{pmatrix}$	$\begin{pmatrix} -qu \\ v \end{pmatrix} = \frac{1}{dg - ef} \begin{pmatrix} gu - fh \\ -eu + dh \end{pmatrix}$

(15)

The only difference in these two systems is that the position of q has shifted in (15). In the OMC scheme q is on the right side of the equation, which allowed the attack to succeed. In the ECQV scheme, the q is on the left side of the equation, which seems to be enough to make this composition attack fail.

4 Passive Security of the ECDSA Composed with ECQV

We now address the special case of a *passive adversary* that only obtains the public key C, but does not make any certificate requests and does not attempt to issue any certificates to or ask for signature from any user. Obviously, resisting attacks from passive adversaries is absolutely essential. That this attack uses a passive adversary suggests that it is nontrivial for an implicitly certified signature scheme to resist passive attacks.

The proof assumes that the adversary works in the combination of the random oracle model (for the hash function) and the generic group model (for the elliptic curve group). The random oracle model is a widely accepted assumption in building security proofs. The generic group model, while not as widely used, is an emerging assumption being accepted in constructing security proofs. A variant of the generic group model used is a simplification of [1], which is also described in § A. It should be emphasized that these models have their limitations.

Theorem 1. *There does not exist a passive adversary against ECDSA combined with ECQV that succeeds in the combined random oracle and generic group model.*

Proof. Consider the adversary's attempted implicitly certified forgery (P, I, M, u, v). Recall ϕ for ECDSA is almost invertible in the sense that we can easily find from u all points of the form $W \in \phi^{-1}(u)$. Without loss of generality, we can presume that validity of the forgery means that

$$vW = H(M)G + uC + uH(P, I)P. \tag{16}$$

Recall that in the generic model for groups, see §A or [1], there are two kinds of points in a generic group session: *independent* points and *dependent* points. Independent points are those that first appear as inputs to the generic group oracle.. The points G and C are presumed to be independent points. Other independent points may be selected by the adversary. Dependent points are those points that first appeared as outputs of the generic group oracle. By observing the queries to the oracle, every dependent point can be expressed as some integer combination of the independent points. The fundamental property of the generic group model is that provided that the number of queries is small enough, there is a essentially a one-to-one correspondence between the points and these integer combinations. The probability that two different integer combinations are equal as points is negligible. We assume henceforth that this negligible event has not happened, so each point has a unique representation as an integer combination. For any point P, we write $c(P)$ for its corresponding integer combination vector. By convention we take C to be the first independent point and G the second, (alphabetical order), so that $c(C) = (1, 0, 0, \dots)$ and $c(G) = (0, 1, 0, \dots)$.

In terms of integer combinations, (16) becomes

$$vc(W) = H(M)c(G) + uc(C) + uH(P, I)c(P). \tag{17}$$

We now look at the first two coordinates equation of this vector equation. Let

$$c(P) = (d, e, \dots), \tag{18}$$
$$c(W) = (f, g, \dots). \tag{19}$$

The actual points P and W could depend on other independent points, such as points that adversary selected and fed to the generic group oracle. Indeed, the point W could be independent, in which case $f = g = 0$ if it is distinct from C and G. Regardless of the other independent points, (17) implies $v \cdot (f, g) = (u, H(M)) + uH(P, I) \cdot (d, e)$, which is to say, the matrix equation

$$\begin{pmatrix} u \\ h \end{pmatrix} = \begin{pmatrix} d & f \\ e & g \end{pmatrix} \begin{pmatrix} -qu \\ v \end{pmatrix}, \tag{20}$$

where $h = H(M)$ and $q = H(P, I)$. Note that we are using the same variable names as in the description of the attack. Again we have some extra dependencies, which we write as

$$q = H(P, I) = H(dC + eG + \dots, I) = Q(d, e), \tag{21}$$
$$u = \phi(W) = \phi(fC + gG + \dots) = U(f, g), \tag{22}$$

where it should be noted that q may depend not only d and e but some other values as well. Similarly, u depends on f and g, but may also depend on some other values. Because we are working in the random oracle model, the function Q can be regarded as a random function of its inputs d and e.

We also claim that U can be regarded as random function of f and g. If W is a dependent point, then its representation is random, so the claim holds. If W is independent, then $(f, g) \in$

$\{(0,0),(0,1),(1,0)\}$. In the first two cases, we get $u = -dqu$, which implies $dQ(d,e) = -1$ or $u = 0$. But since Q is a random function of d, the probability that $dQ(d,e) = -1$ is negligible. Recall that $u = 0$ is not allowed in a valid signature. In the case $(f,g) = (1,0)$, we get that $u = \phi(W) = \phi(C)$ and $h = -equ$, which implies that $eQ(d,e) = -h/\phi(C)$, which occurs with negligible probability since h and Q are random values.

We have $h = H(M) = 0 \bmod n$ with probability about $1/n$, which is negligible. Henceforth we assume that $h \neq 0 \bmod n$. We also view h as a fixed constant with no dependency on the other integers variables d, e, f, g, v. To justify this in a strict sense, we choose h before any calls to the generic group oracle, then select a random index of hash query to produce output h. With probability $1/q_H$, where q_H is the number of hash queries, the adversary's message M will be the input to the hash query we selected. (Thus to model h as fixed we incur a cost in the effective probability of our reduction.)

All we need to show is that no matter what the choices for d, e, f, g, v, if q and u are selected at random, then (20) has negligible probability of holding. Actually, that is not quite enough, because f and g can depend on q (but not u because $u = U(f,g)$), and d and e can depend on u (but not on q because $q = Q(d,e)$). Therefore, let $d = D(u)$, and $e = E(u)$ and $f = F(q)$ and $g = G(q)$, where the functions D, E, F, G can be any functions. Because q and u do not depend on v, we can write $v = V(q,u)$. (We earlier argued that h could not depend on q or u, by invoking the random oracle model.) In this notation, all we need to show is that there is negligible chance that

$$\begin{pmatrix} u \\ h \end{pmatrix} = \begin{pmatrix} D(u) & F(q) \\ E(u) & G(q) \end{pmatrix} \begin{pmatrix} -qu \\ V(q,u) \end{pmatrix}, \tag{23}$$

where h is some given constant, q and u are random, and D, E, F, G, V are any functions whatsoever that the adversary may select. In actuality, this gives the adversary more power than he or she may really have. Consider that $q = Q(d,e) = Q(D(u), E(u))$ and $u = U(f,g) = U(F(q), G(q))$, where Q and U are random functions. It may be impossible for an adversary to find functions D, E, F, G with this property. Nevertheless, we will show that even if the adversary could find these functions D, E, F, G (say if Q and U were not truly random), then the adversary is still doomed to fail.

In the analysis of this attack, we applied the inverse of $\left(\begin{smallmatrix} D(u) & F(q) \\ E(u) & G(q) \end{smallmatrix}\right)$ to both sides. Here, however, we must be careful not to assume that the matrix is invertible. What we can do, instead, is to multiply both sides by the adjugate matrix $\left(\begin{smallmatrix} G(q) & -F(q) \\ -E(u) & D(u) \end{smallmatrix}\right)$, which gives:

$$\begin{aligned} \begin{pmatrix} G(q) & -F(q) \\ -E(u) & D(u) \end{pmatrix} \begin{pmatrix} u \\ h \end{pmatrix} &= \begin{pmatrix} G(q) & -F(q) \\ -E(u) & D(u) \end{pmatrix} \begin{pmatrix} D(u) & F(q) \\ E(u) & G(q) \end{pmatrix} \begin{pmatrix} -qu \\ V(q,u) \end{pmatrix} \\ &= (D(u)G(q) - E(u)F(q)) \begin{pmatrix} -qu \\ V(q,u) \end{pmatrix}. \end{aligned}$$

This translates to a pair of equations:

$$\begin{aligned} G(q)u - F(q)h + qu(D(u)G(q) - E(u)F(q)) &= 0 \\ D(u)h - E(u)u - V(q,u)(D(u)G(q) - E(u)F(q)) &= 0 \end{aligned} \tag{24}$$

While equality for these two equations is necessary for (23) it is not sufficient. In particular, if $F(q) = G(q) = 0$, then this pair of equation holds precisely when $D(u) = h^{-1}uE(u)$. In this case,

(23) becomes

$$\begin{pmatrix} u \\ h \end{pmatrix} = \begin{pmatrix} h^{-1}uE(u) & 0 \\ E(u) & 0 \end{pmatrix} \begin{pmatrix} -qu \\ V(q,u) \end{pmatrix} = \begin{pmatrix} -h^{-1}qu^2E(u) \\ -quE(u) \end{pmatrix},$$

which cannot hold because it implies that $h = -quE(u)$, which has different values for different q whereas h does not. (Recall that $h \neq 0$ because $h = H(M)$ is random. This implies $uE(u) \neq 0$, so that $quE(u)$ varies with q.) Therefore the adversary choice of functions $F(q) = G(q) = 0$ does not yield an implicitly certified forgery. Next, we show that (24) implies that $F(q) = G(q) = 0$.

Suppose (24) holds for random (q, u) with probability at least p. Choose random q, u, u'. Then with probability at least p^2, we have

$$\begin{aligned} G(q)u - F(q)h + qu(D(u)G(q) - E(u)F(q)) &= 0, \\ G(q)u' - F(q)h + qu'(D(u')G(q) - E(u')F(q)) &= 0. \end{aligned}$$

This a standard probability argument that can be proved by the Cauchy-Schwarz inequality, for example. We can view this as a system of two equations in two unknowns $F(q)$ and $G(q)$, which can be written in matrix form as

$$\begin{pmatrix} -(h + quE(u)) & u + quD(u) \\ -(h + qu'E(u')) & u' + qu'D(u') \end{pmatrix} \begin{pmatrix} F(q) \\ G(q) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

If the matrix is invertible then $F(q) = G(q) = 0$ as desired. If the matrix is non-invertible, then the determinant $\Delta(q, u, u')$ is zero. This means that

$$\begin{aligned} 0 &= \Delta(q, u, u') \\ &= -(u' + qu'D(u'))(h + quE(u)) + (u + quD(u))(h + qu'E(u')) \\ &= h(u - u') + hq(uD(u) - u'D(u')) \\ &\qquad - quu'(E(u) - E(u')) + q^2uu'(D(u)E(u') - D(u')E(u)) \\ &= A(u, u') + B(u, u')q + C(u, u')q^2, \end{aligned} \tag{25}$$

letting $A(u, u') = h(u - u')$ and so on. The probability that $A(u, u') = 0$ is $1/n$, hence negligible. Therefore, except with non-negligible probability, the determinant $\Delta(q, u, u')$ is a non-zero polynomial polynomial of q, with degree at most two. It follows that for any pair (u, u') there are at most two values of q making $\Delta(q, u, u') = 0$. The probability that $\Delta(q, u, u') = 0$ is therefore at most $3/n$, which is negligible.

Therefore, except with negligible probability, we have $F(q) = G(q) = 0$, which implies there is no solution. □

As noted earlier, we should wonder why this proof would fail if applied to OMC. The logic of the proof goes through with OMC, albeit with slightly different arithmetic, up to the calculation of the determinant, which now has the form

$$\Delta'(q, u, u') = (A(u, u') - B_2(u, u'))q + (B_1(u, u') + C(u, u'))$$

where $B_1 - B_2 = B$ in (25). This is a polynomial in q, of degree at most one, but we cannot rule out the possibility that both of its coefficients are always zero. Indeed, the choice of function $E(u) = h/u$ makes both coefficients zero, regardless of the function F.

5 Conclusion

We proved, under fairly reasonable assumptions, that ECQV-certified ECDSA resists attacks from an adversary who is passive in the sense of only using the CA's public key, and not otherwise interacting with the CA or users. We also described Qu's attack which shows that OMC-certified ECDSA does not resist such passive attacks.

Acknowledgments

As noted earlier in this paper, Minghua Qu contributed to finding the attack on the composition of ECDSA with OMC.

References

[1] D. R. L. Brown. Generic groups, collision resistance, and ECDSA. *Designs, Codes and Cryptography*, 35:119–152, 2005. http://eprint.iacr.org/2002/026.

[2] D. R. L. Brown. On the provable security of ECDSA. In I. F. Blake, G. Seroussi, and N. P. Smart, editors, *Advances in Elliptic Curve Cryptography*, volume 317 of *London Mathematical Society Lecture Note Series*, pages 21–40. Cambridge University Press, 2005.

[3] D. R. L. Brown, R. P. Gallant, and S. A. Vanstone. Provably secure implicit certificate schemes. In P. F. Syverson, editor, *Financial Cryptography — FC 2001*, volume 2339 of *Lecture Notes in Computer Science*, pages 156–165. Springer, Feb. 2001. http://www.cacr.math.uwaterloo.ca/techreports/2000/corr2000-55.ps.

[4] P. Paillier and D. Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In B. Roy, editor, *Advances in Cryptology — ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 1–20. International Association for Cryptologic Research, Springer, Dec. 2005.

[5] L. A. Pintsov and S. A. Vanstone. Postal revenue collection in the digital age. In Y. Frankel, editor, *Financial Cryptography — FC 2000*, volume 1962 of *Lecture Notes in Computer Science*, pages 105–120. Springer, Feb. 2001.

[6] M. Qu. Attack on ECDSA with original optimal mail certificates. Personal Communication, 2000.

A Generic Group Model

The generic group model is one way to restrict the set of algorithms considered (including adversaries) from exploiting the internal structure of a group. The model does this by only providing oracle access to the group operations. Nechaev first introduced a generic group model, but did not equip the group elements with representations (thus excluding memory-efficient algorithms like the combination Floyd's cycle-finding to Pollard's rho algorithm). Shoup added the concept of random representations, and this is now the usual definition taken for the generic group model. Brown

[1] added to Shoup's model an ability for the algorithm to add arbitrary representations (which the oracle assigns to random group values), in order to better model the ability to sample random elements in the elliptic curve groups. As one would expect, [1] proves that this gives algorithms little extra power.

Furthermore [1] adds to the model a hint command, which is an extra oracle operation that outputs pairs of values in the same space as the group elements values (not the representations). The hint command models an active adversaries ability to obtain ECDSA forgeries. It is perhaps surprising that hint commands do not help algorithms compute discrete logarithms any faster [1].

The version of the generic group model used here is a slight simplification of that defined in [1]. The simplification is that we do not need to include a hint command, because the passive adversaries only the public key of the CA, which is a elliptic curve point.

We can therefore summarize what need of the generic group model from [1]. An algorithm working in the generic group model has access to an oracle G. At any time the oracle maintains a state which is a list of pairs $((A_1, z_1), \ldots, (A_i, z_i))$. The value i is the index and equals the number of queries made to the generic group oracle so far ($i = 0$ corresponds to an empty list). Here the z_i are elements of the cyclic group $\mathbb{Z}_n$ and A_i are some bit string representations of these points. The values z_i are kept secret from the algorithm working in the generic group model, and the values A_i are made available to an algorithm working in the generic group model.

The queries to the generic group oracle fall into two types of two commands: *subtract* and *push*.

- A subtract command can only be invoked when $i \geq 2$. It has no arguments. The index is incremented to $i + 1$. A new state entry $z_{i+1} = z_i - z_{i-1} \bmod n$ is computed and assigned. If $z_{i+1} = z_j$ for some $1 \leq j \leq i$, then the assignment $A_{i+1} = A_j$ is made. If $z_{i+1} \notin \{z_1, \ldots, z_i\}$, then A_{i+1} is selected uniformly at random from those possible group representations distinct from $\{A_1, \ldots, A_n\}$.

- A push command can be invoked at any index value. It has one argument, A_{i+1} is a possible group representation. As the notation indicated this is included as a state entry. If $A_{i+1} = A_j$ for some $1 \leq j \leq i$, then $z_{i+1} = z_j$ is assigned. If $A_{i+1} \notin \{A_1, \ldots, A_i\}$, then z_{i+1} is chosen uniformly at random from $\mathbb{Z}_n \setminus \{z_1, \ldots, z_i\}$.

We think of representations A_i as the public values elements in the groups, and the values z_i as their corresponding private values. More concretely, we may think of z_i as the discrete logarithm of A_i to some base.

When analyzing algorithms that work in the generic group model, the following definitions will be helpful.

- An element A_i is *independent* if it is both
  - from a push command,
  - $A_i \neq A_j$ for all $j < i$.

  Any other element A_i is *dependent*, that is, if it is from subtract command or a push of previous element.

- For any element A_i we define an integer sequence $c(A_i)$ as follows.

- If A_i is independent and is the k^{th} independent element ranked by index i, then $c(A_i)$ is the sequence that is all 0 except for a 1 in position k.
- If A_i is dependent and from a push command with $A_i = A_j$ for $j < i$, then $c(A_i) = c(A_j)$.
- If A_i is dependent and from a subtract command, then $c(A_i) = c(A_{i-1}) - c(A_{i-2})$.

- A *coincidence* has occurred at index i, when for some $j < i$, we have
  - $c(A_i) \neq c(A_j)$,
  - $A_i = A_j$.

A lemma from [1] says that the probability of a coincidence occurring is at most about i^{-2}, no matter what strategy employed by the algorithm interacting with the generic group oracle. For the purposes of this paper, we will use this result as follows. If the adversary working in the generic group model is efficient, then it has negligible chance of finding a collision.

B Further Research Suggestions

Some major improvement to the main result would be

- Formally define, and prove, security against more active adversaries. This paper has only focused on the most basic but crucial type of security. Resistance to active adversaries is important too, though, so it would be more assuring to find security proofs against such adversaries. The main result of this paper does not rule out such adversaries.

- Weaken the set of the assumptions made in the security result. The combination of the random oracle model and generic group model is a very powerful model, and as such very strong assumption. For example, perhaps the generic group model could be replaced by the assumption that elliptic curve discrete logarithms are intractable. Or, the random oracle model could be replaced by the assumption that the hash functions are collision resistant.

- Simplify the current proof. There are many variables and several special cases to consider in the current proof. Arguably, a more complicated security proof provides less assurance than a simple one. Furthermore, a simpler proof is easier to verify. If perhaps the composition of ECQV with ECDSA is so fundamentally complex that a simpler proof is not forthcoming, then perhaps some expositional simplification can be done, such as breaking the proof into various lemmata.

- Generalize the result to other implicit certificate schemes and other signature schemes, or even key agreement schemes. Furthermore, more clearly identify the conditions under which an attack like this is possible and when it is not. Ultimately, perhaps a form of universal composability result may be found. If this is not possible, we suggest that a *relativized* universal composability result may be possible, which could take the following form. If ECQV composed some application scheme resists passive attacks, then it further resists active attacks.