RKHD ElGamal signing and 1-way sums

Daniel R. L. Brown*

October 16, 2017

Abstract

An ECDSA modification with signing equation $s = rk + hd$ has the properties that the signer avoids modular inversion and that passive universal forgery is equivalent to inverting a sum of two functions with freely independent inputs.

Let $\sigma : s \mapsto sG$ and $\rho : R \mapsto -rR$ where r is an integer representation of the point R. The free sum of ρ and σ is $\nu : (R, s) \mapsto \rho(R) + \sigma(s)$. A RKHD signature (R, s) verifies if and only if $\nu(R, s) = hQ$, where h is the hash of the message and Q is the public key. So RKHD security relies upon, among other things, the assumption that free sum ν is 1-way (or unforgoable, to be precise).

Other free sums are 1-way under plausible assumptions: elliptic curve discrete logs, integer factoring, and secure small-key Wegman–Carter–Shoup authentication. Yet other free sums of 1-way functions (integer-factoring based) fail to be 1-way. The ease with which these free sums arise hints at the ease determining RKHD security.

RKHD signatures are very similar to ECGDSA (an elliptic curve version Agnew–Mullin–Vanstone signatures): variable-G forgers of the two schemes are algorithmically equivalent. But ECGDSA requires the signer to do one modular inversion, a small implementation security risk.

1 RKHD signing and verifying

A RKHD signature (R, s) is valid if

$$sG = rR + hQ, \tag{1}$$

*danibrown@blackberry.com

under the notation:

$\mathbb{E}$	finite cyclic subgroup of elliptic curve (with additive notation),
G	fixed generator point in $\mathbb{E}$,
n	order of G (a prime integer with $nG = 0$ and $\lvert\mathbb{E}\rvert = n$),
d	secret signing key (an integer),
Q	public verifying key (a point $Q = dG$),
h	hash value of the message to be signed (an integer),
k	message-unique secret (an integer),
R	first signature component (a point $R = kG$),
r	integer representative of R (obtained from conversion), and
s	second signature component,

where the signer can compute s using the equation:

$$s = rk + hd \bmod n. \tag{2}$$

All integer arithmetic will be done modulo n, unless otherwise noted. In particular, $\mathbb{Z}$ is used denote $\mathbb{Z}/n\mathbb{Z}$ with representatives $\{0, \ldots, n-1\}$. Hence $\mathbb{Z}$ becomes a field, and division is generally defined.

1.1 Implementation issues

Security warnings[1]:

- The signer must reduce $rk+hd$ modulo n. If not, then the secret signing key d will leak after one signature is generated. Indeed, suppose that $u = rk + hd$ as an unreduced integer (without reduction modulo n): then
$$d = u/h \bmod r. \tag{3}$$

- The values k should be indistinguishable from independent and uniformly random integer in the interval $[0, n-1]$. Leaked (or biased) bits of k will leak bits of d, under known lattice attacks.

- The values k should appear to be independent (although repeating k for a common h is okay).

[1] The usual warnings, which apply to ECDSA and many similar variants.

- The values k_i generated in multiple signatures (R_i, s_i) should have no adversarially determinable linear relationship, $\sum_i c_i k_i = 0$, because, if so, the signer's secret key can almost always be computed as:

$$d = \frac{\sum_i c_i s_i / r_i}{\sum_i c_i h_i / r_i} \bmod n. \tag{4}$$

- An especially important case of the restriction above, the same value k must be never be used with two different values of h. Indeed, if $s = rk + hd$ and $s' = rk + h'd$, then $d = (s - s')/(h - h') \bmod n$.

Minor efficiencies:

- Inversion mod n can be avoided by both signer[2] and verifier.
- Reduction mod n can be avoided by verifier (if desired).
- The verifier can, as an optional optimization method, use an alternative verification equation: $vsG = vrR + vhQ$, choosing v such that $vr \bmod n$ and $vh \bmod n$ are not too far above $\sqrt{n}$. (Finding v has cost similar to the Euclidean algorithm, using an algorithm dating back to Gauss.) The modified equation speeds up multiplying R and Q. The verifier can use also pre-computations for G.
- The signer can pre-compute kG shortly in advance of receiving h. The pair (k, kG) is then known as a coupon. Pre-computing the coupon reduce the latency between receiving a message to sign and generating its signature. A risk of coupons is that k might exposed in the interim rest period between message input and signature generation.
- The signer can generate k as a deterministic but pseudorandom function of (d, h). This is not vulnerable to (4) because the repeated instances k for h correspond to repeated equations and variables and have no net effect on the linear system for determining d. This method precludes the use of a coupon.

[2]The signer's avoiding of modular inversion is also a security advantage, because it is reduces the amount of side-channels.

2 Forgery by inverting a sum of functions

Define functions:

$$\rho : R \mapsto -rR, \tag{5}$$
$$\sigma : s \mapsto sG, \tag{6}$$
$$\nu : (R, s) \mapsto \rho(R) + \sigma(s). \tag{7}$$

If $\nu(R, s) = hQ$, then (R, s) will pass the verifying equation for public key Q and message hash value h.

Inverting ν at hQ is therefore equivalent to a passive universal forger of RKHD signatures.[3]

3 Free sums

Let $f : X \to Z$ and $g : Y \to Z$ be functions with a common codomain Z. Furthermore, suppose that Z is equipped with a binary operation written $+$. Usually, we suppose that $+$ makes Z into an abelian group. The **free sum**[4] of f and g is the function

$$f + g : X \times Y \to Z : (x, y) \mapsto f(x) + g(y). \tag{8}$$

In particular, the function $\nu : \mathbb{E} \times \mathbb{Z} \to \mathbb{E}$ is the free sum of the functions $\rho : \mathbb{E} \to \mathbb{E}$ and $\sigma : \mathbb{Z} \to \mathbb{E}$.

Given some function $f : X \to Z$, a **forgoer** or **inverter**[5] of f is any function $i : Z \to X$ such that $f(i(z)) = z$ with high probability given uniformly random $z \in Z$.

We could also call i a probable pre-inverse, or a preimage-finder, of f.

For brevity, f^{-1}, in this report, indicates[6] a potential inverter.

[3] By standard Pointcheval–Stern signer-simulating arguments, active (chosen-message) universal forgery of random-oracle-hash RKHD is equivalent to inverting ν too.

[4] Please excuse the neologism.

[5] Inverter is not an ideal name, due to its ambiguity.

[6] Standard math notation writes f^{-1} for the unique inverse, if it exists, and otherwise $f^{-1}(z)$ is a set of all i with $f(i) = z$. The notation f^{-1} in this report is to suggest an approximation to first meaning, a true inverse, rather than the second meaning. See §A for more discussion.

If no efficient[7] inverter for f exists, then f is said to be **unforgoable**, or a bit more concisely, but less precisely[8], **1-way**.

Because RKHD signatures rely on a 1-way free sum to avoid severe forgery, this report studies the issue 1-way free sums in greater generality (beyond RKHD).

3.1 Some free sums

A brief survey of some 1-way (and one non-1-way) free sums follows.

3.1.1 A 1-way free-sum of 1-way functions

Lemma 1. *Let $f = g = \sigma$, where $\sigma : s \mapsto sG$ as before. Assume that σ is unforgoable (the standard hard ECDLP assumption). Then $f + g$ is unforgoable. In particular, an $f + g$ inverter i is equivalent to a discrete log solver l (an inverter of σ).*

Proof. Essentially, the equivalence is due to the linearity of σ.

Suppose $i : \mathbb{E} \to \mathbb{Z} \times \mathbb{Z}$ is an $f + g$ inverter. Let $p : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z} : (x, y) \mapsto x+y$. Then $l = p \circ i$ is an σ inverter, and thus discrete log solver. Conversely, suppose that $l : \mathbb{E} \to \mathbb{Z}$ is a discrete log solver, and $r : \mathbb{E} \to \mathbb{Z}$ is any function. Then $i : P \mapsto (l(P) - r(P), r(P))$ is an $f + g$ inverter. □

More generally, consider the following.

Lemma 2. *Let $\eta : s \to sH$. Free sum $\sigma + \eta$ is also 1-way (if σ is).*

Proof. Take uniformly random $Z \in \mathbb{E}$ and compute $\sigma^{-1}(Z)$ as follows. Choose random integers r_1, r_2 and compute $R_i = r_i Z$. Compute $(\sigma + \eta)^{-1}(R_i) = (x_i, y_i)$. Then $\sigma^{-1}(Z) = (x_1 y_2 - x_2 y_1)/(r_1 y_2 - r_2 y_1) \bmod n$. □

3.1.2 A non-1-way free sum of 1-way functions

Begin with the famous result:

[7] A function is efficient if it has an efficient evaluation algorithm.

[8] Note: for example, constant functions are 1-way. The Handbook of Applied Cryptography recommends against the phrase "1-way" for just such reasons, but I will use it anyway.

Lemma 3 (Rabin)**.** *Let n be the product of two large primes, an RSA modulus. Let $f : x \mapsto x^2 \bmod n$. Inverting f is finding square roots mod n, which is equivalent to factoring n.*

So, f is 1-way. The free sum of f with a copy of itself is not 1-way:

Lemma 4. *Let $g = f$. The free sum $f + g$ is not 1-way.*

Proof. Free sum $f + g$ has the following inverter i. Given input z to i, first find the smallest prime p with $p = z \bmod n$ and $p = 1 \bmod 4$.

Next, apply Cornacchia's algorithm to p, which gives (x, y) such that $x^2 + y^2 = p$. (Recall that Cornacchia's algorithm is efficient.)

The inversion is verified as follows:

$$(f + g)(i(z)) = (f + g)(x, y) = x^2 + y^2 = p = z \bmod n, \tag{9}$$

so i has found a pre-image of f. □

3.1.3 A 1-way free sum of non-1-way functions

Let L be the set of logarithms[9] of a large set of large primes. Let $L+L$ be set of sums of two elements in L, (so the $L+L$ is the set of logarithms of a set of RSA public keys). Let L^+ be a version of L in which each logarithm appears very many times (far more than $|L|$ times). (In order to be set, instead of a multi-set, allow each copy of a logarithm to vary by a negligible amount.)

Let $Z = L \cup (L + L)$, and let $X = L^+ \cup (L + L)$. Let $f : X \to Z$ be the function maps each copy in L^+ to its original value in L, and maps each element in $L + L$ to itself.

Lemma 5. *Then f is not 1-way.*

Proof. Compute $i(z)$ as follows. If e^z is not prime, then $z \in L + L$, and let $i(z) = z$. If e^z is prime, then $z \in L$. Choose one of the copies z^+ of z and let $i(z) = z^+$. □

Define a binary operator $+$ on Z to be the usual real number addition if both arguments are in the subset L of Z. Otherwise, just set $a + b$ to be some arbitrary element of L, such as the smallest element.

Lemma 6. *If $g = f$ and integer factorization is difficult, then $f+g$ is 1-way.*

[9] Store logarithms of integers with sufficient precision to recover the original integer

Proof. Consider $z \in Z = L \cup (L + L)$ chosen uniformly at random. With overwhelming probability, $z \in L + L$, because $L + L$ has nearly $|L|$ times as many elements as L. In this case, integer e^z is a product of two random primes from a large set of large primes. Finding the factors of e^z should be difficult.

But if i is an efficient pre-inverse of $f + g$, then this means that for $i(z) = (x, y)$, we expect that $(f+g)(x, y) = z$. But $(f+g)(x, y) = f(x)+g(y)$. But this means $e^{f(x)}$ and $e^{g(y)}$ is the prime factorization of e^z, which should be difficult to find. □

3.1.4 Small-key Wegman–Carter–Shoup: a 1-way free sum

Work-in-progress.

Small-key Wegman–Carter–Shoup symmetric authentication can be described as a 1-way free sum using a little notation-reversal trick.

The secret key in WCS is a pair a functions $(x : M \to Z, y : N \to Z)$, where M is a message space, N a finite nonce space, and Z a group. Then WCS authentication with secret key (x, y) is a function $(m, n) \mapsto x(m)+y(n)$. An attacker gets to see $(m, n, z) = (m, n, x(m) + y(n))$ but should not be able to determine[10] (x, y).

The notation reversal trick is to write $x(m) + y(n) = m(x) + n(y) = (m + n)(x, y)$. Under this new notation, m and n are now functions. So, we have fixed m and n. This situation is realistic in the sense an WCS attacker gets to see m and n and z. The attacker seeing the message m and nonce n, is perfectly able to take our reversed-notation view and evaluate m and n as functions of the secrets x and y.

So, let X is the set of all functions $x : M \to Z$, and, similarly, $Y = \{y : N \to Z\}$.

By small-key WCS, we assume that the size of $X \times Y$ is smaller than Z. For a concrete example, suppose that both (x, y) are derived from the same 128-bit key, and Z is the space of 256-bit strings. The small-key is probably not realistic for most WCS implementations. Nonetheless, it remains plausible (to me) that WCS is secure with such small keys.[11]

Suppose that i is an inverter of the free sum $m+n$. Upon seeing (m, n, z) where $z = (m + n)(x, y)$ is generated by a WCS authenticator with secret

[10] among many other things that should be hard for a WCS adversary...

[11] My rationale: instead of calling this small-key WCS, I could have called it big-tag WCS, which sounds quite secure, doesn't it? Actually, well, no.

key (x, y), then i can be applied to z. The result is some (x', y') with $(m + n)(x', y') = z$.

In small-key WCS, there cannot be so many (x, y) mapping to z under $(m + n)$, so we expect that $(x', y') = (x, y)$. In this even, we have recovered the actual WCS secret key, which is essentially a total break: future messages can be forged.

In sum, small-key WCS also seems to rely on a 1-way free sum.

3.1.5 A 1-way free sum: constant plus 1-way

Lemma 7. *If f is constant and g is 1-way, then the free sum $f + g$ is 1-way.*

Proof. Suppose otherwise: that $f + g$ is not 1-way for the given f and g. Then $g^{-1}(z) = ((f + g)^{-1}(z - f(x)))_2$, a contradiction. □

Similar results may hold if f is sufficiently compressive.

Note: the function ρ in RKHD cannot be replaced by a constant.

3.2 Requirements for 1-way free sums

This section lists some requirements for free sum $f + g$ to be 1-way.

3.2.1 In a group, the summands must be 1-way

Lemma 8. *If Z is a group and $f + g$ is 1-way, then, both f and g must be 1-way.*

Proof. Suppose that one of functions, say f, is easy-to-invert, with inverter f^{-1}. Then the function defined by $(f + g)^{-1}(z) = (f^{-1}(z - g(y)), y)$ has the same success rate at inverting $f + g$ as the function f^{-1} has at inverting f. □

Note: the example in §3.1.3 does not contradict this statement, because Z in that example is not a group: subtraction in Z is not available.

3.3 The colliding free-sum inverter

Assume Z is a group and let $N = |Z|$. Select $\sqrt{N}$ random values x_i and $\sqrt{N}$ random values y_j. Compute and store all $f(x_i)$. Sort the resulting list.

Compute each $z - g(y_j)$, and do a binary search to find if it is in the sorted list of $f(x_i)$. If $z - g(y_j) = f(x_i)$ for some i, j, then output $(x, y) = (x_i, y_j)$.

The total number of steps is this algorithm at most $2\sqrt{N}\log(N)$, although it can stop earlier.

This algorithm succeeds as an $f+g$ inverter if the probability p that some uniformly random x_i and y_j will have $(f + g)(x_i, y_j) = z$ for a uniformly random z. Let us call p the **colliding probability** of f and g.

By linearity of expectation, the expected number of hits in the algorithm above is at least Np. So, as long as $p \geq 1/N$, then we can reasonably expect this algorithm to work.

So, we say that f and g **collide** if the colliding probability is at least $1/N$ (approximately).

Colliding between f and g can easily fail in some cases (for example, if $f(X)$ and $g(Y)$ are disjoint). In other cases, colliding between f and g can be easily verified (for example, if f and g are both bijections).

Note: Shanks' baby-step-giant-step algorithm for inverting σ is actually a special case of this colliding algorithm.

4 Recommendation

As far as I know, RKHD signatures have not been standardized or deployed. But the very similar ECGDSA signatures have been.

Somewhat surprisingly to me, as far as I know, 1-way free sums have not been studied nor has the reliance of ECGDSA on 1-way free sums been noted.

I recommend further study of 1-way free sums, specifically those involved in RKHD. The small security advantage[12] of RKHD over ECGDSA is not enough to abandon ECGDSA implementations[13]. This minor advantage may be enough to justify preferring RKHD over ECGDSA in new deployments (unless they need backwards interoperability).

It seems imprudent to use RKHD signatures as a replacement for similar signatures schemes, like ECDSA, EdDSA, ECKDSA or ECGOST, at least not without a more thorough study of 1-way free sums.

Using RKHD in conjunction with other signature schemes seems fairly harmless, except for the extra cost to the signer and verifier, provided the

[12] Avoiding a signer modular inversion.

[13] If already used, then they have already done the modular inversion.

keys are fully separated and so on.

A 1-way formalisms and jargon

Let $f : A \to B$ be a function which is potentially "1-way" in a sense to be formally defined below.

The intuitive idea behind "1-way function" concerns the existence of a function $i : B \to A$ that has some of the properties expected of a true inverse f^{-1} of a bijective function f. We will list some properties soon, but first we discuss more general issues of efficiency and probability.

Efficiency, existence, and evidence-based assumptions Being 1-way encompasses a notion about computational difficulty, so it may not be sufficient to look only at the existence of inverse-like functions i. Indeed a a perfect inverse $i = f^{-1}$ of bijective function i exists by definition, and 1-way bijections, such as that associated with discrete logs and block ciphers, are important for cryptography. So, for f to 1-way, we require that there must not exist inverse-like function i that can evaluated by an efficient algorithm.

In many cases of a specific function, the non-existence of suitable i cannot be proved. Instead, the non-existence of suitable i is merely assumed, but usally based on the ample evidence of non-discovery of suitable i despite extensive effort.

Success rate We define a general relation between two functions $f, g : A \to B$ (that share the same domain and codomain). We say f **approximates** g, and write $f \approx g$, with a **success rate** s if the probability that $f(a) = g(a)$ is at least s, where a is chosen uniformly at random from A.

Probabilistic to deterministic algorithms In this formalism (so far), we will consider a single function $i : B \to A$, which is deterministic. Usually cryptography deals with probabilistic algorithms (especially as adversaries). So suppose $I : R \times B \to A$ is a deterministic function that models a probabilistic algorithm by encoding the random choices of the algorithm into a formalized finite set R, such that uniformly random r selected from R models the random choices of the probabilitic algorithm. (If it is impossible to do this with a finite set R – which seems unlikely to me – then we make R into a probability space and adjust as needed.) Now let $I_r : B \to A : b \mapsto I(r, b)$,

which is a deterministic function of the form i considere previously. We define the success rate of the I to be the success of I_r averaged over uniformly random r. If I has success rate s, there exists r such that I_r, now a deterministic function, has success rate r. (Of course, it may perhaps be difficult to find r, but since the definitions of 1-way concern non-existence of suitable i, this may be adeqaute.) So, this approach, more or less reduces probabilistic algorithm setting to deterministic function setting (up to finding r).

Inverse-like properties We will define several different **flavors** of 1-way properties of f. To do this, we just need to fix a little more notation about involving functions.

For any functions $f : A \to B$ and $g : B \to C$, let gf be the composite function $gf : A \to C : a \mapsto g(f(a))$. (Note many authors write as $g \circ f$ instead.) Note that $f \approx g$ implies $hf \approx hg$.

For any set A, where $1_A : A \to A : a \mapsto a$ for the identity function on the set A. If the domain and codomain are clear from context, then we can omit the subscript A. For example, if $f : A \to A$ and we write $f \approx 1$, then by definition of the symbol $\approx$, it clear that 1 means 1_A.

Recall that we seek a function i that acts like f^{-1} would. Now we have all the tools to formally specify this.

The three main inverse-like properties of a function $i : B \to A$ are:

$$fi \approx 1, \tag{10}$$

$$if \approx 1, \tag{11}$$

$$fif \approx f. \tag{12}$$

If f is actually bijective and $i = f^{-1}$, then each approximation above holds with success rate 1 (so, as identical functions). So, each propertiy holds for a true inverse. If they hold for another function i, then we say that i has an inverse-like property.

In this report, we have used the first property, $fi \approx 1$ in defining what it means to be 1-way. In other words, we have said that f is 1-way if there exists no efficient, high success rate function i such that $fi \approx 1$. (Note $1 = 1_B$ in this context.)

Many other similar inverse-like properties can be specified, as follows. Each property is an approximation between two compositions (like the three properties above). Each has domain-codomain pair which is one of (A, A), (A, B), (B, B) or (B, A). Each composite function is composite of some

non-negative integer number of functions (i or f, alternating in some order). For example, this report's main inverse-like property, $fi \approx 1$, is given by domain-codomain pair (B, B) and composite numbers $\{0, 2\}$.

Among such generalites, the three approximations given above are probably the most important, because of the fact that i appears only once (among other reasons).

The reader might notice that some inverse-like properties of i can imply others. Especially important implications include the following.

- If f is bijective (as is the function $f = \sigma : s \mapsto sG$), then a true inverse function f^{-1} exists. We may compose f^{-1} with some approximation properites. Then the three main approximations for i above just reduce to the simpler approximation approximation $i \approx f^{-1}$, showing that they are all equivalent.

- If $if \approx 1_A$, then $fif \approx f$. (Just apply f.)

Jargon What to call the flavors? We cannot name them all, so only try to name the most fundamental. Begin by naming the functions i that satisfy the three main approximations above.

- If $fi \approx 1_B$, then i is a **forgoer** for f.

- If $if \approx 1_A$, then i is a **reverser** for f.

- If $fif \approx f$, then i is a **conjoiner** for f.

Through most of this report a forgoer has been called an inverter. This is because the other flavors were not needed, and inverter better conveys the similarity to an inverse than forgoer. Anyway, 1-way functions, as defined in main parts of this report, could be more precisely called unforgoable functions.

Note that a success rate 1 reverser for f exists if and only if f is injective. Similarly, a success rate 1 forgoer for f exists if and only if f is surjective. More generally, a reverser has success rate at most $|B|/|A|$, and forgoer has success rate at most $|A|/|B|$. By contrast, a success rate 1 conjoiner exists for every f, although it not necessarily an efficient conjoiner. The existence observations above ignore issues of efficiency: usually the terms reverser, forgoer and conjoiner refer only to efficient functions.

Much literature in cryptology uses the term **preimage finder**, either for our forgoer or our conjoiner.

Note that, in previous work on MQV, I called these three types of i by different names: **inverter**, **reverter**, **opener**, respectively. As far as I know, nobody picked up my past terminology, so I do not feel too obliged to stick with it consistently. More technically, in the MQV work, I found bounds on the various i types given the sizes of A and B, and reductions between the notions relative to other types of f-attack tasks, such as distinguishers and predicters.

For the sake of an example, consider the degenerate case of a constant function $f : A \to B$, where both A and B are large sets. On one hand, f is 1-way in the sense of being unforgoable and irreversible (information-theoretically!). On the other hand, a constant f is not 1-way in the sense that it is easily conjoinable: every function $i : B \to A$ is a success rate 1 conjoiner!

In more generic cryptology, it is common to view constant functions as an uninteresting case, and not worthy of the designation of 1-way. In order not to water down the meaning of "1-way", it seems reasonable to preclude constant function from the special class of "1-way" functions. Defining "1-way" to be our unconjoinable (plus perhaps with additional coniditions) precludes constant functions.

Becuase of this ambiguity, this report re-defines 1-way, and this section introduces an alternative term unforgoable with a more specifc meaning.

Why this reports forgoes unconjoinability In the specifics of this report, the unconjoinability flavor of 1-way seems to complicate some issues. For example, the summands in an unconjoinable free sum do not have to be unconjoinable (consider a constant plus an unconjoinable function). It seems too awkward to state some of results about free sums with special exemptions tailored pre-existing notions of 1-way like unconjoinability.

A forgoer of the free sum $\nu = \rho + \sigma$ corresponds exactly to a passive universal forger of RKHD as noted in the body of the report.

Note that the function $f = \nu : \mathbb{E} \times \mathbb{Z} \to \mathbb{E}$ has a uniformity property: given uniformly input, the output is uniformly random. This uniformity property is inherited from $+$ defining a group and $\sigma : \mathbb{Z} \to \mathbb{E}$ being a bijection. Any function with such a uniformity property has the property any conjoiner is a forgoer, and vice versa. (Check this.)

So, actually, the unconjoinability definition of 1-way would have worked just fine for defining RKDH security. Where unconjoinability was lacking in this report was in generalization to other free sums, where the unforgoability definition is more natural.

B Previous work and references

To be completed.

B.1 Agnew, Mullin and Vanstone

In 1990 (Electronic Letters), Agnew, Mullin and Vanstone (AMV) proposed a variant of ElGamal signatures (in a non-ECC) setting. The AMV signing equation could be written as $h = rk+ds \bmod n$, in the notation of this report. Agnew, Mullin and Vanstone point out that the function $x \mapsto x^x \bmod p$ seems to be 1-way. This function corresonds to our function ρ.

By writing $\sigma'(s) = sQ$, we can also show that the EC version of the AMV signature has a passive universal forgery equivalent to the inversion of a free sum, with AMV signature (R, s) being valid if and only if $\rho(R)+\sigma'(s) = hG$.

Note that AMV signatures require a one-time modular inversion of the secret signing key. This modular inversion is a small cost, but might also represent a small security risk.

B.2 ECGDSA

An EC version of AMV signatures have been standardized in the form of ECGDSA. All the comments above about AMV apply.

B.3 Nyberg and Rueppel

In Eurocrypt 1994, Nyberg–Rueppel proposed six variants of ElGamal with message recovery. One of these six seems to correpsond the RKHD variant (upon removal of message recovery), but they do not seem focus on the RKHD variant, nor on the security of free sums.

B.4 Handbook of Applied Cryptography

Menezes, van Oorschot and Vanstone, in the Handbook of Applied Cryptography adapt Nyberg and Rueppel's six variants of ElGamal to the traditional setting with appendix signatures (no message recovery).

B.5 Horster, Peterson and Michels

In 1994 (ACM), Horster, Petersen and Michels, survey 13,000 variants of ElGamal signatures. RKHD signatures, in their classification scheme, are type EG I, and number 4, and so on. They do not emphasize anything like free sums (or products). However, they claim an equivalence in security between number 2 and number 4 signatures, which in our notation, corresponds to swapping the roles of G and Q. They did not provide details of this equivalence.

I speculate that "equivalence" Horster, Petersen and Michels were referring to is the following. Suppose that the forger's input consists not only of the public key Q, but also the generator G. This forger is a free-generator forger. Then an free-generator RKDH forger is clearly equivalent to a free-generator ECGDSA-forger, just by swapping the inputs G and Q.

In this report, I assume that G is fixed, and therefore not an input to the forger. This formalism models the common practice that G is fixed, so that the public key Q can be as small as possible. Formally, it also allows for non-intuitive possibility that forger specific to G. I have so far failed[14] to find an equivalent between RKHD and ECGDSA in this fixed-G setting. As noted earlier, the security is still quite similar without such a literal reductive equivalence. In both cases, passive universal forgery is equivalent to inverting a free sum.

B.6 Schnorr

Schnorr (1994?) signatures are similiar to RKHD signatures but with $\rho'(R) = R$ instead of ρ and h' being the hash of R and m instead of h. The natural combination[15] of Schnorr with RKHD would use the 1-way ρ and h', would

[14] For example, the naive approach of dividing the verifying equation by d partly does the trick but it modifies the function ρ.

[15] The naive combination, taking ρ' and h, is faster, but not secure, because, for example, ρ' is not 1-way.

probably have all the security properties of Schnorr signatures (proofs in models), but I have so far failed to an equivalentce between passive universal forgery and a 1-way free sum, so its security properties seem (to me) different from RKHD.

Note also that Schnorr and RKHD share the feature of avoiding any modular inversion.

B.7 EdDSA

The recent EdDSA signature scheme is an EC variant of the Schnorr signatures, in which the ephemeral secret k is a deterministic function of the message and hash. Most of the comments above about Schnorr signatures should apply.