A Security Analysis of the NIST SP 800-90 Elliptic Curve Random Number Generator

Daniel R. L. Brown[1] and Kristian Gjøsteen[2]

[1] Certicom Research, dbrown@certicom.com
[2] Department of Mathematical Sciences, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway, kristian.gjosteen@math.ntnu.no

Abstract. An elliptic curve random number generator (ECRNG) has been approved in a NIST standard and proposed for ANSI and SECG draft standards. This paper proves that, if three conjectures are true, then the ECRNG is secure. The three conjectures are hardness of the elliptic curve decisional Diffie-Hellman problem and the hardness of two newer problems, the x-logarithm problem and the truncated point problem. The x-logarithm problem is shown to be hard if the decisional Diffie-Hellman problem is hard, although the reduction is not tight. The truncated point problem is shown to be solvable when the minimum amount of bits allowed in NIST standards are truncated, thereby making it insecure for applications such as stream ciphers. Nevertheless, it is argued that for nonce and key generation this distinguishability is harmless.

Key Words: Random Number Generation, Elliptic Curve Cryptography.

1 Introduction

Certain random number generator (RNG) algorithms, such as the Blum-Micali [1] and Kaliski [2] generators, have been proven secure — assuming the conjectured hardness of associated number-theoretic problems. Recently, a new random number generator has been undergoing standardization (see [3–6]). In this paper, this new generator is called the *Elliptic Curve Random Number Generator (ECRNG)*. Like Kaliski's generator, the ECRNG is based on elliptic curves and is adapted from the Blum-Micali generator. Compared to many other number-theoretic RNGs, the ECRNG is considerably more efficient, because it outputs more bits per number-theoretic operation. The ECRNG is different from the Kaliski RNG in two major respects:

– The ECRNG uses ordinary elliptic curves, not the supersingular elliptic curves of Kaliski RNG. Supersingular curves can make the state transition function a permutation, which is advantageous for making the Blum-Micali proof work, but is disadvantageous because of the Menezes-Okamoto-Vanstone (MOV) attack, which requires a larger curve to make the proof give a useful assurance.[3] For ordinary curves, the state transition function is many-to-one, often two-to-one. This paper adapts the Blum-Micali proof by introducing the x-logarithm problem to overcome the obstacle introduced by state transition function not being a permutation.

- The ECRNG produces at each state update an output with almost as many bits as in the x-coordinate of an elliptic curve point, whereas the Kaliski ECRNG outputs just a single bit. Therefore the ECRNG is considerably more efficient than the Kaliski RNG if operated over the same elliptic curve. The Kaliski RNG outputs a single bit that is a hardcore predicate for the elliptic curve discrete logarithm problem (ECDLP). The ECRNG output function essentially uses a conjectured hardcore function of the ECDLP. The basis of this conjecture is the elliptic curve DDH problem, and the truncated point problem (TPP), defined below.

This paper proves that ECRNG is secure if the following problems are hard:

- The elliptic curve version of the well known *decisional Diffie-Hellman problem* (DDH). This is now widely accepted for certain groups. These groups include most small cofactor order elliptic curve groups defined over finite fields, such as the NIST curves. Boneh [7] gives an excellent survey about the DDH problem.
- The *x-logarithm problem* (XLP): a new problem, which is, given an elliptic curve point, determine whether its discrete logarithm is congruent to the x-coordinate of an elliptic curve point. This problem is discussed further in §5. We provide some evidence that the XLP problem is almost as hard as the DDH problem. The evidence takes the form of loose reduction between a related problem, AXLP (defined below), and the DDH problem.
- The *truncated point problem* (TPP): a new problem, which is, given a bit string of a certain length, determine whether it is obtained by truncating the x-coordinate of a random elliptic curve point. The TPP problem concerns extraction of pseudorandom bits from random elliptic curve points. El Mahassni and Shparlinski [8] give some results about extraction of pseudorandom bits from elliptic curve points. Gürel [9] also gives some results, although with fewer bits extracted than in the ECRNG. We discuss the TPP problem in §7. We find that if too few bits are truncated, then the result is distinguishable from a random bit string. Schoenmakers and Sidorenko [10] independently found a similar result.

Naor and Reingold [11] constructed pseudorandom functions secure as the hardness of the DDH problem, while Gertner and Malkin constructed such a pseudorandom number generator based on the same assumption. Farashahi, Schoenmakers and Sidorenko [12] recently constructed pseudorandom number generators following a modified version of the ECRNG, which are secure as DDH over certain groups.

[3] This is not to say that MOV attack could be applied against the Kaliski RNG for smaller sized curves.

This paper does not attempt to analyze the various issues surrounding entropy of the secret state of the RNG. Prediction resistance is the ability of RNG to add additional entropy into the secret state to recover completely from a circumstance where an adversary has information about the previous state. Initialization and prediction resistance are general RNG issues, and indeed the standards specifying the ECRNG do not treat the ECRNG especially different from other RNGs with respect initialization and prediction resistance. This paper deliberately restricts itself to ECRNG specific issues.

2 The Elliptic Curve Random Number Generator

Let $\mathbb{F}_q$ be a finite field with q elements. An elliptic curve E over $\mathbb{F}_q$ is defined by a nonsingular cubic polynomial in two variables x and y with coefficients in $\mathbb{F}_q$. This paper considers only cubics in a specially reduced Weierstrass form

$$E(x, y) = y^2 + cxy - (x^3 + ax^{1+c} + b) = 0 \tag{1}$$

where c is 0 if q is odd and 1 if q is odd, since these are most often used in cryptography, and particularly in the ECRNG. We define the rational points of the curve to be

$$E(\mathbb{F}_q) = \{(x, y) \in \mathbb{F}_q^2 : E(x, y) = 0\} \cup \{0\}. \tag{2}$$

An addition law is defined on $E(\mathbb{F}_q)$ using the well-known chord-and-tangent law. For example, $(u, v) + (x, y) = (w, z)$ is computed as follows. Form a line through (u, v) and (x, y), which intersects the curve $E(x, y) = 0$ in three points, namely (u, v), (x, y) and some third point $(w, -z)$, which defines the desired sum by negating the y-coordinate.

In the ECRNG, and in elliptic curve cryptography more generally, one defines some base point P on the curve. One assumes that P has prime order n in the elliptic curve group, so that $nP = 0$. Generally, the number of points in $E(\mathbb{F}_q)$ is hn, where the cofactor h is usually quite small, typically with $h \in \{1, 2, 4\}$. We say that a point Q is *valid* if it is an additive multiple of P. We will generally only consider valid points in this paper, so when we say a random point, we mean a random valid point.

The ECRNG maintains a state, which is an integer $s_i \in [0, \max\{q-1, n-1\}]$. The iteration index i increments upon each output point of the ECRNG. The ECRNG is intended to be initialized by choosing the initial state s_0 uniformly at random from $[0, n - 1]$.

For a point $P = (x, y) \in E(\mathbb{F}_q)$, we write $x(P) = \bar{x}$, where $\bar{x} \in \mathbb{Z}$ is obtained by taking the bit representation of the $x \in \mathbb{F}_q$ and considering this is as the bit representation of an integer. When q is prime, we essentially have $\bar{x} = x$, but when q is not a prime, the value of $\bar{x}$ depends on the representation used for the finite field $\mathbb{F}_q$. (We may arbitrarily define $x(0) = 0$, but we will encounter this case negligibly often in our analysis.) Therefore, to fully specify the ECRNG, one needs to define a field representation, because the function $x(\cdot)$ has an important rôle in the ECRNG, as we see below.

The ECRNG has another initialization parameter, which is a point Q. The point should ideally be chosen at random, preferably verifiably at random, such as by deriving it from the output of secure hash function or block cipher.

When the state is s_i, the (raw) output point is defined as

$$R_i = s_i Q. \tag{3}$$

The actual output of the ECRNG applies further processing to R_i. The final output is $r_i = t(x(R_i))$, where t is a function that truncates certain bits from the bit string representation of an elliptic curve point. The purpose of t is to convert the x-coordinate of a pseudorandom EC point to a pseudorandom bit string.

After generating an output point R_i, the state is updated as

$$s_{i+1} = x(s_i P). \tag{4}$$

It is convenient to adopt the following notation. We define the *prestate* at iteration $i+1$ as $S_{i+1} = s_i P$. Note that $s_{i+1} = x(S_{i+1})$. We may think of the prestate being updated as

$$S_{i+2} = x(S_{i+1})P. \tag{5}$$

The following notation for the ECRNG will be convenient. Let s_0 be the initial state. We define $g_m(Q, s_0) = (R_0, R_1, \ldots, R_m)$ inductively by

$$g_0(Q, s_0) = (s_0 Q) \text{ and } g_m(Q, s_0) = (s_0 Q, g_{m-1}(Q, x(s_0 P))),$$

where we use the convention that a comma indicates concatenation of point sequences. We shall continue to use this convention for the remainder of the paper.

3 Lemmas on Indistinguishability

Random variables X and Y are *computationally indistinguishable* if, given a sample value u that has probability $\frac{1}{2}$ of coming from X and $\frac{1}{2}$ from Y, an adversary cannot distinguish , with a feasible cost of computation and reasonable success rate, whether u comes from X or from Y. Pseudorandomness is indistinguishability from a uniform (equiprobable) distribution. Indistinguishability is a well known notion in cryptology, but for completeness, this section introduces some general notation and lemmas on indistinguishability that are convenient for the proofs of the main theorems.

We write $X \sim Y$ to indicate that random variables X and Y are indistinguishable. Where needed, we write $X \overset{\sigma}{\sim} Y$ to quantify the indistinguishability by some parameters σ, such as success rate or computational cost. We write $X \cong Y$ when random variables X and Y are identically distributed. Obviously, $X \cong Y$ implies $X \sim Y$.

Intuitively, one expects indistinguishability ($\sim$) to be an equivalence relation. Certainly, $\sim$ is reflexive and symmetric, and more interestingly, it is transitive ([13, Ex. 27], for example). This is such a fundamental point it is worth repeating here.

Lemma 1. *If $X \sim Y$ and $Y \sim Z$, then $X \sim Z$.*

A second lemma, which one also intuitively expects, makes proofs cleaner through separating complicated constructions from indistinguishability.

Lemma 2. *If f is an efficiently computable function, and $X \sim Y$, then $f(X) \sim f(Y)$.*

It is worth noting that the converse to this lemma does not necessarily hold: generally, $f(X) \sim f(Y)$ does not imply $X \sim Y$. A constant function f is a trivial counterexample. Nontrivial counterexamples exist too, such as f being a bijection whose inverse is not efficiently computable.

A third lemma, which one again intuitively expects, allows one to analyze distributions by analyzing independent components.

Lemma 3. *If $X \sim Y$ and $W \sim Z$, and X and W are independent variables, as are Y and Z, and X and Z can be efficiently sampled, then $(X, W) \sim (Y, Z)$.*

This lemma also applies under our notational convention that if X and Y are sequences, then (X, Y) is their concatenation.

4 The Decisional Diffie-Hellman Problem

The *decisional Diffie-Hellman problem (DDH)* for a given elliptic curve E and a base point P is to distinguish between a triple $(Q, R, S) = (qP, rP, qrP)$ and a triple $(Q, R, Z) = (qP, rP, zP)$ where q, r, z are integer random variables uniformly distributed in the interval $[0, n-1]$. (Note this q is not to be confused with the field order.) The triple (Q, R, S) is often called a Diffie-Hellman triple. For certain elliptic curves, it is conjectured that the DDH problem is hard.

Conjecture 1. If q, r and z are independent random integers uniformly distributed in $[0, n-1]$, then $(qP, rP, qrP) \sim (qP, rP, zP)$.

This, if true, provides a nontrivial counterexample to the converse to Lemma 2 because random variables $X = (q, r, qr)$ and $Y = (q, r, z)$ are distinguishable, but if one applies the function f defined by $f(x, y, z) = (xP, yP, zP)$, then the conjecture says $f(X) \sim f(Y)$.

The conjectured hardness of the DDH problem, for certain groups, is widely believed among cryptologists. One should be aware that for certain elliptic curves, however, there are efficiently computable so-called pairings that can be used to distinguish Diffie-Hellman triples. Pairings exist for all elliptic curves, but only for a very few are they known to be efficiently computable. For most elliptic curves, one can verify that the known pairings are extremely inefficient and infeasible to use in practice. This has been confirmed for most of the NIST recommended elliptic curves.

5 The x-Logarithm Problem

The *x-Logarithm Problem (XLP)* for elliptic curve $E(\mathbb{F}_q)$ and base point P is to distinguish between dP and xP where: d is an integer chosen uniformly at random in $[0, n-1]$; and $x = x(Z)$ for a point Z chosen uniformly at random in $E(\mathbb{F}_q)$. We conjecture that the x-logarithm problem is hard for most elliptic curves:

Conjecture 2. If d and z are random integers uniformly distributed in the interval $[0, n-1]$, then $dP \sim x(zP)P$.

Now d and $x = x(zP)$ are generally distinguishable. Firstly, known tests on x quickly determine whether there exists a $y \in \mathbb{F}_q$ such that $(x, y) \in E(\mathbb{F}_q)$. Secondly, when the cofactor $h > 1$, we expect to have $x > n$ for at least about half of the x-coordinates x of random points, whereas for d, we always have $d < n$. Therefore, this conjecture, if true, gives another counterexample to the converse of Lemma 2.

An intuitive reason for the plausibility of the XLP conjecture is that given public key dP, one expects that nothing substantial is leaked about the private key d. This intuition derives from the conjectured hardness of the elliptic curve discrete logarithm problem (ECDLP). However, a formal argument that the ability to determine whether $dP = x(zP)P$ for some z, implies an ability to find d is not known to the authors. In fact, conceivably, the ECDLP could be hard, even though certain information about the discrete logarithm, such as whether it is congruent to an x-coordinate, is easily discernible.

Certain bits in the binary representation of the d have been shown by Kaliski [2] to be as hard to find as the whole of d. A bit of information with such a property is known as a *hardcore predicate* for the ECDLP. Kaliski's proof that certain bits of the binary representation of the discrete logarithm are hardcore predicate works with a reduction, as follows. Given $Q = dP$, determine from Q a bit of information about d. Then transform Q to some $Q' = d'P$ in such a way that there is an known relation between d and d', and d' has one less bit of freedom than d. Next determine a bit of information about d', then d'' and so on. The transformation is such that all bits of information learnt are independent and can be easily be reconstituted to learn d in its entirety.

What would be ideal to make Conjecture 2 into a theorem, would be another transformation with comparable properties to Kaliski's for determining the discrete logarithm d using an oracle for solving XLP. Instead we have the following result, Theorem 1, which is not ideal in that it

1. is not a tight reduction,
2. concerns a harder variant of the XLP.

The *Arbitrary-base x-Logarithm Problem (AXLP)* for elliptic curve $E(\mathbb{F}_q)$ is to distinguish between (P, dP) and (P, xP) where: d is an integer chosen uniformly at random in $[0, n-1]$; and $x = x(Z)$ for points P and Z chosen uniformly at random in $E(\mathbb{F}_q)$. The distinction between AXLP and XLP is that in XLP, the adversary needs only to succeed for fixed base P, whereas in AXLP, the adversary must succeed for any base P.

Note that for r chosen uniformly at random from $[0, n-1]$, we have that $(P, dP) \cong (rP, rdP)$ and $(P, xP) \cong (rP, rxP)$. This means that any adversary A can be replaced by an equally effective adversary A' that first randomizes its input by multiplying both points by a random integer r. Let

$$f_i = \Pr[A(P, iP) = 1] \text{ and } f = \Pr[A(P, Q) = 1] = \frac{1}{n}\sum_{i=0}^{n-1} f_i \tag{6}$$

where the probabilities are taken over random P and Q.

Lemma 4. *For any adversary A against AXLP, there exists an adversary B against DDH with advantage*

$$\epsilon = \frac{2}{n}\sum_{i=0}^{n-1}(f_i - f)^2. \tag{7}$$

Proof. The idea is that for a DDH triple (Q, R, S), $\log_P Q = \log_R S$. If we run A with the pairs (P, Q) and (R, S) as input, the output should be correlated. But if (Q, R, S) is a random triple, the output should be uncorrelated.

Let B be the algorithm that on input of (Q, R, S) samples u and v uniformly at random from $0, 1, ..., n-1$, then runs A on (uP, uR) and (vQ, vS). If the outputs are equal, B outputs 1, otherwise 0.

We compute the advantage of B in distinguishing DDH tuples from random tuples as $\epsilon = |\delta_0 - \delta_1|$, where δ_0 is the probability that B outputs 1 on input of a DDH triple, and δ_1 is the probability that B outputs 1 on input of a random triple.

For a DDH triple $(Q, R, S) = (qP, rP, qrP)$, the two runs of A will have identical the same input distributions. That is, we can deal with each possible logarithm $r = i$ separately and get

$$\begin{aligned}
\delta_0 &= \Pr[B(Q, R, S) = 1] \\
&= \Pr[A(uP, ruP) = A(vQ, rvQ) = 1] + \Pr[A(uP, ruP) = A(vQ, rvQ) = 0] \\
&= \sum_i (\Pr[A(P, iP) = 1]^2 + \Pr[A(P, iP) = 0]^2)/n \\
&= \sum_i (f_i^2 + (1 - f_i)^2)/n \\
&= \sum_i (1 + 2f_i^2 - 2f_i)/n.
\end{aligned} \tag{8}$$

For a random triple, the two runs of A will be independent, and we get

$$\begin{aligned}
\delta_1 &= \Pr[A(P, jP) = 1 \mid j \text{ random}]^2 + \Pr[A(P, jP) = 0 \mid j \text{ random}]^2 \\
&= f^2 + (1 - f)^2 = 1 + 2f^2 - 2f \\
&= \sum_i (1 + 2f^2 - 2f_i)/n.
\end{aligned} \tag{9}$$

Summing up, we get

$$\begin{aligned}\epsilon &= \left|\sum_i (1+2f_i^2-2f_i-(1+2f^2-2f_i))/n\right| \\ &= (2/n)\left|\sum_i (f_i^2-f^2)\right| = (2/n)\sum_i (f_i-f)^2\end{aligned} \tag{10}$$

which completes the proof. □

We now need to show that this adversary against DDH has a significant advantage if the AXLP adversary has a significant advantage.

Let a_i be the probability that the x-coordinate of a random point Z of order n is i modulo n, that is

$$a_i = \Pr[x(Z) \equiv i \pmod{n} \mid Z \text{ random of order } n]. \tag{11}$$

The signed advantage of an adversary A against AXLP is

$$\begin{aligned}\epsilon' = \Pr[&A(P, iP) = 1 \mid i = x(Z), Z \text{ random of order } n] - \\ &\Pr[A(P, iP) = 1 \mid i \text{ random}] \\ = &\sum_i \Pr[A(P, iP) = 1](a_i - 1/n) \\ = &\sum_i f_i(a_i - 1/n).\end{aligned} \tag{12}$$

Since $\sum_i (a_i - 1/n) = 0$, we can write

$$\epsilon' = \sum_i (f_i - f)(a_i - 1/n). \tag{13}$$

Next, let t be the maximal number of points on a curve with the same x-coordinate modulo n. (That is, t is at most twice the cofactor.) Then $|a_i - 1/n| \leq (t-1)/n$, or alternatively

$$|a_i - 1/n|\frac{n}{t-1} \leq 1. \tag{14}$$

Lemma 5. *For any adversary A against AXLP, we have*

$$\frac{2}{n}\sum_i (f_i - f)^2 \geq \frac{2}{t-1}(\epsilon')^2, \tag{15}$$

where f_i, f, t and ϵ' are as defined above.

Proof. We multiply each term in the sum with $((a_i - 1/n)\, n/(t-1))^2 \leq 1$, getting

$$\frac{2}{n}\sum_i (f_i - f)^2 \geq \frac{2}{(t-1)^2}\,\frac{1}{n}\sum_i (f_i - f)^2 (a_i - 1/n)^2 n^2 = \mu. \tag{16}$$

Using the fact that the average sum of squares is greater than the square of the average (Jensen's inequality applied to $z \mapsto z^2$), we get that

$$\mu \geq \frac{2}{(t-1)^2} \left(\sum_i \frac{1}{n}(f_i - f)(a_i - 1/n)n \right)^2 = \frac{2}{(t-1)^2}(\epsilon')^2, \tag{17}$$

which concludes the proof. □

Theorem 1. *If the DDH problem is hard and the cofactor is small, then AXLP is hard.*

Proof. If there exists an adversary A against AXLP with advantage $|\epsilon'|$, then by Lemma 4 and 5 there exists an adversary against DDH with advantage

$$\epsilon \geq \frac{2}{(t-1)^2}(\epsilon')^2. \tag{18}$$

If $|\epsilon'|$ is non-negligible and the cofactor (and hence t) is small, then ϵ is non-negligible. □

This reduction is not tight. If $|\epsilon'| \approx \frac{1}{2}$ and the cofactor is 1, then $\epsilon \approx \frac{1}{2}$. On the the other hand, if $|\epsilon'| \approx 2^{-40}$, then $\epsilon \approx 2^{-80}$. It may be the case that a feasible distinguisher for DDH exists with this advantage, in which case an efficient distinguisher for AXLP with advantage 2^{-40} could not be ruled out.

In practice, the bound given by t is not sharp. For the cofactor two NIST curves, one can prove that for almost all $d \in [0, n-1]$, at most one of d and $d+n$ can be the x-coordinate of a valid point. This is because a valid x-coordinate has a certain trace, and in trinomial or pentanomial basis representation, the trace depends on a few bits of the x-coordinate including the least significant bit. From this we see that $t = 2$ would work. For cofactor four curves, we only have a heuristic estimate for t, which presumably could be confirmed (or denied) by further analysis.

The advantage of B in the proof above can possibly be increased by also comparing A's run on (T, V) and (U, W) and other such pairings, although one does not expect a significant increase compared to increased runtime.

Because AXLP appears to be a hard problem, it seems quite reasonable to conjecture that XLP is also a hard problem. It should be noted that the hardness of XLP suggests that the output of the ECRNG, before truncation, is suitable to use for the generation of an ECC private key. One would conjecture that it would therefore be just as sensible to concatenate enough output of the properly truncated ECRNG, and use these as a ECC private key, since presumably truncation and concatenation should not decrease the security.

6 Security of the Raw ECRNG Outputs Points

The traditional notion of security for an RNG is that its output is indistinguishable from random. We prove below in Theorem 2 that the raw output points of the ECRNG are indistinguishable from random points, that they are pseudorandom as points. Furthermore, we prove that the points are forward secure, as defined below.

The following proof is not substantially different than the proof for the Blum-Micali generator [1] or the Kaliski generator [2]. However, unlike these generators, which used a hardcore bit of the discrete logarithm, the ECRNG uses a hardcore function — as suggested, for example, in [14] — which yields greater efficiency provide that one accepts hardness of the corresponding problem, DDH, to ensure the function is hardcore. A second reason for providing the proof anew here is that the state update transition function is not a permutation. This issue is addressed via recourse to the hardness of the x-logarithm problem. Roughly speaking, hardness of XLP ensures that the state transition function is indistinguishable from a permutation.

In cryptology, *forward secrecy* refers to the following property: present secrets remain secret into the future, even from an adversary who acquires all future secrets. So, in forward secrecy, the secrecy of present secrets extends forward into the future indefinitely and without depending on protection of some future secrets. Many key agreement schemes, and even some digital signature schemes, claim forward secrecy. When implementing these schemes, one likely needs to ensure that any RNGs used have forward secrecy too. In [5, 3], *forward secrecy* has been renamed[4] *backtracking resistance* to convey the notion that an adversary cannot use future secret to backtrack to present secrets.

To model forward secrecy, we let adversary see the latest prestate, but still it cannot distinguish previous output points from random points.

Theorem 2. *If the DDH and XLP problems are hard, and $Q, Z_0, \ldots, Z_m, Z_{m+1}$ are independent and uniformly distributed random points, and s_0 is a random integer uniformly distributed in $[0, n-1]$, and $g_m(Q, s_0) = (R_0, \ldots, R_m)$, with the next prestate of the ECRNG being S_{m+1}, then*

$$(Q, R_0, \ldots, R_m, S_{m+1}) \sim (Q, Z_0, \ldots, Z_m, Z_{m+1}). \tag{19}$$

Proof. The case of $m = 0$ is to show $(Q, R_0, S_1) \sim (Q, Z_0, Z_1)$. This follows directly from hardness of the DDH problem. Assume by induction that

$$(Q, R_0, \ldots, R_{m-1}, S_m) \sim (Q, Z_0, \ldots, Z_{m-1}, Z_m). \tag{20}$$

The current outputs and prestate are given by

$$(Q, R_0, \ldots, R_{m-1}, R_m, S_{m+1}) = (Q, R_0, \ldots, R_{m-1}, x(S_m)Q, x(S_m)P) \tag{21}$$

Combining (20) and (21), and applying Lemma 2, we get

$$(Q, R_0, \ldots, R_{m-1}, R_m, S_{m+1}) \sim (Q, Z_0, \ldots, Z_{m-1}, x(Z_m)Q, x(Z_m)P). \tag{22}$$

[4] Breaking precedent not only with wider usage in cryptology but also with other ANSI standards such as X9.42 and X9.62, which use forward secrecy.

Hardness of XLP gives $x(Z_m)P \sim Z_{m+1}$. Writing $Q = qP$, Lemma 2 gives

$$(Q, Z_0, \ldots, Z_{m-1}, x(Z_m)Q, x(Z_m)P) \sim (qP, Z_0, \ldots, Z_{m-1}, qZ_{m+1}, Z_{m+1}). \tag{23}$$

Hardness of DDH gives $(qP, qZ_{m+1}, Z_{m+1}) \sim (Q, Z_m, Z_{m+1})$ where $Q = qP$, so Lemma 3 gives

$$(qP, Z_0, \ldots, Z_{m-1}, qZ_{m+1}, Z_{m+1}) \sim (Q, Z_0, \ldots, Z_{m-1}, Z_m, Z_{m+1}). \tag{24}$$

Lemma 1 on transitivity connects (22) to (23) to (24) to complete the inductive step, getting us our desired result. □

This proof makes essential use of Q being random. The reason for this is more than just to make the proof work. If Q is not random, then it may be the case the adversary knows a d such that $dQ = P$. Then $dR_i = dS_{i+1}$, so that such a distinguisher could immediately recover the secret prestates from the output. Once the distinguisher gets the prestates, it can easily distinguish the output from random. Therefore, it is generally preferable for Q to be chosen randomly, relative to P.

Although Theorem 2 says that hardness of the DDH problem is one of the *sufficient* conditions for indistinguishability of the ECRNG output points, it is not at all clear whether or not hardness of the DDH problem is a *necessary* condition. It is clear that hardness of the *computational* Diffie-Hellman problem (CDH) is a necessary condition in that S_{i+1} is the Diffie-Hellman product of P and R_i to the base Q.

Hardness of XLP, however, is necessary for indistinguishability of the raw output points. Output $R_1 = s_1 Q = x(S_1)Q$, so distinguishing it from random Z_1 is essentially XLP. Distinguishing output $R_j = x(S_{j-1})Q$ from random is almost XLP except that S_{j-1} is not necessarily a random point. However, if distinguishing algorithm A is an XLP solver, then one heuristically expects that A could distinguish R_j from a random point. Algorithm A would only fail if the S_{j-1} were distributed with a bias such that A reports that $x(S_{j-1})Q$ was not of the form $x(Z)P$ for some valid point Z. Therefore one cannot hope to strengthen Theorem 2 by replacing the hardness of XLP with a weaker yet still natural assumption. One could improve the result, however, by proving that XLP is as hard as some other problems, such as DDH or ECDLP.

7 Truncated Point Problem and Security of the Full ECRNG

For appropriate choice of truncation function $t(\cdot)$, we conjecture the following.

Conjecture 3. Let R be a random point and b a random bit string of length matching the output length of $t(\cdot)$. Then $t(x(R)) \sim b$.

We call the problem of distinguishing between $t(x(R))$ and b, the *Truncated Point Problem* (TPP). This paper does not substantially address this conjecture, but rather uses it to prove something about the final output of the ECRNG, rather than just its raw output points.

Theorem 3. *If the DDH, XLP and TPP problems are hard, then the ECRNG has forward secrecy.*

Proof. Apply Theorem 2 to get that the raw outputs are indistinguishable. By the assumed hardness of the TPP problem, each truncated point is indistinguishable from random bit strings. Apply the lemmas as necessary and get that the ECRNG output bit strings are indistinguishable from random bit strings, even from an adversary that gets to see the latest prestate. □

Although hardness of XLP is necessary for the raw output points to be pseudorandom, it does not seem necessary for the full ECRNG output bit strings to be pseudorandom. Likewise, hardness of CDH may not be necessary for security of the full ECRNG, even if it is necessary for the indistinguishability of the raw output points. Truncation of the raw output points may yield bit strings are that unusable even by an XLP distinguisher or a CDH solver to distinguish the ECRNG outputs.

We note that it is straightforward to generalize the construction to any group G with suitable maps $x : G \to \mathbb{Z}_n$ and $t : G \to \{0,1\}^l$. If the corresponding DDH, XLP and TPP problems are hard, the generator will have forward secrecy. If the map x is a permutation, the corresponding XLP will be trivially hard.

The proposed truncation function drops some number of the leftmost bits of the bit representation of the x-coordinate. The number of bits dropped is at least $13+\log_2(h)$, where h is the cofactor. The number of bits dropped must also be such that resulting length is a multiple of eight. Current (draft) standards allow any number of bits to be dropped that meets these conditions.

We consider if $B \sim t(x(R))$ where R is a random point and B is a random bit string whose output length l matches that of the truncation function. Let k be the number of bits truncated from $x(R)$, which has length $m = k + l$.

It is well-known that the advantage of any distinguisher is bounded above by the statistical distance between the distributions B and $t(x(R))$, and that the optimal distinguisher has advantage equal to the statistical distance. The statistical distance Δ between B and $t(x(R))$ is by definition

$$\Delta = \Delta(B, t(x(R))) = \sum_b \left|\Pr[t(x(R)) = b] - \Pr[B = b]\right|. \tag{25}$$

The easier probability to compute is $\Pr[B = b] = 2^{-l}$ because all 2^l bit strings b are equally likely. The other probability has theoretical formula given by $\Pr[t(x(R))] = \frac{n(b)}{n}$, where $n(b)$ is the number of valid points R such that $t(x(R)) = b$. Note the $n(b)$ is always even, if we ignore the negligibly frequent case $R = 0$, because $x(R) = x(-R)$. Also, as k bits of the x-coordinate are truncated, we have $0 \leq n(b)/2 \leq 2^k$. Let B_i be the number of b such that $n(b) = 2i$. Then

$$\Delta = \sum_{i=0}^{2^k} B_i \left|\frac{2i}{n} - 2^{-l}\right|. \tag{26}$$

Now we make some heuristic assumptions. Assume that the set X of valid x-coordinates is a random subset of bit strings of length $k + l$, such that each bit string belongs to X with probability $1/(2h)$, where h is the cofactor. Consider cofactor $h = 1$. Our first heuristic assumption implies B_i has a binomial distribution, so its approximate expectation is:

$$E(B_i) \approx 2^{l-2^k} \binom{2^k}{i}, \tag{27}$$

where E is not to be confused with the elliptic curve equation. This distribution is because there are 2^k bit strings of length $k + l$ that truncate to a given bit string b of length l, and each of these completions of b has probability $\frac{1}{2}$ of belonging to X. Typically a few B_i may veer off considerably from the expected value. Nevertheless, by linearity of expectation, we can substitute these expected values into (25), getting expected statistical distance:

$$E(\Delta) \approx \sum_{i=0}^{2^k} 2^{l-2^k} \binom{2^k}{i} \left| \frac{2i}{n} - 2^{-l} \right|. \tag{28}$$

Take the approximation $n \approx 2^{l+k}$, to get a second heuristic assumption that $\frac{2i}{n} \approx \frac{2i}{2^{k+l}}$. Pulling a common factor through the sum gives

$$E(\Delta) \approx 2^{-2^k-k+1} \sum_{i=0}^{2^k} \binom{2^k}{i} \left| i - 2^{k-1} \right|. \tag{29}$$

The terms with $0 \leq i \leq 2^{k-1}$ are identical to those with $2^k \geq i \geq 2^{k-1}$, and the term with $i = 2^{k-1}$ is zero, so we can eliminate the absolute value signs, getting

$$E(\Delta) \approx 2^{-2^k-k+2} \sum_{i=0}^{2^{k-1}} \binom{2^k}{i} \left(2^{k-1} - i\right). \tag{30}$$

Using the general identity $i\binom{j}{i} = j\binom{j-1}{i-1}$, with a convention that $\binom{j-1}{-1} = 0$, gives

$$E(\Delta) \approx 2^{-2^k-k+2} \sum_{i=0}^{2^{k-1}} \left(2^{k-1}\binom{2^k}{i} - 2^k \binom{2^k-1}{i-1} \right). \tag{31}$$

Pulling out common factor 2^{k-1} from the sum and the general identity $\binom{j}{i} = \binom{j-1}{i-1} + \binom{j-1}{i}$ gives

$$E(\Delta) \approx 2^{-2^k+1} \sum_{i=0}^{2^{k-1}} \left(\binom{2^k-1}{i} - \binom{2^k-1}{i-1} \right). \tag{32}$$

This summation telescopes, giving

$$E(\Delta) \approx 2^{-2^k+1} \binom{2^k-1}{2^{k-1}} \tag{33}$$

For large even J, Stirling's approximation gives a third heuristic assumption that $\binom{J}{J/2} \approx \frac{2^J}{\sqrt{2\pi J}}$, and clearly $\binom{J-1}{J/2} = \frac{1}{2}\binom{J}{J/2}$. Applying this to (33) with $J = 2^k$ gives

$$E(\Delta) \approx \frac{1}{\sqrt{2\pi 2^k}} \tag{34}$$

as the heuristic value for the statistical distance. For $k = 16$, the heuristic for the expected statistical distance is about $\frac{1}{641}$. Although this is just a heuristic, it may be prudent to pay the price of using a larger k when a high degree of indistinguishability is desired. If only unpredictability is desired, say if the ECRNG is used for nonces or keys, but not as one-time pads, then this may not be so critical.

An optimal distinguisher can be constructed by computing $n(b)$, which has a computational cost of about 2^k times the cost of validating a potential x-coordinate. If $\frac{n(b)}{n} > 2^{-l}$ report $t(x(R))$, otherwise report B. Provided that k is small enough, then one heuristically expects that an efficient distinguisher exists with advantage of about that given by (34).

Rather than making all these highly heuristic assumptions, one can also gather some empirical data by sampling random B to infer an approximate distribution for $n(B)$. The inferred distribution can be used as estimates for the quantities B_i and thus the statistical distance Δ. An accurate inference requires a large random sampling. We have carried out quite extensive experiments, and they confirm the heuristic estimates.

One possibility for repairing the ECRNG is to use standard techniques for entropy extraction to process the raw outputs. In order to increase the asymptotic output rate, one may want to extract output from tuples of points instead of single points. This introduces a higher startup cost and requires a buffer to hold the points, but this can be tolerated in many practical applications.

Acknowledgments

The first author thanks the ANSI X9F1 working group for introducing him to the ECRNG, and Certicom colleagues for valuable discussions, especially Matt Campagna for a careful reading. The second author thanks John Kelsey for introducing him to the ECRNG. We would also like to thank the anonymous referees for helpful comments.

References

1. Blum, M., Micali, S.: How to generate cryptographically strong sequences of pseudo-random bits. SIAM Journal on Computing **13** (1984) 850–864
2. Kaliski, B.S.: A pseudo-random bit generator based on elliptic logarithms. In Odlyzko, A.M., ed.: Advances in Cryptology — CRYPTO '86. Volume 263 of LNCS., Springer (1986) 84–103

3. Barker, E., Kelsey, J.: Recommendation for Random Number Generation Using Deterministic Random Bit Generators. National Institute of Standards and Technology. (2006) http://csrc.nist.gov/CryptoToolkit/RNG/SP800-90_June2006.pdf.
4. Johnson, D.B.: X9.82 part 3 number theoretic DRBGs. Presentation at NIST RNG Workshop (2004) http://csrc.nist.gov/CryptoToolkit/RNG/Workshop/NumberTheoreticDRBG.pdf.
5. Barker, E.: ANSI X9.82: Part 3 — 2006, Random Number Generation, Part 3: Deterministic Random Bit Generators. American National Standards Institute. (2006) Draft. http://www.x9.org/.
6. Standards for Efficient Cryptography Group: SEC 1: Elliptic Curve for Cryptography. Draft 1.7 edn. (2006) http://www.secg.org/.
7. Boneh, D.: The decision Diffie-Hellman problem. In Buhler, J.P., ed.: Algorithmic Number Theory, ANTS-III. Volume 1423 of LNCS., Springer (1998) 48–63 http://crypto.stanford.edu/~dabo/abstracts/DDH.html.
8. Mahassni, E.E., Shparlinksi, I.: On the uniformity of distribution of congruential generators over elliptic curves. In: International Conference on Sequences and Their Applications, SETA '01, Springer (2002) 257–264
9. Gürel, N.: Extracting bits from coordinates of a point of an elliptic curve. ePrint 2005/324, IACR (2005) http://eprint.iacr.org/.
10. Schoenmakers, B., Sidorenko, A.: Cryptanalysis of the dual elliptic curve pseudorandom generator. ePrint 2006/190, IACR (2006) http://eprint.iacr.org/.
11. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In IEEE, ed.: Foundations of Computer Science, FOCS '97, IEEE Computer Society Press (1997) 458–467 http://www.wisdom.weizmann.ac.il/~reingold/publications/GDH.PS.
12. Farashahi, R.R., Schoenmakers, B., Sidorenko, A.: Efficient pseudorandom generators based on the DDH assumption. ePrint 2006/321, IACR (2006) http://eprint.iacr.org/.
13. Luby, M.: Pseudorandomness and Cryptographic Applications. Princeton University Press (1996)
14. Goldreich, O.: Foundations of Cryptography. Cambridge University Press (2001)
15. Smart, N.P.: A note on the x-coordinate of points on an elliptic curve in characteristic two. Information Processing Letters **80**(5) (2001) 261–263

A Unpredictability of the Next State from the Current Output

Unpredictability is a much weaker property than indistinguishability, but is also much more important. If the ECRNG outputs are used as cryptographic keys, very little harm may come from them being distinguishable. If they are predictable, however, then all may be lost. Indistinguishability implies unpredictably, so in fact, we have already proven unpredictability.

The theorem below, however, proves a little bit of unpredictability under weaker, arguably more accepted, conjectures, such as hardness of CDH instead of the hardness of the DDH problem.

Theorem 4. *If CDH and XLP are hard, and q and s_0 are independent random integers uniformly distributed in $[0, n-1]$, and $g_m(qP, s_0) = (R_0, \ldots, R_m)$ and*

$Q = qP$, then an adversary who gets to see only Q and R_m cannot compute the next prestate S_{m+1}.

Proof. Clearly $S_1 \sim Z$ where $Z = zP$ and z is a random integer uniformly distributed in $[0, n-1]$. Indeed, $s_0 \cong z$, so $S_1 = s_0 P \cong zP = Z$. Assume by induction that $S_{j-1} \sim Z$. Now $S_j = x(S_{j-1})P \sim x(Z)P \sim Z$, with the second indistinguishability flowing from the hardness of XLP. Therefore $S_{m+1} \sim Z$. Since q is independent of z, we have $(Q, S_{m+1}) \sim (Q, Z)$. Now $(Q, R_m) = (Q, qS_{m+1}) \sim (Q, qZ) \sim (Q, Z)$, with the second indistinguishability flowing from Z being able to absorb q by independence.

Suppose adversary A takes (Q, R_m) and outputs $S_{m+1} = q^{-1}R_m$. Then adversary can also take (Q, Z) and output $U = q^{-1}Z$, because otherwise A could distinguish (Q, R_m) from (Q, Z). Let $(X, Y) = (xP, yP)$ with x, y independent random integers uniformly distributed in $[0, n-1]$. We will use A to compute xyP. Pick a random integer u with the same distribution. Let $U = uP$. Apply A to (X, U) to get $V = x^{-1}U = x^{-1}uP$. Let $W = u^{-1}V = x^{-1}P = wP$. Apply A to (W, Y) to get $w^{-1}Y = (x^{-1})^{-1}Y = xY = xyP$, as desired. Because we assumed that CDH is hard, adversary A cannot find xyP, so we get a contradiction. □

The simple proof techniques above do not seem to rule out an adversary who could use two output points to find the next state, or one output point to find the next output point. The obstacle in the first case seems to be that output points obey a relationship that needs to be simulated if we wish to solve CDH. The obstacle in the second case is that the next output can be thought of as a one-way function of the Diffie-Hellman product of public values, and we seem to need to invert it to solve CDH.

B A Caution About the Truncated Point Problem for Binary Curves

It should be noted that for the NIST recommended curves defined over the binary field $\mathbb{F}_{2^{409}}$, valid elliptic curve points have a fixed rightmost bit in their canonical representation. Therefore, for the curves B-409 and K-409, the truncation function should also drop the rightmost bit. The explanation for this phenomenon (see also [15]) is that one of the conditions for a point to have the correct order can be characterized by the trace of the x-coordinate have a fixed value. The trace depends on the field representation. For trinomial and pentanomial field representations, the trace simplifies to a sum of just a few of the bits, the *trace bits*, in the representation. In all fields, the rightmost bit is a trace bit. For the 409-bit field, the rightmost bit is the only trace bit. For the other four NIST recommended binary fields, there is at least one trace bit among the leftmost truncated bits. Consequently, the constant trace condition does not leak any information after truncation in these cases.