The Twisted Dual Elliptic Curve Deterministic Random Bit Generator

Holden Mui, Linus Tang, Noah Walsh

May 2025

Abstract

For our 6.5610 (Applied Cryptography) final project, we present an backdoorable DRBG based on Dual EC DRBG that, unlike Dual EC DRBG, is a true DRBG under suitable assumptions. Our algorithm, which we call the *Twisted Dual EC DRBG*, involves iteratively multiplying points on an elliptic curve or its quadratic twist.

1 Introduction

The Dual EC DRBG, short for Dual Elliptic Curve Deterministic Random Bit Generator, is an algorithm for random bit generation developed by the National Security Agency (NSA) and standardized by the National Institute of Standards and Technology (NIST) in 2006. Dual EC DRBG is famous for being potentially backdoored by the NSA, and it was formally withdrawn by NIST in 2014.

1.1 Importance

Random numbers are critical for the secure operation of many cryptographic protocols, with applications from key generation to cryptographic salt or padding. It is important that these random numbers are unguessable, or else the security of any scheme using them would be compromised.

To generate random numbers, software developers rely on algorithms known as deterministic random bit generators (DRBGs). DRBGs take some small amount of "seed" randomness as input and amplify it into a long string of random bits; in typical applications, the random seed is derived from some physical process capable of producing entropy.

Sometimes software developers design their own DRBGs, but more often than not, they rely on existing standards for pseudorandom bit generation. This is because it is difficult to design a DRBG without introducing bias or patterns in the output bits, and existing standards have already been subject to years of public scrutiny. Existing standards are usually issued by standards organizations such as the National Institute of Standards and Technology (NIST) and the International Organization for Standardization (ISO).

1.2 History

Dual EC DRBG was developed by the NSA in the early 2000s and first publicly presented in 2004 at a NIST workshop on random number generation [6]. It was noted that Dual EC DRBG was a slow DRBG, but the presentation claimed "increased assurance". In 2005, NIST published a draft of the complete Dual EC DRBG algorithm, asking for public comments.

Three months later, Kristin Gjøsteen pointed out that Dual EC DRBG had biases in the output bits [5]. Berry Schoenmakers and Andrey Sidorenko improved the distinguisher a few months later, showing that Dual EC DRBG bits could be distinguished from truly random bits with advantage 4.9%, using 2.5 hours of computation on a personal computer [3]. They state that the attack could be patched by masking more output bits, but NIST never adopted this recommendation. In hindsight, this decision was understandable because this modification would have closed the back door. In June 2006, NIST formally adopted Dual EC DRBG despite biases in its output.

The Dual EC DRBG was also adopted as a standard by the ISO [1]. The story behind the inclusion of Dual EC DRBG in this multi-year process of standardization is also interesting. A 2003 draft of the standards, which did not include Dual EC DRBG, was approved by 19 out of 24 voting countries. However, the United States did not approve the standard; instead, it sent an objection stating that the standards "lacked sufficient depth". Instead, the United States proposed its own set of standards that included Dual EC DRBG, stating that their standards were "much further developed" [7]. The ISO ultimately ended up adopting the standards in the United States' proposal in late 2005.

In August of 2007, Dan Shumow and Niels Ferguson announced the possibility of a back door in Dual EC DRBG at an annual cryptography conference [13]. They showed that the elliptic curve points P and Q used in the Dual EC DRBG algorithm could have been generated in a way that enables the generating party to predict all future bits given a single 240-bit block of the DRBG. Shortly after, Wired published an article [11] calling Dual EC DRBG "an algorithm that is slow, badly designed and just might contain a backdoor for the National Security Agency". In response, NIST sent Wired a letter stating that there is "no evidence that anyone has... the 'secret numbers' for the back door" [2].

That was the end of the discussion until 2013, when Edward Snowden leaked sensitive government documents to journalists. According to the New York Times, "Classified N.S.A. memos appear to confirm that the fatal weakness, discovered by two Microsoft cryptographers in 2007, was engineered by the agency," referring to the backdoor discovered by Shumow and Ferguson [10]. Furthermore, it was revealed that the NSA secretly paid $10 million to RSA Security to use Dual EC DRBG as its default random number generator [8].

In 2014, NIST finally withdrew Dual EC DRBG from its list of standard random number generators, recommending that users switch to one of the three other standards [9]. At this point, the damage had been done; the inclusion of Dual EC DRBG in NIST's standards undermined trust in the government among the cryptographic community.

The story behind Dual EC DRBG is truly fascinating. For more details, there is a comprehensive timeline given in [4].

1.3 Our Work

Recall that Gjøssten, Schoenmakers, and Sidorenko showed that Dual EC DRBG was not a true DRBG because it was possible to distinguish Dual EC DRBG bits from truly random bits with high probability. Our 6.5610 project is to modify the Dual EC DRBG algorithm to remove the distinguisher while retaining the backdoor.

There are several naive modifications that do not work.

- Following the suggestion of [12], one could modify Dual EC DRBG to output fewer bits. However, this closes the back door.
- One could modify Dual EC DRBG to use a different elliptic curve and/or different points P and Q. However, the distinguishing attack still exists because it does not depend on the curve used, nor the points P or Q.

Instead, we propose an algorithm based on Dual EC DRBG that additionally involves a pair of elliptic curves related by a quadratic twist. We call our algorithm the *Twisted Dual EC DRBG*. The Twisted Dual EC DRBG has the property that the points in the algorithm could be chosen so that the generating party can predict all future bits given a single output block of the DRBG. However, the distinguishing attack of Gjøssten, Schoenmakers, and Sidorenko no longer applies because the nonuniformity of output bits is smoothed out by using both the elliptic curve and its twist.

1.4 Outline

Useful definitions are given in Section 2.1. Section 2.2 contains the complete description of the Dual EC DRBG algorithm, as described in [3]. Section 2.3 contains a description of the possible backdoor, as described in [13]. Section 2.4 contains the algorithm for the distinguisher, as described in [12]. Additionally, we prove that the distinguisher works under suitable assumptions, which Schoenmakers and Sidorenko do not provide, since their work only verifies the distinguisher empirically.

Our Twisted Dual EC DRBG algorithm is given in Section 3.1. As promised, the DRBG is still backdoorable; this process is described in Section 3.2. However, unlike the Dual EC DRBG, the Twisted Dual EC DRBG is secure in the sense that no probabilistic polynomial-time algorithm can distinguish the Twisted Dual EC DRBG output from truly random output. A proof of this fact is given in Section 3.3 with suitable assumptions. A runtime analysis of the Twisted Dual EC DRBG is given in Section 3.4. Finally, an implementation of the Twisted Dual EC DRBG is given in Section 3.5.

Finally, possible future directions are discussed in Section 4, and the work distribution is given in 5 as required by the final project instructions.

2 Background

2.1 Definitions

A *deterministic random number generator*, or DRBG, is an algorithm that outputs a bit sequence given an input seed. It must satisfy two properties:

- the DRBG must be deterministic; that is, it produces the same output given the same seed.
- the DRBG must be secure; that is, indistinguishable from true randomness without knowledge of the seed.

Formally, a DRBG is parameterized by a security parameter $\lambda \in \mathbb{N}$. It consists of the following algorithms:

- A probabilistic polynomial-time algorithm $\text{Gen}(1^\lambda) \to \text{param}$.
- A polynomial-time algorithm $\text{DRBG}(\text{param}, \text{seed}) \to \{0,1\}^{\text{poly}(\lambda)}$ which takes as input a λ-bit seed and outputs a $\text{poly}(\lambda)$-bit string.

For security, it must have the property that for every probabilistic polynomial-time distinguisher $\mathcal{D}(\text{param}, \cdot)$ taking values in $\{0,1\}$,

$$|\Pr[\mathcal{D}(\text{param}, \text{DRBG}(\text{param}, \text{seed})) = 1] - \Pr[\mathcal{D}(\text{param}, R) = 1]| < \text{negl}(\lambda)$$

where R is a random $\text{poly}(\lambda)$-bit string, and the probability is taken over param, R, and $\mathcal{D}$'s internal randomness.

A DRBG is *backdoored* if there exists some probabilistic polynomial-time algorithm $\text{KeyGen}(1^\lambda) \to (\text{param}, \text{key})$ extending $\text{Gen}(1^\lambda)$ and a probabilistic polynomial-time distinguisher $\mathcal{D}(\text{param}, \text{key})$ with output in $[0,1]$ such that

$$\Pr[\mathcal{D}(\text{param}, \text{key})(\text{DRBG}(\text{param}, \text{seed})) = 1] - \Pr[\mathcal{D}(\text{param}, \text{key})(R) = 1] > 1 - \text{negl}(\lambda)$$

where R is a random $\text{poly}(\lambda)$-bit string, and the probability is taken over param, R, and $\mathcal{D}$.

For a field $\mathbb{F}$ with $\text{char}(\mathbb{F}) \notin \{2,3\}$, an *elliptic curve* over $\mathbb{F}$ is the zero set $(x : y : z)$ of the homogeneous polynomial

$$y^2 z = cx^3 + axz^2 + bz^3$$

in the projective plane, where $a, b, c \in \mathbb{F}$ are constants satisfying $-16c^2(4a^3 + 27b^2c) \neq 0$. All points on an elliptic curve can be written in the form $(x : y : 1)$ via division by z, except for the point $(0 : 1 : 0)$, called the *point at infinity*. Thus, elliptic curve points will be denoted by the coordinate pair (x, y) with the assumption that the z-coordinate is 1.

The set of points on an elliptic curve form an abelian group. The group operation is implied by the following two constraints.

- The point at infinity is the identity.

- The three points of intersection of any line with the elliptic curve sum to the identity.

Inverses can be computed using lines passing through the point at infinity, and the sum of any two points A and B is given by the inverse of the third point of intersection of line AB and the curve. (If $A = B = P$, then line AB is the line through P tangent to the elliptic curve. If $\mathbb{F}$ is a finite field, then the tangency condition is given by a constraint of the formal derivative of the elliptic curve and the slope of the line through P.)

The identity axiom, inverse axiom, and commutativity follow directly. Associativity can be proved via a long and involved algebraic manipulation. Points on elliptic curves can be multiplied by integers, where

$$nP = \underbrace{P + \cdots + P}_{n\ P\text{'s}}$$

for positive integers n, $nP = (-n)(-P)$ for negative n, and $0P$ is the point at infinity.

Given an elliptic curve $y^2 = cx^3 + ax + b$ over a field $\mathbb{F}$, its *quadratic twist* by d is the elliptic curve

$$dy^2 = x^3 + ax + b$$

where $d \in \mathbb{F}_p$ is not a square. Mathematically, all quadratic twists are isomorphic to each other, so it does not matter which quadratic nonresidue $d \in \mathbb{F}_p$ is chosen to construct a twist.

2.2 The Dual EC DRBG algorithm

This section will restate the complete Dual EC DRBG algorithm as given in [3] for the security parameter $\lambda = 256$. The *Dual* EC DRBG takes its name from the fact that the algorithm uses *two* elliptic curve points.

Consider the elliptic curve P-256, which is defined by the equation $x^3 - 3x + b$ over $\mathbb{F}_p$ with parameters given in hexadecimal by

$$\begin{aligned} p &= 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1 \\ &= \texttt{0xffffffff00000001000000000000000000000000ffffffffffffffffffffffff} \\ b &= \texttt{0x5ac635d8aa3a93e7b3ebbd55769886bc651d06b0cc53b0f63bce3c3e27d2604b} \end{aligned}$$

Let P and Q be the points on the curve given by coordinates

$$\begin{aligned} P_x &= \texttt{0x6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296} \\ P_y &= \texttt{0x4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5} \\ Q_x &= \texttt{0xc97445f45cdef9f0d3e05e1e585fc297235b82b5be8ff3efca67c59852018192} \\ Q_y &= \texttt{0xb28ef557ba31dfcbdd21ac46e2a91e3c304f44cb87058ada2cb815151e610046}. \end{aligned}$$

One can verify that these points both lie on the elliptic curve, and that the order of the elliptic curve is the prime number

$$\texttt{0xffffffff00000000ffffffffffffffffbce6faada7179e84f3b9cac2fc632551}.$$

Let $x(A)$ denote the x-coordinate of the point A.

The Dual EC DRBG takes a seed $s_0 \in \mathbb{F}_p$ and a positive integer k as input, and it outputs $240k$ pseudorandom bits. It is given by the following algorithm, which keeps track of an internal state $s \in \mathbb{F}_p$. Initially, the internal state s is set to the seed s_0. Then, the following steps are repeated k times:

- Replace s with $x(sP)$.
- Output the last 240 bits of $x(sQ)$.

Note that when computing sP and sQ, s must first be converted from an element of $\mathbb{F}_p$ to an element of $\mathbb{Z}$. This is done via the most natural mapping $\mathbb{F}_p \to \{0, 1, \ldots, p-1\}$.

2.3 The Dual EC DRBG backdoor

This section will restate the Dual EC DRBG backdoor attack as given in [13].

Suppose an adversary knows the discrete logarithm of P with respect to Q; that is, some $k \in \mathbb{Z}$ such that $P = kQ$. Such a number must exist because the order of the elliptic curve group is prime. If the adversary also knew the coordinates of sQ, then they could recover the secret state s because the next secret state is

$$sP = s(kQ) = k(sQ).$$

Now, observe that the Dual EC DRBG reveals the last 240 bits of $x(sQ)$ during each iteration, so an attacker knows all but 16 bits of the x-coordinate of sQ. Thus given a single 240-bit block of Dual EC DRBG output, the adversary can brute force the coordinates of kQ, giving at most $2^{16} = 65536$ possible values for the secret state s. From here, only a few more bits of Dual EC DRBG output are needed to determine which of the 2^{16} possible values of s is the correct secret state. Once s is known, all future output can be predicted with 100% accuracy.

2.4 The Dual EC DRBG distinguisher

This section will restate the Dual EC DRBG distinguisher as given by Schoenmakers and Sidorenko in [12]. This algorithm allows an attacker without knowledge of the backdoor to distinguish Dual EC DRBG bits from random bits. Schoenmakers and Sidorenko were able to empirically distinguish them with 54.9% accuracy using 2.5 hours of computation on a 3GHz Linux machine with one gigabyte of memory. For better accuracy, this process can be repeated multiple times.

To give the distinguishing attack, first define $\phi : \{0,1\}^{240} \to \mathbb{Z}^+$ to be the number of points $\phi(d)$ on the P-256 elliptic curve whose x-coordinate has d as its last 240 binary digits. The attack is to sample the distribution of $\phi(b)$ for many 240-bit blocks b, where $\phi(b)$ is the number of points on the elliptic curve whose x-coordinate has d as its last 240 digits.

The distribution of $\phi(b)$ across truly random 240-bit blocks b has mean $\mu \approx 2^{16}$ and standard deviation $\sigma \approx 256$. However, the distribution of $\phi(b)$ across Dual EC DRBG 240-bit blocks b has mean $\mu \approx 2^{16} + 1$ and standard deviation $\sigma \approx 256$.[1] Thus, if the mean of $\phi(b)$ could be sampled to $\pm\frac{1}{2}$ accuracy, then Dual EC DRBG bits can be distinguished from random bits. For a 95% confidence interval, this would require n samples, where

$$\frac{1.96\sigma}{\sqrt{n}} = \frac{1}{2} \implies n \approx 10^6.$$

Schoenmakers and Sidorenko only empirically verified the discrepancy in the means of the distributions of $\phi(b)$. In what follows, we provide a proof of their claim assuming mild assumptions on the map $\mathcal{E} \to \{0,1\}^{240}$ given by taking the last 240 bits of the x-coordinate. Heuristically, one should expect their means to differ by approximately 1. This is because in the Dual EC DRBG algorithm, the point sQ is already given as a point whose last 240 bits is the output block, increasing the expected number of points having those bits as their last 240 bits of their x-coordinate by 1. This heuristic is proved rigorously via the claim below.

Claim 2.1. *Suppose that $\phi : \mathcal{E} \to \{0,1\}^{240}$ is modeled as a random map $\mathcal{E} \to \{0,1\}^{240}$. Then with high probability,*[2]

(a) The distribution of $\phi(b)$ across truly random 240-bit blocks b has mean $2^{16} + O(2^{-16})$.

(b) The distribution of $\phi(b)$ across Dual EC DRBG 240-bit blocks b has mean $2^{16} + 1 + O(2^{-16})$.

Proof. For each nonnegative integer n, let $r(n)$ denote the probability that exactly n elliptic curve points have an x-coordinate ending in a randomly chosen 240-bit string. Then r gives the distribution of ϕ over random 240-bit blocks, and

$$s(n) = \frac{nr(n)}{\sum_{i\in\mathbb{N}} ir(i)}$$

gives the distribution of ϕ over 240-bit output blocks. The mean of r is

$$\mu = \sum_{i\in\mathbb{N}} ir(i) = \frac{1}{2^{240}} \sum_{i\in\mathbb{N}} i \cdot 2^{240} r(i) = \frac{\#\mathcal{E}}{2^{240}} = \frac{2^{256} - O(2^{224})}{2^{240}} = 2^{16} + O(2^{-16}).$$

The mean of s minus the mean of r is

$$\sum_{i\in\mathbb{N}} is(i) - \sum_{i\in\mathbb{N}} ir(i) = \frac{\sum_{i\in\mathbb{N}} i^2 r(i) - (\sum_{i\in\mathbb{N}} ir(i))^2}{\sum_{i\in\mathbb{N}} ir(i)} = \frac{\sigma^2}{\mu}$$

[1] Schoenmakers and Sidorenko claim that the standard deviation is ≈ 255.6. However, we believe they did not take enough samples to determine the standard deviation to within four significant figures. We will value of 256 later under mild assumptions.

[2] The probability here is probably something like $1 - 2^{O(-240)}$, but the precise details are tedious to work out and will be omitted because they are unenlightening and better left to the theoretical statisticians.

where μ is the mean of r and σ is the standard deviation of r.

Now, assume that $\phi : \mathcal{E} \to \{0,1\}^{240}$ is a random map. Then with high probability, r will have a distribution given by the sum of $|\mathcal{E}|$ Bernoulli random variables with mean $1/2^{240}$, up to some small error. For this distribution,

$$\sigma^2 = |\mathcal{E}| \left(\frac{1}{2^{240}}\right)\left(1 - \frac{1}{2^{240}}\right),$$

so the mean of s is

$$\mu + \frac{\sigma^2}{\mu} = \mu + |\mathcal{E}| \left(\frac{1}{2^{240}}\right)\left(1 - \frac{1}{2^{240}}\right)\frac{2^{240}}{\#\mathcal{E}} = \mu + 1 - \frac{1}{2^{240}} = 2^{16} + 1 + O(2^{-16}). \qquad \square$$

In fact, the same assumption can be used to prove that the standard deviation of the distribution of $\phi(b)$ across both truly random 240-bit blocks b and Dual EC DRBG 240-bit blocks b is 256.

Claim 2.2. *Suppose that $\phi : \mathcal{E} \to \{0,1\}^{240}$ is modeled as a random map $\mathcal{E} \to \{0,1\}^{240}$. Then with high probability,*

(a) The distribution of $\phi(b)$ across truly random 240-bit blocks b has variance $2^{16} + O(2^{-16})$.

(b) The distribution of $\phi(b)$ across Dual EC DRBG 240-bit blocks b has variance $2^{16} + O(2^{-16})$.

Proof. The variance of r is

$$\left(\sum_{i\in\mathbb{N}} i^2 r(i)\right) - \left(\sum_{i\in\mathbb{N}} i r(i)\right)^2.$$

The variance of s is

$$\left(\sum_{i\in\mathbb{N}} i^2 s(i)\right) - \left(\sum_{i\in\mathbb{N}} i s(i)\right)^2 = \frac{(\sum_{i\in\mathbb{N}} i r(i))(\sum_{i\in\mathbb{N}} i^3 r(i)) - (\sum_{i\in\mathbb{N}} i^2 r(i))^2}{(\sum_{i\in\mathbb{N}} i r(i))^2}.$$

By assumption, r is a binomial distribution with $n = |\mathcal{E}|$ and probability $p = \frac{1}{2^{240}}$, so r's variance is

$$np(1-p) \approx \frac{|\mathcal{E}|}{2^{240}} \approx 2^{16} + O(2^{-16}).$$

To calculate the variance of s, first recall the moments

$$\begin{aligned} u_1 &= np \\ u_2 &= n(n-1)p^2 + np \\ u_3 &= n(n-1)(n-2)p^3 + 3n(n-1)p^2 + np. \end{aligned}$$

Thus, s's variance is

$$\begin{aligned}
&\frac{np(n(n-1)(n-2)p^3 + 3n(n-1)p^2 + np) - (n(n-1)p^2 + np)^2}{(np)^2} \\
&= (n-1)(n-2)p^2 + 3(n-1)p + 1 - ((n-1)p+1)^2 = (n-1)p(1-p) \\
&\approx \frac{|\mathcal{E}|}{2^{240}} \approx 2^{16} + O(2^{-16}).
\end{aligned}$$

□

3 Twisted Dual EC DRBG

3.1 The Twisted Dual EC DRBG algorithm

This section will define the complete Twisted Dual EC DRBG algorithm for the security parameter $\lambda = 384$. For other security parameters λ, one should choose suitable curves over finite fields of size $\approx 2^\lambda$ along with points on those curves.

Let $\mathcal{E}$ be the P-384 elliptic curve, which is defined by the equation $y^2 = x^3 - 3x + B$ over $\mathbb{F}_p$ with parameters given in hexadecimal by

$$\begin{aligned}
p &= 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1 \\
&= \texttt{0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffeffffffff0000000000000000ffffffff} \\
B &= \texttt{0xb3312fa7e23ee7e4988e056be3f82d19181d9c6efe8141120314088f5013875ac656398d8a2ed19d2a85c8edd3ec2aef.}
\end{aligned}$$

The number of points on $\mathcal{E}$ is

$$\texttt{ffffffffffffffffffffffffffffffffffffffffffffffffc7634d81f4372ddf581a0db248b0a77aecec196accc52973},$$

which is prime. Additionally, let $\mathcal{E}'$ denote the curve

$$y^2 = -x^3 + 3x - B = (-x)^3 - 3(-x) - B.$$

This is a quadratic twist of $\mathcal{E}$ because $p \equiv 3 \pmod 4$ implies -1 is a quadratic nonresidue modulo p. The number of points on $\mathcal{E}'$ is

$$\begin{aligned}
\#\mathcal{E}' &= 2p + 2 - \#\mathcal{E} \\
&= \texttt{0x100000000000000000000000000000000000000000000000389cb27e0bc8d21ea7e5f24bb74f58851313e697333ad68d}
\end{aligned}$$

which is also prime.

Now, let P and Q be the points on $\mathcal{E}$ given by coordinates

$$\begin{aligned}
P_x &= \texttt{0x43a8df013becd277e4025216bc90897fde4ca279023e0a4e44e56e8901a6c99760b571f069a57870fbe2056786383c5f} \\
P_y &= \texttt{0xb3b25f84b93f6e08fff60c6c2a54a265d6011fc2dc1e4b77f3225615f083371e8336b0cf482a8359ccf23701bd2ee36c} \\
Q_x &= \texttt{0x1c36145bead2c4d731d8c42e5f2f99ff6863828635fa58039baeaad0b7cb1f2f1a327c68e4a9e365a02787f213f69873} \\
Q_y &= \texttt{0xb823b33bfcf9984ec3238e0651299a0c40739bdf10fee3ed97573747a85fa0d1d46d9a40a23bf25bed06afc63f2daaa9},
\end{aligned}$$

and let P' and Q' be the points on $\mathcal{E}'$ given by coordinates

$$P'_x = \texttt{0x37bd2d02280518a1a17231c472c31167df26db67a66067274f1bc82b0cdaa5d7f80150bb0e8d8d26fd9dd95143e180c1}$$
$$P'_y = \texttt{0x5ddc86454aab639bf39efa0efd3fe6a4568bbe2708844b370e1083b8047868c6c9d75fefcae4a2eb8009319d37745a14}$$
$$Q'_x = \texttt{0x4a1d7984226f32ae743c629d1a738192d190d7807bc11e25ea3c83fd5c41b228b256d475e8c1a945b7c93d6c834973aa}$$
$$Q'_y = \texttt{0x9d4c10281697f6ceb4d48166d6fe88ca04e2e57f39b7dec79314163e421e004c10ba1f21d9930c2060d0acbdf6ad6016.}$$

The Twisted Dual EC DRBG takes a seed $s_0 \in \mathbb{F}_p$ and a positive integer k as input, and it outputs $384k$ pseudorandom bits. It is given by the following algorithm, which keeps track of an internal state $s \in \mathbb{F}_p$. Initially, the internal state s is set to the seed s_0. Then, the following steps are repeated k times:

- Let a be the last bit of s and let b be the second-to-last bit of s.
- If $b = 0$, output all 384 bits of the x-coordinate of sQ. If $b = 1$, output all 384 bits of the x-coordinate of sQ'.
- If $a = 0$, replace s with the x-coordinate of sP. If $a = 1$, replace s with the x-coordinate of sP'.

Note that the output step and the state update step have been swapped from the original algorithm.

Intuitively, this modified algorithm avoids the bias that Schoenmakers and Sidorenko's distinguisher takes advantage of because every number in $\mathbb{F}_p$ is the x-coordinate of some point in $\mathcal{E}$ or some point in $\mathcal{E}'$ (and no number is an x-coordinate of points in both curves).

3.2 The Twisted Dual EC DRBG Backdoor

The Twisted Dual EC DRBG Backdoor is very similar to the backdoor given in Section 2.3. If an adversary knows an integer k for which $P = kQ$, or an integer k' for which $P = kQ'$, then the adversary can recover the secret state from any 384-bit block of Twisted Dual EC DRBG output. This allows them to predict all future output with 100% accuracy. In fact, the backdoor algorithm is faster than the one given in Section 2.3 because *all* bits of the x-coordinate of sQ or sQ' are revealed in the Twisted Dual EC DRBG algorithm.

To see why, suppose the adversary knows an integer k for which $P = kQ$. Let the old secret state be s. If last two bits of s, then the algorithm outputs the x-coordinate of sQ and updates the secret state to the x-coordinate of sP. In this case, an adversary can compute the x-coordinate of

$$sP = skQ = k(sQ)$$

as the next secret state. On average, the last two bits of the secret state are both zero $\frac{1}{4}$ of the time, so an adversary knowing k can expect to read four 384-bit blocks before recovering the secret state.

If the adversary also knows an integer k' for which $P' = k'Q'$, then the adversary can also recover the next secret state if the current secret state's last two bits are both one. In

this case, an adversary knowing both k and k' can expect to read two 384-bit blocks before recovering the secret state.

3.3 Security Proof

3.3.1 Setup and Hardness Assumptions

We first enumerate the assumptions necessary to prove the security of our PRG.

We assume that p is in $[2^\lambda - 2^{\lambda/2}, 2^\lambda]$, so that $|2^\lambda - p| \leq \sqrt{p}$. Let $q = \#\mathcal{E}$ and $q' = \#\mathcal{E}'$. Assume that q and q' are distinct primes.Let $\mathfrak{E}$ denote the elliptic curve defined by the same equation as $\mathcal{E}$ but over $\mathbb{F}_p^2$ instead of $\mathbb{F}_p$. Because $\mathfrak{E}$ has subgroups isomorphic to $\mathcal{E}$ and $\mathcal{E}'$, and because q and q' are distinct primes, it follows that $\mathfrak{E} \simeq \mathcal{E} \times \mathcal{E}' \simeq \mathbb{Z}/qq'\mathbb{Z}$. Note that the projection of $\mathfrak{E}$ onto $\mathcal{E}$ can be computed efficiently; it simply requires multiplication by $q'(q'^{-1} \pmod q)$. Similarly, the projection of $\mathfrak{E}$ onto $\mathcal{E}'$ is efficiently computable.

We define the following problems:

- The *Decisional Diffie-Hellman Problem (DDHP)* on $\mathcal{E}$ with base point P is the problem of distinguishing (aP, bP, cP) and (aP, bP, abP) for uniformly random a, b, and c in $[0, q)$. This problem is similarly defined for $\mathcal{E}'$ with basepoint P'.
- The *Decisional k-bounded Discrete Logarithm Problem (k-DBDLP)* on $\mathfrak{E}$ with base point R is the problem of distinguishing aR, where a is uniform from $[0, k)$, and bR, where b is uniform from $[0, qq')$.

Concretely, the DDHP is believed to take time at least $\Theta(\sqrt{q})$ to solve. The authors were unable to find any references for the latter problem, but believe that the naive $\Theta(\sqrt{k})$ algorithm of running the baby-steps giant-steps algorithm with an upper bound of k is the most efficient way to solve it.

3.3.2 Single-output security

We first prove that a single round of our DRBG is a secure pseudorandom generator.

Lemma 3.1. *Assume that* $\mathrm{Gen}(1^\lambda)$ *chooses appropriate* $\mathcal{E}$, $\mathcal{E}'$, P, *and* P', *and also chooses* Q *and* Q' *uniformly at random on* $\mathcal{E}$ *and* $\mathcal{E}'$.

Sample $param \leftarrow \mathrm{Gen}(1^\lambda)$ *and* $s_0 \leftarrow \{0,1\}^\lambda$. *Let* r_1 *be the first block of* λ *bits of randomness output by the Twisted Dual EC DRBG using parameters param and seed* s_0, *and let* s_1 *be the new internal state. Then*

$$(param, s_1, r_1) \approx_c (param, \$, \$).$$

Proof. **Step 1: Replacing bitstrings with random $\mathbb{F}_p$ elements.**

First, because $|2^\lambda - p| \leq \sqrt{p}$, a random element of $\{0,1\}^\lambda$ and a random element of $\mathbb{F}_p$ are statistically indistinguishable, with a statistical distance of $O(1/\sqrt{p})$. Therefore, we can

replace s_0 with s_0', a random element of $\mathbb{F}_p$, and if r_1' and s_1' are the new values of r_1 and s_1, and y and y' are uniform elements of $\mathbb{F}_p$, it suffices to show that

$$(\text{param}, s_1', r_1') \approx_c (\text{param}, y, y')$$

Step 2: Proving indistinguishability from random elliptic curve points, for fixed a and b.

From now on we fix $\mathcal{E}$, $\mathcal{E}'$, P, and P', and prove that

$$(Q, Q', s_1', r_1') \approx_c (Q, Q', y, y').$$

Let a be the last bit of s_0' and let b be the second-to-last bit of s_0'. Let t_0 be the remaining bits of s_0', so $s_0' = a + 2b + 4t_0$.

Recall that s_1' and r_1' are both x-coordinates of points on either $\mathcal{E}$ or $\mathcal{E}'$. Further, a and b determine which curve is used for s_1' and r_1' respectively. We claim that for a fixed choice of a and b,

$$(Q, Q', r_1', s_1') \approx_c (Q, Q', z, z')$$

where z is the x-coordinate of a uniform point on the curve indicated by a and similarly for z'. Let k be the integer in $[0, q)$ such that $Q = kP$ and let k' be the integer in $[0, q')$ such that $Q' = k'P'$.

First we consider the case where $a = b$. Say that $a = b = 0$; the other case is similar. Then

$$\begin{aligned}(Q, Q', r_1', s_1') &= (kP, k'P', x(s_0'kP), x(s_0'P)) \\ &\approx_c (kP, k'P', z, x(s_0P)) \\ &\approx_c (Q, Q', z, z')\end{aligned}$$

where c is a random element of $[0, q)$. The first equivalency follows from the assumed hardness of the DDHP and the fact that k' is independent of everything else in the distribution. (We are ignoring the fact that the lower two bits of s_0' are fixed, since a distinguisher when two bits of s_0' are fixed still works a quarter as often in general.) The second equivalency follows from assuming $p/4$-BDBLP on $\mathcal{E}$ with basepoint $4P$.

We now consider the harder case where $a \neq b$. Say that $a = 0$ and $b = 1$. We first observe that it suffices to show $(Q, Q', s_0'Q', s_0'P) \approx_c (Q, Q', j'P', jP)$, where $j' \in [0, q')$ and $j \in [0, q)$.

Recalling the efficiently computable isomorphism of $\mathfrak{E}$ and $\mathcal{E} \times \mathcal{E}'$, let $R = Q' + P \in \mathfrak{E}$, which is a generator of $\mathfrak{E}$. Let $Z \in \mathfrak{E}$ be uniformly random. We see that it suffices to show $(Q, Q', s_0'R) \approx_c (Q, Q', Z)$. But this follows from p-BDBLP on $\mathfrak{E}$.

Step 3: Providng indistinguishability of random elliptic curve points and random $\mathbb{F}_p$ elements.

At this point we are essentially left with the following problem:

Let r_1' be the x-coordinate of a random point on $\mathcal{E}$ with probability $1/2$ and the x-coordinate of a random point on $\mathcal{E}'$ with probability $1/2$. Choose s_1' in the same way. Prove that (r_1', s_1') is indistinguishable from two random elements of $\mathbb{F}_p$.

It turns out that these distributions are actually statistically indistinguishable. By the Weil bounds, q and q' are in $[p+1-2\sqrt{p}, p+1+2\sqrt{p}]$, so r_1' and s_1' have statistical distance $O(1/\sqrt{p})$ from a distribution where they are chosen uniformly from $\mathcal{E} \cup \mathcal{E}'$.

The equation for $\mathcal{E}$ is $y^2 = x^3 - 3x + B$, and the equation for $\mathcal{E}$ is $-y^2 = x^3 - 3x + B$. -1 is a quadratic nonresidue modulo p, so for any value of x, either there will be two solutions in y for the first equation and 0 for the second, vice versa, or 0 will be the unique solution for both. Therefore the uniform distribution of x-coordinates of points in $\mathcal{E} \cup \mathcal{E}'$ is actually the same as the uniform distribution on $\mathbb{F}_p$ so we are done. □

3.3.3 Multi-output security

We now prove security over many outputs.

Theorem 3.2. *The Twisted Dual EC DRBG with randomized parameters is a secure DRBG.*

Proof. If $r_1, r_2, \ldots, r_m$ are the output blocks, we must show

$$(Q, Q', r_1, r_2, \ldots, r_m) \approx_c (Q, Q', \$, \$, \ldots, \$)$$

This is a simple hybrid argument. For $0 \leq i \leq m$ let H_i be a hybrid where r_1 through r_i are truly random and r_{i+1} through r_m are generated with the DRBG from a random seed. By Lemma 3.1, H_i and H_{i+1} are indistinguishable, so we are done. □

Remark 3.3. *In fact, the proof still carries through if we add the secret state s_m to the distribution, which proves that this DRBG is forward secure. This is the reason for swapping the order of the randomness output and state update steps as compared to the original.*

Remark 3.4. *Assuming optimal hardness for ECDLP and DBDLP, the scheme as described offers essentially a $\lambda/2$-bit security level.*

3.4 Runtime Analysis

In order to generate $384k$ bits, the Twisted Dual EC DRBG must perform $2k$ scalar multiplications on the elliptic curve: k multiplications $s \to x(sP)$ to update the state and k multiplications $s \to x(sQ)$ to generate the output.

Using the Montgomery Ladder algorithm for elliptic curve multiplication, each scalar multiplication can be performed using approximately $2 \cdot 384$ additions.

Therefore, the Twisted Dual EC DRBG can generate 1 bit for every 4 elliptic curve additions on average.

3.5 Implementation

Our implementation of the Twisted Dual EC DRBG and its backdoor can be found at `https://github.com/aredheron/Twisted-Dual-EC-DRBG/`.

Some notes about the implementation:

- The implementation uses the tinyec library, which means that the algorithm used for scalar multiplication may be susceptible to side channel attacks.
- The x-coordinates of the parameters (points) on the twisted curve are negative those specified in 3.1. This is because the tinyec library requires curves to be specified by monic cubics, in this case $y^2 = x^3 + ax - b$ instead of $y^2 = -x^3 - ax - b$.

4 Future Directions

There are several directions that this project could take in the future.

- Our Twisted Dual EC DRBG specification only gives explicit curves and points for $\lambda = 384$. It would be nice to have explicit curves and points for other values of the security parameter λ. However, in practice this is difficult because of three constraints:
  - The underlying finite field should have order extremely close to a power of 2.
  - Both the elliptic curve and its quadratic twist must have prime order.
  - Both the elliptic curve and its quadratic twist must be safe curves.

  Notably, the curve we chose for $\lambda = 384$ is not a safe curve because its constants could have been manipulated by the U.S. government. Ideally, we would choose a safer curve, but finding one is computationally expensive.
- Investigate variants involving different extractions of a and b from the secret state s.
- Rigorize the meaning of "high probability" in Claims 2.1 and 2.2.
- Develop a more detailed implementation of the Twisted Dual EC DRBG resistant to side channel attacks.
  - Scalar multiplication on an elliptic curve can be implemented in a way that works directly with the x-coordinates, which resists against side channel attacks and speeds up the DRBG.

5 Work Distribution

Holden devised the algorithm and worked on writing the paper. Linus worked on the implementation and runtime analysis. Noah worked on the security analysis.

References

[1] Information technology – Security techniques – Random bit generation, November 2005.

[2] Bill Barker and Schneier Bruce. Letter to Bruce Schneier from Bill Barker – Wired Commentary. https://github.com/matthewdgreen/nistfoia/blob/master/6.4.2014%20production/109%20-%20Nov%2028%202007%20Letter%20to%20Bruce%20from%20Barker%20-%20Wired%20Commentary%20.pdf, nov 2007. Retrieved from GitHub FOIA archive.

[3] Elaine Barker and John Kelsey. Recommendation for Random Number Generation Using Deterministic Random Bit Generators, 2006-06-13 2006.

[4] Daniel Bernstein, Tanja Lange, and Ruben Niederhagen. *Dual EC: A Standardized Back Door*, volume 9100, pages 256–281. 03 2016.

[5] Kristian Gjøsteen. Comments on Dual-EC-DRBG/NIST SP 800-90, Draft December 2005, 04 2006.

[6] Don B. Johnson. X9.82 Part 3: Number Theoretic DRBGs. https://csrc.nist.gov/csrc/media/events/random-number-generation-workshop-2004/documents/numbertheoreticdrbg.pdf, July 2004. Presented at the NIST Random Number Generation Workshop, Gaithersburg, MD, July 20, 2004.

[7] Joint Technical Committee ISO/IEC JTC 1, Subcommittee SC 27. US National Body Comments on ISO/IEC 2nd CD 18031. https://projectbullrun.org/dual-ec/documents/us-comment-to-iso.pdf, 2003. Attachment 10 to SC27 N3685.

[8] Joseph Menn. Exclusive: Secret contract tied NSA and security industry pioneer. *Reuters*, December 2013. Accessed: 2025-05-04.

[9] National Institute of Standards and Technology. NIST Removes Cryptography Algorithm from Random Number Generator Recommendations, April 2014. Accessed: 2025-05-04.

[10] Nicole Perlroth, Jeff Larson, and Scott Shane. N.S.A. Foils Much Internet Encryption, September 2013. The New York Times.

[11] Bruce Schneier. Did NSA Put a Secret Backdoor in New Encryption Standard? *WIRED*, November 2007.

[12] Berry Schoenmakers and Andrey Sidorenko. Cryptanalysis of the Dual Elliptic Curve Pseudorandom Generator. *IACR Cryptology ePrint Archive*, 2006:190, 01 2006.

[13] Dan Shumow and Niels Ferguson. On the Possibility of a Back Door in the NIST SP800-90 Dual Ec Prng. Presented at the CRYPTO 2007 Rump Session, August 2007. Slides available online.