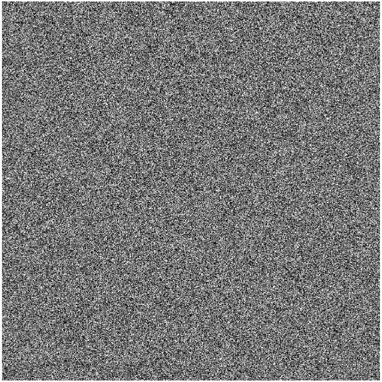
Elliptic Curve DRNGs



Elliptic Curve DRNGs

- Linear Congruential Generator
- Power Generator
- Naor-Reingold Generator
- Dual EC RNG

Elliptic Curve DRNGs

- Weierstrass form of elliptic curves has been the standard tool
- Interesting applications of character sums, combinatorics, and curves
- Requirement R1 is usually assumed
- Requirement R2: Security proofs of elliptic curve DRNGs are based on the elliptic curve discrete logarithm problem:

$$\boxed{\text{Given } P \text{ and } Q\text{, compute } d \text{ in } Q = [d]P}$$

Sequence from Points

- Map elliptic curve points $P_j = (x_j, y_j) \in \mathcal{E}(a, b, p)$ into $[0, 1) \times [0, 1)$
- Since $x_j, y_j \in \mathrm{GF}(p)$, there is a natural map

$$P_j \to \left(\frac{x_j}{p}, \frac{y_j}{p} \right)$$

  since $\mathrm{GF}(p)$ consists of field elements $\{0, 1, \ldots, p-1\}$
- Some applications use only the x coordinate or apply maps to the coordinate values, for example, hash functions or trace maps

Elliptic Curve Linear Congruential Generator

- Start with the initial value $Q_0 \in \mathcal{E}(a, b, p)$
- Compute the sequence

$$Q_j = P \oplus Q_{j-1}$$

  for $j = 1, 2, \ldots$
- No security implied
- Given Q_{j-1}, we compute $Q_j = P \oplus Q_{j-1}$
- Given Q_j, we compute $Q_{j-1} = Q_j \ominus P$

Elliptic Curve Linear Congruential Generator

- Easy to construct the next element given the current one and P

$$\begin{aligned}
Q_1 &= P \oplus Q_0 \\
Q_2 &= P \oplus Q_1 = P \oplus (P \oplus Q_0) = [2]P \oplus Q_0 \\
Q_3 &= P \oplus Q_2 = P \oplus ([2]P \oplus Q_0) = [3]P \oplus Q_0 \\
&\vdots \\
Q_j &= P \oplus Q_{j-1} = P \oplus ([j-1]P \oplus Q_0) = [j]P \oplus Q_0
\end{aligned}$$

- Given Q_j and Q_0 the computation of j is the ECDLP, provided that j is sufficiently large

Elliptic Curve Linear Congruential Generator

- Let $Q_j = (x_j, y_j)$ and use $(x_j)_{j=0}$ as sequence in GF(p) or normalize it to $[0, 1)$ using an enumeration of the field and dividing by p
- The period of the LCG is related to the number of points in $\mathcal{E}$

Elliptic Curve Power Generator

- For integer $d \geq 2$, consider the sequence starting with $Q_0 = P$

$$Q_j = [d]Q_{j-1} = [d^j \bmod n]P$$

- Here n is the order of P

$$\begin{aligned}
Q_1 &= [d]Q_0 = [d]P \\
Q_2 &= [d]Q_1 = [d]([d]P) = [d^2 \bmod n]P \\
Q_3 &= [d]Q_2 = [d]([d^2 \bmod n]P) = [d^3 \bmod n]P \\
&\vdots \\
Q_j &= [d]Q_{j-1} = [d]([d^{j-1} \bmod n]P) = [d^j \bmod n]P
\end{aligned}$$

- Computing d given Q_j and Q_{j-1} is equivalent to the ECDLP

Elliptic Curve Naor-Reingold

- A point on the curve $P \in \mathcal{E}(a, b, p)$ with order n
- An integer vector of dimension m defined as $A = (a_1, a_2, \ldots, a_m)$
- The elements $a_i \in [1, n]$ for $i = 1, 2, \ldots, m$,
- Consider the m-dimension of the binary vector representation of the integer $d = (d_1, d_2, \ldots, d_m)$ with $d < 2^m$, such that $d_i \in \{0, 1\}$
- Consider the integer valued function $f(d, A)$ based on d and A as

$$f(d, A) = \prod_{i=1}^{m} a_i^{d_i} = a_1^{d_1} \cdot a_2^{d_2} \cdots a_m^{d_m}$$

- For example, given $m = 4$, $A = (2, 5, 3, 4)$, and $d = (0, 1, 1, 1)$

$$f(7, A) = 2^0 \cdot 5^1 \cdot 3^1 \cdot 4^1 = 60$$

Elliptic Curve Naor-Reingold

- Compute the sequence of points on $\mathcal{E}(a, b, p)$ defined as

$$Q_{d,A} = [f(d, A) \bmod n]P$$

for $d = 1, 2, \ldots, 2^m - 1$

- For example, given the group order $n = 17$, and the same m and A as above, we compute $Q_{d,A}$ for $d = 1, 2, \ldots, 2^m - 1$ as

$$\begin{array}{lclcl}
f(1, A) & \leftarrow & 2^0 \cdot 5^0 \cdot 3^0 \cdot 4^1 = 4 & \rightarrow & [4]P \\
f(2, A) & \leftarrow & 2^0 \cdot 5^0 \cdot 3^1 \cdot 4^0 = 3 & \rightarrow & [3]P \\
f(3, A) & \leftarrow & 2^0 \cdot 5^0 \cdot 3^1 \cdot 4^1 = 12 & \rightarrow & [12]P \\
f(4, A) & \leftarrow & 2^0 \cdot 5^1 \cdot 3^0 \cdot 4^0 = 5 & \rightarrow & [5]P \\
f(5, A) & \leftarrow & 2^0 \cdot 5^1 \cdot 3^0 \cdot 4^1 = 20 & \rightarrow & [20]P = [3]P \\
 & \vdots & & &
\end{array}$$

Research on EC-LCG, EC-PG, and EC-NRG

- Recent work of Tanja Lange, David Kohel, Igor Shparlinski, Berry Schoenmakers, and Vladimir Sidorenko
- Some results of theoretical and practical value
- Other results are more practical
- If the order of P is at least $p^{0.5+\epsilon}$ then all three sequences (LCG, Power and Naor-Reingold) are reasonably well distributed

Dual EC Random Number Generator

- Dual EC RNG is an algorithm to compute pseudorandom numbers starting from some random seed
- It was first time presented in 2004 at a NIST workshop
- Dual EC RNG was standardized by NIST in early 2006, and subsequently appeared in ANSI and ISO standards, among other algorithms to generate deterministic random numbers
- Dual EC RNG was in dozens of commercial cryptographic software libraries
- It was even the default deterministic number generator in RSA Security's BSAFE library








Dual EC RNG Algorithm

- Dual EC RNG algorithm is based on the NIST approved curves with two associated points P and Q on them
- The original standard document, NIST Special Publication 800-90A by the authors E. Barker and J. Kelsey, recommends the use of NIST P-256, P-384, and P-521
- The points P and Q are special points on the curve, generated according to the specification described in the Publication 800-09A

Dual EC RNG Algorithm based on NIST P-256

- The field is GF(p) for $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$
- The elliptic curve $y^2 = x^3 + ax + b$
- The parameters a and b, and the group order n are given as

$$
\begin{aligned}
a &= -3 \bmod p \\
&= 115792089210356248762697446949407573530 \\
&\quad\; 086143415290314195533631308867097853948 \\
b &= 410583637251521421293261297800472684091 \\
&\quad\; 144410159937255548352563140394674012891 \\
n &= 115792089210356248762697446949407573529 \\
&\quad\; 996955224135760342422259061068512044369
\end{aligned}
$$

Dual EC RNG Algorithm based on NIST P-256

- The Dual EC algorithm uses two points on the curve P and Q whose coordinates are given as

$$
\begin{aligned}
P_x &= 48439561293906451759052585252797914202762949526041747995844080717082404635286 \\
P_y &= 36134250956749795798585127919587881956611106672985015071877198253568414405109 \\
\\
Q_x &= 91120319633256209954638481795610364441930342474826146651283703640232629993874 \\
Q_y &= 80764272623998874743522585409326200078679332703816718187804498579075161456710
\end{aligned}
$$

Dual EC RNG Algorithm based on NIST P-256

- The algorithm starts with a seed $s_0 \in \{0, 1, \ldots, n-1\}$
- Let $\mathrm{LSB}_i(s)$ denote the least significant i bits of s
- For example, $\mathrm{LSB}_3(23) = 7$ since $23 = (10\underline{111})_2$
- The point multiplications are performed using the points P and Q over the curve NIST P-256

input: $s_0 \in \{0, 1, \ldots, n-1\}$ and $k > 0$
output: $240k$ bits
for $i = 1$ to k
  $s_i = x$ coordinate of $[s_{i-1}]P$
  $t_i = x$ coordinate of $[s_i]Q$
  $r_i = \mathrm{LSB}_{240}(t_i)$
return: $r_1, r_2, \ldots, r_k$

Security of the Dual EC RNG Algorithm

- The security of the Dual EC RNG seems to depend on the difficulty of the ECDLP
- Given s_{i-1}, the computation of s_i in "$s_i = x$ coordinate of $[s_{i-1}]P$" is the point multiplication problem: **easy**
- Given s_i, the computation of s_{i-1} in "$s_i = x$ coordinate of $[s_{i-1}]P$" is the elliptic curve discrete logarithm problem: **hard**
- Backtracking (in other words, predicting) is defined as discovering a previous value in the sequence

Security of the Dual EC RNG Algorithm

- Prediction is equivalent to distinguishing the output of the deterministic random number generator from the sequence of uniformly distributed random bits
- It was first shown by Gjosteen and then by Schoenmakers and Sidorenko in 2006 that the output of the Dual EC RNG can be efficiently distinguished from the sequence of uniformly distributed random bits
- The distinguishing attack does not imply solving the ECDLP
- This means that the Dual EC RNG is **insecure** and cannot be used for cryptographic purposes

InSecurity of the Dual EC RNG Algorithm

- Furthermore, Shumow and Ferguson announced in 2007 that there was a "possibility of a back door" in Dual EC
- Shumow and Ferguson explained a way for whoever had generated the special points P and Q to start from one random number produced by Dual EC RNG, and predict all subsequent random numbers
- By the end of 2007, in the view of the public cryptographic community, Dual EC RNG was dead and gone

InSecurity of the Dual EC RNG Algorithm

- The media picked up the story in 2013, however, it had a twist :)
- The source was Snowden, and in particular reports on Project Bullrun and the SIGINT Enabling Project

  Cryptographers have long suspected that the agency planted vulnerabilities in a standard adopted in 2006 by the NIST and later by the ISO, which has 163 countries as members. Classified NSA memos appear to confirm that the fatal weakness, discovered by cryptographers in 2007, was engineered by the agency.

- The surprise for the public cryptographic community was not so much this confirmation of what had already been suspected, but rather that NSA's back-dooring of Dual EC RNG was part of an organized approach to weakening cryptographic standards

InSecurity of the Dual EC RNG Algorithm

- Not mentioned in the reports was the biggest surprise, namely that Dual EC was not dead at all
- A list of "validations" published by NIST showed that Dual EC RNG was provided in dozens of commercial cryptographic software libraries
- Dual EC RNG was even the default pseudorandom number generator in RSA Security's BSAFE library
- Reuters reported that NSA paid RSA $10 million in a deal that set Dual EC RNG as the default method for number generation in the BSAFE library