On the Possibility of a Back Door in the NIST SP800-90 Dual Ec Prng

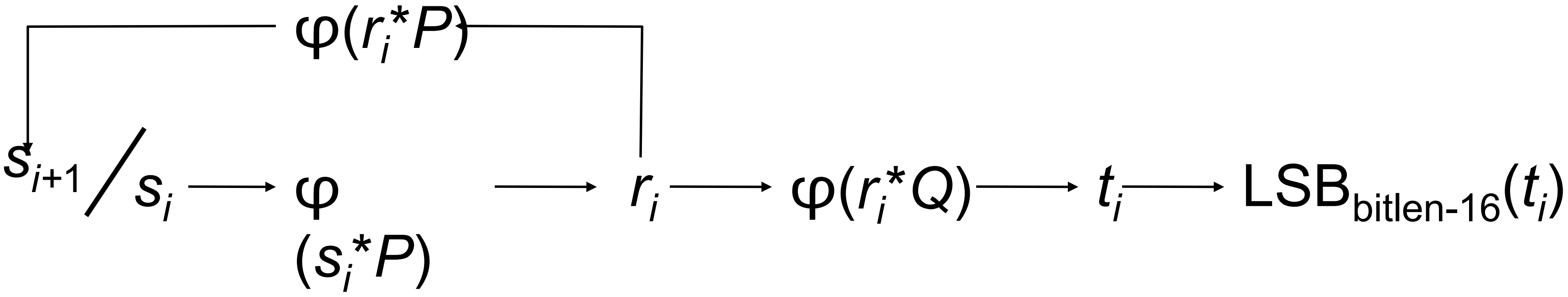
Dan Shumow

Niels Ferguson

Microsoft

The Dual Ec PRNG

- φ : prime curve → integers
  $\varphi(x,y) = x$
- P, Q points on the curve (per SP800-90)



Equations:

$r_i = \varphi(s_i{*}P)$ $t_i = \varphi(r_i{*}Q)$ $s_{i+1} = \varphi(r_i{*}P)$

The Objection

- Point *P* is generator of the curve (per SP800-90).
- Point *Q* is a specified constant. It is not stated how it was derived.
- NIST prime curves have prime order. So there exists *e* such that $Q^e = P$.

The Attack

- Output: S, the set of possible values of s_{i+1} the internal state of the Dual Ec PRNG at the subsequent step.
- Suppose an attacker knows value e.

```
Given: a block of output $o_i$ from a Dual EC PRNG Instance
Set $S = \{\}$.
For $0 \le u \le 2^{16} - 1$
    $x = u|o_i$
    $z \equiv x^3 + ax + b \bmod p$.
    If $y \equiv z^{1/2} \bmod p$ exists => $A = (x,y)$ is on the curve
        $S = S \cup \{\varphi(e*A)\}$.
```

How this works:

- One of the values $x = t_i$

  If A is the point with x coordinate t_i then:

  $$A = r_i * Q$$

  Thus:

  $$\varphi(e*A) = \varphi(e* r_i * Q) = \varphi(r_i * P) = s_{i+1}.$$

  => s_{i+1} is in S.
- $|S| \approx 2^{15}$

Experimental Verification

1. Pick NIST P-256 Curve
2. Chose random *d*
3. Chose $Q_2 = d*P$
4. Replace Q with Q_2
5. Given |Output| = 32 > out block length
6. Filter out s_{i+1} values that do not generate next 2 bytes.

In every experiment 32 bytes of output was sufficient to uniquely identify the internal state of the PRNG.

The Main Point

- If an attacker knows *d* such that *d*P* = *Q* then they can easily compute *e* such that *e*Q* = *P* (invert mod group order)
- If an attacker knows *e* then they can determine a small number of possibilities for the internal state of the Dual Ec PRNG and predict future outputs.
- We do not know how the point *Q* was chosen, so we don’t know if the algorithm designer knows *d* or *e*.

Conclusion

- WHAT WE ARE NOT SAYING:
  NIST intentionally put a back door in this PRNG
- WHAT WE ARE SAYING:
  The prediction resistance of this PRNG (as presented in NIST SP800-90) is dependent on solving one instance of the elliptic curve discrete log problem.
  (And we do not know if the algorithm designer knew this before hand.)

Suggestions for Improvement

- Truncate off more than the top 16 bits of the output block.
  - Results on extractors from *x* coordinates of EC points of prime curves suggest truncating off the top bitlen/2 bits is reasonable.
- Generate a random point *Q* for each instance of the PRNG.