A flaw in a theorem about Schnorr signatures

Daniel R. L. Brown[*]

May 26, 2015

Abstract

An alleged theorem of Neven, Smart and Warinschi (NSW) about the security of Schnorr signatures seems to have a flaw described in this report.

Schnorr signatures require representation of an element in a discrete logarithm group as a hashable bit string. This report describes a defective bit string representation of elliptic curve points. Schnorr signatures are insecure when used with this defective representation. Nevertheless, the defective representation meets all the conditions of the NSW theorem.

Of course, a natural representation of an elliptic curve group element would not suffer from this major defect. So, the NSW theorem can probably be fixed.

1 Background

Neven, Smart and Warinschi [NSW09, §2] describe the generic group model based on Shoup's well-known generic model:

> Let G be an abstract group of prime order q, which we will shall write additively; one can think of G as the group of integers modulo q under addition. In particular we do not make any assumption as to whether discrete logarithms are hard to compute in G. We let $s = \lceil \log_2 q \rceil$.
>
> In a cryptographic scheme elements of G are encoded by bit strings of length ℓ. Solving the discrete logarithm problem is essentially equivalent to discovering the precise encoding used, it is this intuition which sits behind the generic group model.

[*]Certicom/BlackBerry, dbrown@certicom.com

> We let $\mathbb{G}$ be the set of bit strings of length ℓ, and we let $\tau : G \to \mathbb{G}$ be the "natural representation" of G in $\mathbb{G}$. We shall represent the induced group operation on the set $\tau(G) \subset \mathbb{G}$ multiplicatively. Thus we will use additive notation for the representation in which discrete logarithms might be easy (since we think of this representation as the addition group of integers modulo q) and we use multiplicative notation for the representation in which believe discrete logarithms to be hard.

They next describe the notion of a conversion function and a security condition upon conversion functions that they call "almost-invertibility":

> In a large number of protocols one needs to also map group elements in $\mathbb{G}$ to the smaller set $\{0,1\}^d$ for $d \leq \ell$, often because ℓ is too large for practical use, or for other efficiency reasons. We therefore assume the existence of a "conversion" function $f : \mathbb{G} \to \{0,1\}^d$. The conversion function does not necessarily preserve any properties of the group law in $\tau(G)$, nor is it necessarily invertible (see the examples below). An important quantity in our analysis is the *conversion density*
>
> $$\delta = \frac{|f(\tau(G))|}{2^d}.$$
>
> We call the function f "almost-invertible" if there is an efficient randomized algorithm which given a random bit string $R \xleftarrow{\$} \{0,1\}^d$, with probability δ, computes a preimage $\mathfrak{R} \in \tau(G) \subset \mathbb{G}$ such that $f(\mathfrak{R}) = R$.
>
> The conversion function f plays a crucial role in Brown's analysis of EC-DSA [8,9], and its existence explains the distinct difference between the existence of a proof of security of EC-DSA in the generic group model and the absence of one for normal DSA. In [8,9] a similar definition of "almost-invertible" is given for the conversion function f, however Brown's definition is stricter than what we will need. Our analysis only requires the relatively weak definition given above.

Refer to this condition on the conversion function as the *Neven–Smart–Warinschi almost-invertibility* (NSWAI). As noted above, NSWAI relaxes a stricter original [Bro05] definition of almost-invertibility of the conversion function.

They then define several security conditions for hash functions, whose details are not essential to this report. Finally, they state and prove their main (alleged) theorem:

> **Theorem 6.1.** *Let* $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n \hookrightarrow \mathbb{Z}_q$ *be some hash function, and let* G *be some fixed group modelled as a generic group over the set of bit strings* $\mathbb{G}$*. Let* $f : \mathbb{G} \to \{0,1\}^d$ *be an almost-invertible conversion function. If the* $\mathsf{rpp}[D]$ *and the* $\mathsf{rpsp}[D]$ *are hard for* H*, with respect to the domain* $D = \{0,1\}^d$*, then the Schnorr signature scheme* $\mathsf{Sch}[\mathsf{H}]$ *is secure in the generic group model.*

This report shows that, strictly speaking, their main theorem appears to be wrong. The reason is that a defective NSWAI conversion function makes Schnorr signatures easy to forge.

In short, the defect is that the conversion function can have an output that clusters to one particular too often, and still be NSWAI. Indeed, this report exhibits a defective conversion function that is almost constant. Clustering leads to a forgery. The following few sections discuss this idea more formally.

2 Defective conversion function

The defective conversion function will assume that d is smaller than $s = \lceil \log_2 q \rceil$. For an example, we will take $d = 128$ while $s = 256$. In this setting, only 128 bits representing a discrete logarithm group element are actually hashed.

Encode bit strings in $\{0,1\}^d$ as non-negative integers between 0 and $2^d - 1$ inclusive, in the natural manner: we generally identify the bit strings with the corresponding integers for the sake of comparison and so on. Fix some set $S \subseteq \tau(G) \subseteq \mathbb{G}$ of size 2^d, written as $\{s_0, s_1, \ldots, s_{2^d-1}\}$, such that s_i is easily determined from i: a specific example will given later below. Fix some integer o with $0 \leq o \leq 2^d - 1$.

Abstractly, the defective conversion function is:

$$f : \mathbb{G} \to \{0,1\}^d : g \mapsto f(g) = \begin{cases} i & \text{if } g = s_i, \\ o & \text{if } g \notin S. \end{cases} \tag{1}$$

To make this concrete, one must define S and so on.

For a concrete example, suppose that the group G being modelled is the elliptic curve group NIST P256. The set $\mathbb{G}$ consists of bit strings of length

512 bits: with the two consecutive 256-bit sub-strings, representing finite field coordinates (x, y) per the short Weierstrass coordinates that standards use to specify NIST P256.

Suppose that $d = 128$. To form s_i from i in this setting, working in bit string form, set $x = i\|j$ for the smallest 128-bit string j such that there exists y with point (x, y) on the curve. Each $x = i\|j$ can be tested by computing that the appropriate Legendre symbol $\left(\frac{x^3+ax+b}{p}\right) = 1$. For valid x, solve for y using the quadratic equation, and choose the smaller of the two y candidates. This defines s_i unless no such j exists. If no such j exists, then s_i is undefined, and so is f.

The choice of o does not matter for the analysis, but the sake of concreteness, choose $o = 0$.

Assume a heuristic that, for each x, there is an independent probability $1/2$ that y exists. Under this heuristic, the probability that, for each i, that no j exists is about $(1/2)^{\#j} = 2^{-2^{128}}$. Taken over all i, the heuristic probability that any s_i, and thus f, is undefined, is about $(\#i)(1/2)^{\#j} = 2^{-2^{128}+2^7}$, which is overwhelmingly small. We henceforth assume the technicality that f is well-defined function, with the heuristic justification above.

So, to compute f, proceed as follows. Take input of (x, y) represented as a bit string of length 512 bits. Consider the first 128 bits, as a bit string i. Then consider the next 128 bits as bit string j. If j is the smallest possible such that $x = i\|j$ is valid as an x-coordinate of a point in the group NIST P256, then output $f(x, y) = i$. Otherwise output $f(x, y) = o$.

Disproportionately many points (x, y) will have $f(x, y) = o$: so one can say that f is clustered at output value o, or o-clustered.

We assume henceforth that the defective conversion function f is described as above.

3 Defective conversion forgery

The next theorem shows that if 2^d is sufficiently smaller than the size q of the discrete logarithm group, then universal (any-message) forgery of the defective conversion function version of Schnorr signatures is efficient.

Theorem 1. *Given message m, do the following:*

1. *Choose random integer s such that $0 \leq s < q$.*
2. *Compute $h = \mathsf{H}(o\|m)$.*
3. *Output signature (s, h).*

Under a given public key pk and group generator $\mathfrak{g}$, the pair (s, h) is a valid signature of message m with probability at least

$$1 - 2^d/q.$$

Proof. Recall the definition of Schnorr signature verification [NSW09, §3]:

```
$\mathsf{Vfy}(pk, m, (s, h))$:
    $R \leftarrow f(\mathfrak{g}^s \cdot pk^{-h})$
    If $\mathsf{H}(R \| m) = h$
    Then return 1
    Else return 0.
```

Because s is drawn independently of pk, $\mathfrak{g}$, and h, and because $\mathfrak{g}$ has order q, it follows that

$$g = \mathfrak{g}^s \cdot pk^{-h}$$

is, as a random variable, uniformly distributed in the discrete logarithm group $\tau(G)$. It follows that the probability that $g \in S$ is at most $|S|/|\tau(G)| = 2^d/q$.

Therefore, with probability at least $1 - 2^d/q$, the event $g \notin S$ occurs. In this event, $R = f(g) = o$, by definition of f. In this event, $\mathsf{H}(R\|m) = \mathsf{H}(o\|m) = h$, by definition of h. So, $\mathsf{Vfy}(pk, m, (s, h))$ returns 1, and the signature (s, h) is valid. □

Putting $d = 128$ for $q \approx 2^{256}$ provides an overwhelming probability of success for this forger. Even putting $d = 2^{255}$ gives the forger an unacceptably high success rate of approximately 1/2 for NIST P256.

This forger is a universal (selective) forger, not just an existential forger, in that it can sign any message given to it, not just some artificial junk message as an existential forger might. This forger is a passive (key-only or no-message) forger in that it requires no signature queries from the signer to help make its forgery work. In fact, this forger does not even need the signer's public key: it is a no-key forger. Consequently, its forgeries are valid across multiple keys.

4 Defective conversion is NSWAI

Theorem 2. *The defective (o-clustered) conversion function f that compresses the NIST P256 elements to 128 bits is Neven–Smart–Warinschi almost-invertible (NSWAI), under a heuristic assumption stated below in the proof.*

Proof. First consider the conversion density: $\delta = |f(\tau(G))|/2^d$ of the conversion function f. From the definition of S, we have $S \subseteq \tau(G)$, so $\{0,1\}^d = f(S) \subseteq f(\tau(G))$. Consequently, the conversion density is $\delta = 1$.

Next, consider the task of inverting f. Given $i = R \xleftarrow{\$} \{0,1\}^d$, we output s_i. By definition of f, we have $f(s_i) = i = R$, so inversion has been achieved. In the abstract definition of f, we have stipulated the recovering s_i from i must be easy.

In the more specific example of the elliptic curve group NIST P256, we parsed the x-coordinate as $x = i \| j$. A heuristic estimate for the cost of recovering s_i from i is, on average, $2 (= 1 + \frac{1}{2} + \frac{1}{4} + \dots)$ Legendre symbol computations: to test each candidate j. This average cost is the heuristic mentioned in the theorem statement above.

Computing an average of two Legendre symbols is fairly efficient, thereby satisfying the requirement of Neven–Smart–Warinschi almost invertibility. $\square$

This defective conversion function f fails to meet the stricter, original notion of almost-invertibility that was used in a security proof for ECDSA [Bro05]. The stricter definition has a condition that the resulting inversion is computationally indistinguishable from a uniformly distributed group element. The inversion of f described above always yields an element of S. But elements of S are easily distinguished from random elements of $\tau(G)$.

5 An alternative defect

The example defective conversion function f is highly compressive: squeezing a 512-bit elliptic curve representation down to 128 bits as input to the hash. If we had instead compressed to d bits instead, Theorem 1 still seems to depend on $2^d \leq q$, which is a form of compression. It is therefore natural to ask if some kind of compression is essential to the defect.

To help answer that question, one can consider the following alternative compression function f'. We start off by changing $\mathbb{G}$ to already include point compression, in which the y coordinate is replace by a single bit z which allows recovery of y from x in some standardized manner. For NIST P256, we set $\mathbb{G} = \{0,1\}^{257}$. Set $d = 257$, fix some $o \in \{0,1\}^d$, and define conversion function:

$$f'(u) = \begin{cases} o & \text{if } \mathsf{H}(u) \equiv o \bmod 2^{64}, \\ u & \text{otherwise.} \end{cases} \tag{2}$$

We leave it to the reader to verify that f' is NSWAI and that the Schnorr using f' is forgeable with an average cost of about 2^{64} evaluations of H. This form of f' may also guide the reader in correcting any flaws in the proof of the theorem.

One might object that the conversion f' involves point compression within $\mathbb{G}$. So, instead let $\mathbb{G}$ not use point compression, but otherwise define f'' similarly to f', per (2), using the identity function except with a 2^{-64} probability clustering to value o. In this case, a random element of $\{0,1\}^d$ has probability at most $\delta = \tau(G)/2^{512} \approx 2^{-256}$ of being invertible under f''. So the conversion density is extremely small, yet f'' still meets the NSWAI condition. Neither the NSWAI condition nor the NSW main theorem statement place any condition on the size of δ, so f'' serves the same role as f or f'. The actual proof of the NSW theorem uses the quantity δ internally to bound the effectiveness of the reduction: this dependency on δ was not carried over to the theorem statement. So, the conversion function f'' does not contradict the internals of the proof, even though it contradicts the theorem statement.

6 Discussion

6.1 Hash requirements

Neven, Smart and Warinschi describe some security conditions [NSW09, §4] on hash functions. These requirement are a hypothesis of the main NSW theorem.

The forgeability under the defective conversion function f falsifies the main NSW theorem conclusion. Of course, the theorem conclusion could fail if any one of the theorem hypotheses fails. We argued that the NSWAI hypothesis is met by f.

It is possible that hash requirements hypothesis fails, in which case the theorem might, strictly speaking, be correct. So, suppose that the theorem is actually correct, strictly speaking. Because the defective conversion function does not involve the hash function H, this could only happen if the NSW hash requirements fail for all hash functions H. Such a failure of all hash functions seems implausible. But, for the sake of an argument, suppose that, indeed, all hashes fail. In that disastrous scenario, the main NSW theorem would hold vacuously, but it would be a totally useless result by never successfully implying its conclusion.

It seems quite reasonable to presume that NSW hash requirements are feasible for some hash function. In that case, the NSW main theorem seems flawed, and needs some kind of correction.

6.2 Where is the proof flaw?

As noted above, the defective conversion forgery strongly suggests that the NSW theorem is false, strictly speaking. If so, then its proof must have a flaw.

Intuitively, it would seem that the flaw in the proof should occur soon after the step where f is inverted. The inversion step can be achieved with the defective conversion function, but the resulting bias towards the elements of set $S \subset \tau(G)$ may somehow undermine the logic of the proof, such as by invalidating the properties of the generic group model, whereby the success rate of the reduction unravels. Abstractly, it seems that a bias in a point representation automatically cancels the protections provided by the generic group model, but that is not a fully satisfactory explanation. Rather, it would be ideal to trace the exact point where this unraveling occurs: why the reduction fails to defeat the hash function when the conversion function's inversion is biased, or at what point do the low probabilities ensured by the generic group model break down.

6.3 How can the theorem be fixed?

A nature first guess at a fix would be to restore the original stricter definition of almost-invertibility from [Bro05]. Perhaps, with this simple fix, the current proof of the NSW theorem will apply to the fixed statement, without any further modification (as is).

Immediate acceptance of such an assertion, without first seeing a clear identification of the flawed proof step(s), would seem slightly imprudent.

6.4 Impact on the ECDSA proof

In elliptic curves with co-factor two or more, the standard ECDSA conversion functions can be inverted easily, but the inverse suffers from a small bias because some bit strings may represent two x-coordinates, some may represent one x-coordinate. Although this does not seem to lead to any sort of attack, it is perhaps a violation of the strict reading of the ECDSA proof in [Bro05]. This suggests further review of [Bro05] is merited.

6.5 Rigor of the flaw

The flaws as described in this report rest on some plausible heuristic assumptions, which have been explicitly described in this report. It would be preferable to find a completely rigorous proof of the flaws, that does not depend on such heuristics: future work may identify such examples.

6.6 Another gap in the NSW proof?

Another possible gap (unrelated to the NSWAI condition) in the NSW proof concerns the handling of the generic group model. Shoup's model, and the version described in the NSW introduction does not include any oracle outputs in the additive group G. Signature queries seem to include values related to the group G: the s values. The same issue was encountered in [Bro05], and was addressed by amending the generic group model slightly by introduction of a new type of query, called a *hint* query. A lemma from [Bro05] verifies that additional hint queries do not undermine the fundamental guarantees provided by the generic group model. As with the the flawed NSWAI condition, it is natural to suspect that this potential gap in the proof of the NSW theorem, can be filled in by following the earlier approach from [Bro05] more closely.

6.7 Implausibility of academic subversion

The recent Snowden leaks suggest that NSA has been subverting the security of cryptographic standards. Security proofs are sometimes cited in standards development organizations to support one algorithm over another. So, an extreme skeptic might inspect any such security proof for some deception. For example, one might ask: could the flaw in the NSW theorem have deceived some implementers into using an insecure conversion function with Schnorr signatures? There seems to be no good reason why any implementer would deliberately choose a clustered conversion function, so the answer to that question seems to be no.

6.8 Progress in provability

It is inevitable that errors will occur in a few proof attempts, and therefore the focus should be on error correction.

Although the NSW theorem conclusion fell severely, in a reading of their theorem in isolation, it might seem only to be a simple oversight. One could say that the theorem was merely stated without sufficient rigor, by omitting an exceptional degeneracy condition. This would be like stating Fermat's marginal conjecture (now Wiles' theorem) as $a^n+b^n=c^n$ as having no integer solutions while forgetting to mention some technical conditions (such $abc \neq 0$ and $n \geq 3$) needed to avoid some trivial counterexamples. Such a misstatement is correct in spirit, but wrong in details.

Alas, the NSW introduction emphasizes the fact that the NSWAI condition is less strict (and simpler) than previous condition. In other words, they make a point of not requiring the critical conditions that are needed to avoid degenerate exceptions. So, in reading the paper as a whole, it seems that that the flaw is a bit more than an oversight.

The intuition that Schnorr signature's feature of hashing the point representation provide an extra layer of protection against clustered conversion functions is somewhat plausible on the surface, but unfortunately does not pan out.

7 Conclusion

Neven, Smart and Warinschi made some overstatements about the security of Schnorr signatures, including a theorem statement that appears to be false. Although it seems likely that their theorem would be fixed merely by reverting to the stricter definition of previous research, it seems prudent to first fully identify the flaw in their current proof.

The theoretical attack in this report would never arise in practice. The defective conversion function used in the theoretical attack is too contrived for any implementer to have considered using it. The main impact of the theorem flaw is a question of rigor and the assurances that security proofs aim to provided to cryptography users.

Acknowledgments

Alfred Menezes, Neal Koblitz and Nigel Smart provided some useful comments on drafts of this report.

References

[Bro05] Daniel R. L. Brown. Generic groups, collision resistance, and ECDSA. *Designs, Codes and Cryptography*, 35:119–152, 2005. `http://eprint.iacr.org/2002/026`.

[Bro14] Daniel R. L. Brown. Schnorr v clustered f. Personal email, July 2014.

[MLPSS02] John Malone-Lee, David Pointcheval, Nigel P. Smart, and Jacques Stern. Flaws in applying proof methodologies to signature schemes. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 93–110. Springer-Verlag, 2002.

[NSW09] Gregory Neven, Nigel P. Smart, and Bogdan Warinschi. Hash function requirements for Schnorr signatures. *Journal of Mathematical Cryptology*, 3(1):69–87, January 2009.

[Sho01] Victor Shoup. OAEP reconsidered. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 239–259. Springer-Verlag, 2001. `http://eprint.iacr.org/2000/060`.