StackExchange sign up log in tour help

Cryptography Stack Exchange is a question and answer site for software developers, mathematicians and others interested in cryptography. It's 100% free, no registration required. Sign up ×

# Should we trust the NIST-recommended ECC parameters?

Recent articles in the media, based upon Snowden documents, have suggested that the NSA has actively tried to enable surveillance by embedding weaknesses in commercially-deployed technology -- including at least one NIST standard.

The NIST FIPS 186-3 standard provides recommended parameters for curves that can be used for elliptic curve cryptography. These recommended parameters are widely used; it is widely presumed that they are a reasonable choice.

**My question.** Can we trust these parameters? Is there any way to verify that they were generated in an honest way, in a way that makes it unlikely they contain backdoors?

**Reasons for concern.** Bruce Schneier has written that he has seen a bunch of secret Snowden documents, and after seeing them, he recommends classical integer discrete log-based cryptosystems over elliptic curve cryptography. When asked to elaborate on why he thinks we should avoid elliptic-curve cryptography, he writes:

> I no longer trust the constants. I believe the NSA has manipulated them through their relationships with industry.

This suggests we should look closely at how the "constants" (the curve parameters) have been chosen, if we use ECC. This is where things look concerning. I recently read a message on the tor-talk mailing list that seems to suggest the NIST curve parameters were not generated in a verifiable way. That message examines how the parameters were generated:

> I looked at the random seed values for the P-xxxr curves. For example, P-256r's seed is c49d360886e704936a6678e1139d26b7819f7e90. No justification is given for that value.

and ultimately concludes:

> I now personally consider this to be smoking evidence that the parameters are cooked.

Based upon my reading of FIPS 186-3, this appears to be an accurate description of the process by which the P-xxxr curves were generated. So, should people be concerned about this? Or is this just paranoia based upon loss of trust in the NSA?

See also these slides from Dan Bernstein and Tanja Lange, particularly pp.6-7, 8-10, 14-17 for further discussion about the NIST parameter choices.

elliptic-curves backdoors nist nsa

edited Sep 10 '13 at 23:26

asked Sep 9 '13 at 3:07
D.W.
22.8k ● 4 ● 32 ● 82

1 I never trusted them in the first place, if you want an alternative, try Shamus Standard Curves – Richie Frame Sep 9 '13 at 3:12

4 Since the seed isn't short (why???) the creator could have tried many curves (say $2^{70}$) in hopes of finding a weak one. So the question is if there is a significant fraction of $b$ values for which the curve is weak, and the NSA knew about that weakness back when the parameters were generated. – CodesInChaos ♦ Sep 9 '13 at 7:04

7 When you ask "can we trust these parameters", I see in front of me Schneier's comment "Cryptographers are a conservative bunch: We don't like to use algorithms that have even a whiff of a problem". Since there is a legitimate concern now among the memebrs of the community who are tasked with applying the standards in real-life products, the onus is on the standards body, namely NIST, to come up with a good explanation about the source of these parameters. As long as they decline, I'd say to avoid anything based on those parameters. – Ninveh Sep 9 '13 at 9:39

5 As for "smoking guns", I say that the whole proceedings reek more of some bureaucratic blunder than foul play. Chances are that whoever produced the seeds just used random bytes from some PRNG in all good faith, and once the parameters were out it was too late to change them. At least half of my daily work environment was produced that way... – Thomas Pornin Sep 9 '13 at 18:19

4 @ThomasPornin: These curves were recommended by NIST, with input from NSA, in the early 2000s. A "bureaucratic blunder" by those folks, in a document with this purpose, has near-zero probability IMO. The magic numbers in SHA-1 (for instance) are clear "nothing up my sleeve" values. These should and could have been just as clear. – Nemo Sep 9 '13 at 21:58

## 3 Answers

**Edit:** I have made some tests and I found something weird. See at the end.

**Initial answer:**

At least the **Koblitz curves** (K-163, K-233... in NIST terminology) cannot have been specially "cooked", since the whole process is quite transparent:

- Begin with a binary field $GF(2^m)$. For every *m* there is only one such field (you can have several representations, but they are all isomorphic).
- Restrict yourself to prime values of *m* to avoid possible weaknesses by plunging into sub-fields.
- Consider curves $Y^2 + XY = X^3 + aX^2 + b$ where $b \neq 0$; this is the normal form of non-supersingular curves in binary fields.
- You only want curves where $a = a^2$ and $b = b^2$, so that you can speed up computations with the Frobenius endomorphism (basically, you replace point doublings with simply squaring both coordinates, which is very fast).
- When $a = 0$, the curve order is necessarily a multiple of 4; when $a = 1$, necessarily a multiple of 2.

Then you want a curve order which is "as prime as possible", i.e. equal to $2p$ or $4p$ for a prime $p$ (depending on whether $a = 1$ or $0$). For $m$ ranging in the "interesting range" (say 160 to 768), you will not find a lot of suitable curves (I don't remember the exact count, but it is something like 6 or 7 curves). NIST simply took the 5 of them corresponding to the $m$ values which were closest to (but not lower then) their "security levels" (80, 112, 128, 192 and 256-bit "equivalent strength"). There is no room for "cooking" here.

So I would say that at least Koblitz curves are demonstrably free from all these "cooking" rumours. Of course, some other people argue that Koblitz curves have some special structure which might be leveraged for faster attacks; and that's true in two ways:

- Faster computations mean faster attacks, mechanically;
- One can solve discrete logarithm "modulo the Frobenius endomorphism" which means that K-233 is about as strong as a 225-bit curve (because 233 is an 8-bit number).

I still consider such curves to be reasonable candidates for serious cryptographic work. They have been "in the wild" for more at least 15 years and are still unscathed, which is not bad, as these things go.

---

**Edit:** I have made a few tests, enumerating all Koblitz curves in $GF(2^m)$ for $m$ ranging from 3 to 1200. For each $m$, there are two curves to test, for $a = 0$ and $a = 1$. We consider the curve "appropriate" if its order is equal to $4p$ (for $a = 0$) or $2p$ (for $a = 1$) with $p$ prime (this is the "best possible" since the curve is always an extension of the same curve in $GF(2)$, so the curve order is necessarily a multiple of the curve in $GF(2)$, and that's 4 or 2, depending on $a$). For the "interesting range" of $m$ between 160 and 768, there are **fourteen appropriate curves**:

- $m = 163, a = 1$
- $m = 233, a = 0$
- $m = 239, a = 0$
- $m = 277, a = 0$
- $m = 283, a = 0$
- $m = 283, a = 1$
- $m = 311, a = 1$
- $m = 331, a = 1$
- $m = 347, a = 1$
- $m = 349, a = 0$
- $m = 359, a = 1$
- $m = 409, a = 0$
- $m = 571, a = 0$
- $m = 701, a = 1$

NIST's target was their five "security levels" of 80, 112, 128, 192 and 256 bits, and a curve would match that level only if its size is at least twice the level. So the standard curve for each level ought to be the smallest curve which is large enough for that level. This should yield Koblitz curves in fields of size 163, 233, 277, 409 and 571 bits, respectively.

Strangely enough, this matches NIST's choices *except* for the "128-bit" level, in which they chose $m = 283$ instead of $m = 277$. I don't know the reason for this. For both field sizes, the smallest possible reduction polynomial is a pentanomial ($X^{277} + X^{12} + X^6 + X^3 + 1$ for $m = 277$, $X^{283} + X^{12} + X^7 + X^5 + 1$ for $m = 283$), so neither field is at a computational advantage on the other is using polynomial bases (well, the 277-bit field is a bit shorter, so a bit faster). With normal bases, the 277-bit field is actually more efficient, because it accepts a "type 4" Gaussian normal basis, while the 283-bit field is a "type 6" (smaller types are better for performance). The list of all suitable Koblitz curves is easy to rebuild and, indeed, NIST / NSA did it (e.g. see this article from NSA-employed J. A. Solinas -- search for "277").

Why they chose the 283-bit field is mysterious to me. I still deem it very improbable that this constitutes "cooking"; this is a backdoor only if NIST (or NSA) knows how to break Koblitz curves in a 283-bit field and not in a 277-bit field, which not only requires an assumption of "unpublished big cryptanalytic advance", but also requires that supposed novel breaking technique to be quite weird.

edited Sep 29 '13 at 17:15

answered Sep 9 '13 at 18:15

Thomas Pornin
**39k** ● 6 ● 88 ● 158

2 Is there any possibility ( for any of the curves) that they picked them to be hard to get constant time operations on? – imichaelmiers Sep 11 '13 at 5:24

2 That's improbable. In the curve equations, the $a$ parameter is used only for doublings, and $b$ is not used at all, meaning that it cannot impact computation speed. For the P-* curves, $a = -3$, which saves one operation in the doublings, making it actually slightly *easier* to get constant-time operations. – Thomas Pornin Sep 11 '13 at 11:00

(That Tor mailing list link appears to be broken at the moment)

Your question is at least partially answered in FIPS 186-3 itself…

**Appendix A** describes how to start with a seed and use an iterative process involving SHA-1 until a valid elliptic curve is found.

**Appendix D** contains the NIST recommended curves and includes the seed used to generate each one according to the procedure in *Appendix A*.

So to believe that NSA cooked the constants, you would have to believe one of two things: Either they can invert SHA-1, or a sufficient fraction of curves would have to meet their hidden conditions that they could find appropriate SEED values by a brute force search.

Customarily, "nothing up my sleeve" constructions start with something simple, like the $sin$ (for MD5) or $sqrt$ (for SHA-1) of small integers. To my knowledge (am I wrong?), the SEED values for the NIST curves are not so easily described, which is itself arguably suspicious.

On the other hand, these are the curves commercial software must support to receive FIPS certification, allowing it to be purchased by U.S. government agencies and used for the protection of classified data.

So if NSA did cook the constants, they did a moderately good job of hiding it, and they have some confidence that other people will not find the holes any time soon.

The Bernstein/Lange criticisms are based on other properties, like how easy it is to botch an implementation using the NIST curves.

That said...

The preponderance of evidence from the latest revelations suggest NSA knows **something** cryptographically relevant about SSL/TLS. Maybe that means ECDHE, and maybe not. (Heck, maybe it just means certain common implementations.)

But given that we have alternatives from the likes of Dan Bernstein (Curve25519), I see no compelling reason to use NIST's curves even if you want to ignore Schneier's gut feeling to avoid ECC altogether.

[Update]

The Bernstein/Lange presentation says the NIST elliptic curves were created by "Jerry Solinas at NSA". I missed that on the first reading.

I have sent this question to Perry Metzger's cryptography list:

http://www.metzdowd.com/pipermail/cryptography/2013-September/017446.html

Maybe somebody can get in touch with Mr. Solinas and ask him how he chose the seed values and why. It would be interesting to hear the answer from the source, even if nobody is likely to believe it.

[Update 2]

See also http://safecurves.cr.yp.to/rigid.html

edited Oct 15 '13 at 23:20

answered Sep 9 '13 at 4:24

Nemo
**730** ● 4 ● 13

2 "or a sufficient fraction of curves would have to meet their hidden conditions that they could find appropriate SEED values by a brute force search." I would not be surprised if this method was used to create a weakened curve. – Richie Frame Sep 9 '13 at 5:48

ECDHE, by my understanding, is not heavily deployed in TLS and hence wouldn't amount to a lot of data being intercepted . Certainly not the massive amount from an "enormous breakthrough" that according to James Bamford may have necessitated Bluffdale. – imichaelmiers Sep 11 '13 at 5:29

@imichaelmiers: ECDHE is the default for current OpenSSL, which means it is the default for any modern Linux Web server, browser permitting. (Go to self-evident.org in Chrome and click the lock icon. I am using the defaults for that server.) I do not know about current IIS and IE. I made this a question, but no answers so far. I think the "breakthrough" is a combination of EC knowledge (possibly just about the NIST curves) plus convincing everyone to start switching to ECC. But then I am just some crank... – Nemo Sep 11 '13 at 15:32

If the NSA knew a sufficiently large weak class of elliptic curves, it is possible for them to have chosen weak curves and have them standardized.

As far as I can tell, there is no hint about any sufficiently large class of curves being weak.

Regarding choosing the curves: It would have been better if NIST had used an "obvious" string as the seed, e.g. "seed for P-256 no. 1", "seed for P-256 no. 2", etc., incrementing the counter until a good (according to the specified criteria) was found. (We know that NSA and NIST know about and use "obvious" strings from the constants in (say) SHA-1.)

Should we take the fact that they did not do it like this as evidence that they know a large class of weak curves? When an honest person generates the curves, choosing random seeds is just as good as "obvious" strings. It seems reasonable that an honest person did not anticipate the current paranoia level, and therefore did not choose "obvious" strings, but just generated some randomness. This is therefore not evidence that NSA knows about a large class of weak elliptic curves, because of the simpler explanation that is a mistake.

Should we use the NIST curves today? We now have 13 more years of experience and the uncertainty brought on by these leaks. The Bernstein-Lange slides suggests that the NIST curves are not the best choice (curves exist where faster arithmetic is easier to implement correctly and securely). We should not hesitate to make better choices now.

Bruce Schneier's suggestion to avoid elliptic curves seems like overkill, but Schneier never liked elliptic curves.

edited Sep 10 '13 at 13:33

answered Sep 9 '13 at 6:43

K.G.
2,200 ● 3 ● 16

I'm not looking to fault anyone or to avoid ECC. I just want to know whether there are justifiable grounds for trusting the NIST ECC parameters. I don't think that's unreasonable or harsh. (I think the world of NIST; they are truly a boon to society, and I have great trust and respect for the NIST employees that I've gotten to know. But that doesn't make the question go away; there is still the question of whether there are justifiable technical grounds for confidence in these parameters.) – D.W. Sep 9 '13 at 6:57

2 Contrary to what you wrote, choosing random seeds is not just as good, because it provides no way to demonstrate that they didn't cook the parameters. Suppose that, say, 1/1000000000 of curves are vulnerable to some obscure attack that the NSA has discovered but that is not known to the public world. Then choosing an arbitrary seed would let them select a curve that is secretly vulnerable to their secret attack. (For what it's worth, the person who chose the ECC parameters for NIST was apparently... wait for it... a NSA employee.) – D.W. Sep 9 '13 at 6:58

@user7863: I suppose you were the same one as who proposed the edit ... you can claim ownership of your post again by registering an account, using the same mail address. You might try to do this from the same browser where you posted the answer, then it gets easier ... otherwise there's a bit more bureaucracy involved. – Paŭlo Ebermann Sep 9 '13 at 17:23

2 "It seems reasonable that an honest person did not anticipate the current paranoia level" I don't think so, as the discussion of the DES S-boxes is much older, and other people used nothing-up-my-sleeve numbers as a matter of course. – starblue Sep 11 '13 at 18:48

**protected** by Community ♦ Sep 16 '13 at 16:32

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site.

Would you like to answer one of these unanswered questions instead?