NSA Backdoors and Bitcoin

Many cryptographic standards widely used in commercial applications were developed by the U.S. Government's National Institute of Standards and Technology (NIST). Normally government involvement in developing ciphers for public use would throw up red flags, however all of the algorithms are part of the public domain and have been analyzed and vetted by professional cryptographers who know what they're doing. Unless the government has access to some highly advanced math not known to academia, these ciphers should be secure.

We now know, however, that this isn't the case. Back in 2007, Bruce Schneier reported on a backdoor found in NIST's Dual_EC_DRBG random number generator:

But today there's an even bigger stink brewing around Dual_EC_DRBG. In an informal presentation(.pdf) at the CRYPTO 2007 conference in August, Dan Shumow and Niels Ferguson showed that the algorithm contains a weakness that can only be described as a backdoor.

This is how it works: There are a bunch of constants - fixed numbers - in the standard used to define the algorithm's elliptic curve. These constants are listed in Appendix A of the NIST publication, but nowhere is it explained where they came from.

What Shumow and Ferguson showed is that these numbers have a relationship with a second, secret set of numbers that can act as a kind of skeleton key. If you know the secret numbers, you can predict the output of the random-number generator after collecting just 32 bytes of its output. To put that in real terms, you only need to monitor one TLS Internet encryption connection in order to crack the security of that protocol. If you know the secret numbers, you can completely break any instantiation of Dual_EC_DRBG.

This is important because random number generators are widely used in cryptographic protocols. If the random number generator is compromised, so are the ciphers that use it.

Thanks to the heroic work of Edward Snowden we now know that Dual_EC_DRBG was developed by the NSA, with the backdoor, and given to NIST to disseminate. The scary part is that RSA Security, a company that develops widely used commercial encryption applications, continued use of Dual_EC_DRBG all the way up to the Snowden revelations despite the known flaws. Not surprising this brought a lot of heat on RSA which denies they intentionally created a honeypot for the NSA.

UPDATE: RSA was paid $10 million by the NSA to keep the backdoor in there.

All of this has been known for several months. What I didn't know until reading Vitalik Buterin's recent article Satoshi's Genius: Unexpected Ways in which Bitcoin Dodged Some Crytographic Bullets, is that a variant of an algorithm used in Bitcoin likely also contains a NSA backdoor, but miraculously Bitcoin dodged the bullet.

Bitcoin uses the Elliptic Curve Digital Signature Algorithm (ECDSA) for signing transactions. This is how you use your private key to "prove" you own the bitcoins associated with your address. ECDSA keys are derived from elliptic curves that themselves are generated using certain parameters. NIST has been actively recommending that everyone use the secp256r1 parameters because they "are the most secure". However, there appears to be some funny business with secp256r1 that is eerily similar to the backdoor in Dual_EC_DRBG.

Secp256r1 is supposed to use a random number in generating the curves. The way it allegedly creates this random number is by using a one-way hash function of a "seed" to produce a nothing up my sleeve number. The seed need not be random since the output of the hash function is not predictable. Instead of using a relatively innocuous seed like, say, the number 15, secp256r1 uses the very suspicious looking seed: c49d360886e704936a6678e1139d26b7819f7e90. And like Dual_EC_DRBG, it provides no documentation for how or why this number was chosen.

Now as Vitalik pointed out, even if the NSA knew of a specific elliptic curve with vulnerabilities, it still should have been near impossible for them rig the system due to the fact that brute-forcing a hash function is not feasible. However, if they discovered a flaw that occurred in say, one curve in every billion, then they only need to test one billion numbers to find the exploit.

However, the kicker in all this is that the parameters for secp256r1 were developed by the head of elliptic curve research at the NSA!

The unbelievable thing is that rather than using secp256r1 like nearly all other applications, Bitcoin uses secp256k1 which uses Koblitz curves instead of pseudorandom curves and is still believed to be secure. Now the decision to use secp256k1 instead of secp256r1 was made by Satoshi. It's a mystery why he chose these parameters instead of the parameters used by everyone else (the core devs even considered changing it!). Dan Brown, Chairman of the Standards for Efficient Cryptography Group, had this to say about it:

I did not know that BitCoin is using secp256k1. Indeed, I am surprised to see anybody use secp256k1 instead of secp256r1.

Just wow! This was either random luck or pure genius on the part of Satoshi. Either way, Bitcoin dodged a huge bullet and now almost seems destined to go on to great things.