

# A Few Thoughts on Cryptographic Engineering

Some random thoughts about crypto. Notes from a course I teach. Pictures of my dachshunds.

Thursday, September 5, 2013

## On the NSA

Let me tell you the story of my tiny brush with the biggest [crypto story of the year](#).

A few weeks ago I received a call from a reporter at [ProPublica](#), asking me background questions about encryption. Right off the bat I knew this was going to be an odd conversation, since this gentleman seemed convinced that the NSA had vast capabilities to defeat encryption. And not in a 'hey, d'ya think the NSA has vast capabilities to defeat encryption?' kind of way. No, the defeating was a given. We were just haggling over the details.

Oddness aside it was a fun (if brief) set of conversations, mostly involving hypotheticals. If the NSA could do this, how might they do it? What would the impact be? I admit that at this point one of my biggest concerns was to avoid coming off like a crank. After all, if I got quoted sounding *too much* like an NSA conspiracy nut, my colleagues would laugh at me. Then I might not get invited to the cool security parties anymore.

All of this is a long way of saying that I was totally unprepared for today's bombshell revelations [describing the NSA's efforts to defeat encryption](#). Not only does the worst possible hypothetical I discussed appear to be true, but it's true on a scale I couldn't even imagine. I'm no longer the crank. I wasn't even close to cranky enough.

And since I never got a chance to see the documents that sourced the NYT/ProPublica story -- *and I would give my right arm to see them* -- I'm determined to make up for this deficit with sheer speculation. Which is exactly what this blog post will be.

### 'Bullrun' and 'Cheesy Name'

If you haven't read the [NYT](#) or [Guardian](#) stories, you probably should. The TL;DR is that the NSA has been doing some very bad things. At a combined cost of \$250 million per year, they include:

1. Tampering with national standards ([NIST](#) is specifically mentioned) to promote [weak, or otherwise vulnerable](#) cryptography.
2. Influencing standards committees to weaken protocols.
3. Working with hardware and software vendors to weaken encryption and random number generators.
4. Attacking the encryption used by 'the next generation of 4G phones'.
5. Obtaining cleartext access to 'a major internet peer-to-peer voice and text communications system' (Skype?)
6. Identifying and cracking vulnerable keys.
7. Establishing a Human Intelligence division to infiltrate the global telecommunications industry.
8. And worst of all ([to me](#)): somehow decrypting SSL connections.

All of these programs go by different code names, but the NSA's decryption program goes by the name 'Bullrun' so that's what I'll use mostly use here.

### How to break a cryptographic system

There's almost too much here for a short blog post, so I'm going to start with a few general thoughts. Readers of this blog should know that there are basically three ways to break a cryptographic system. In no particular order, they are:

1. **Attack the cryptography.** This is difficult and unlikely to work against the standard



## About Me



### Matthew Green

I'm a cryptographer and research professor at Johns Hopkins University. I've designed and analyzed cryptographic systems used in wireless networks, payment systems and digital content protection platforms. In my research I look at the various ways cryptography can be used to promote user privacy.

[My website](#)  
[Guide to this blog](#)  
[My twitter feed](#)  
[Useful crypto resources](#)  
[RSS](#)  
[Bitcoin tipjar](#)  
[View my complete profile](#)

## Popular Posts

### [Dear Apple: Please set iMessage free](#)

Normally I avoid complaining about Apple because (a) there are plenty of other people carrying that flag, and (b) I honestly like Apple ...



### [Here come the encryption apps!](#)

It seems like these days I can't eat breakfast without reading about some new encryption app that will (supposedly) revolutionize our c...



### [Zerocoin: making Bitcoin anonymous](#)

This is what it's like to die of stupid. Wow, what the heck is going on with Bitcoin? When I started this post, the value of a si...



### [Attack of the week: RC4 is kind of broken in TLS](#)

Update: I've added a link to a page at Royal Holloway describing the new attack. Listen, if you're using RC4 as your primary c...



### [Can Apple read your iMessages?](#)

(source: Gizmodo ) About a year ago I wrote a short post urging Apple to publish the technical details of iMessage encryption. I...



algorithms we use (though [there are exceptions!](#)) However there are many complex protocols in cryptography, and sometimes [they are vulnerable](#).

2. **Go after the implementation.** Cryptography is almost always implemented in software -- and [software is a disaster](#). Hardware isn't that much better. Unfortunately active software exploits only work if you have a target in mind. If your goal is to mass surveillance, you need to build insecurity in from the start. That means working with vendors to add [backdoors](#).
3. **Access the human side.** Why steal someone's key if you can get them to give it to you?

Bruce Schneier, [who has seen the documents](#), says that 'math is good', but that 'code has been subverted'. He also says that the NSA is 'cheating'. Which, assuming we can trust these documents, is a huge sigh of relief. But it also means we're seeing a lot of (2) and (3) here.

### So which code should we be concerned about? Which hardware?

This is probably the most relevant question. If we're talking about commercial encryption code, the lion's share of it uses one of a small number of libraries. The most common of these are probably the [Microsoft CryptoAPI](#) (and Microsoft [SChannel](#)) along with the [OpenSSL library](#).

Of the above list, Microsoft is probably due for the most scrutiny. While Microsoft employs good (and paranoid!) people to vet their algorithms, their ecosystem is obviously deeply closed-source. You can view Microsoft's code (if you sign enough licensing agreements) but you'll never build it yourself. Moreover they have the market share. If any commercial vendor is weakening encryption systems, Microsoft is probably the most likely suspect.

And this is a problem because Microsoft IIS powers around 20% of the web servers on the Internet -- an absolute number that's less important than *which* servers those are. Moreover, even third-party encryption programs running on Windows often depend on CAPI components, including the random number generator. That makes these programs somewhat dependent on Microsoft's honesty.

Probably the second most likely candidate is OpenSSL. I know it seems like heresy to imply that OpenSSL -- an open source and widely-developed library -- might be vulnerable. But at the same time it powers an enormous amount of secure traffic on the Internet, thanks not only to the dominance of [Apache SSL](#), but also due to the fact that OpenSSL is *used everywhere*. You only have to glance at the [FIPS CMVP validation lists](#) to realize that many 'commercial' encryption products are just thin wrappers around OpenSSL.

Unfortunately while OpenSSL is open source, it periodically coughs up vulnerabilities. Part of this is due to the fact that it's a patchwork nightmare originally developed by a [novice who thought it would be a fun way to learn C](#). Part of it is because *crypto is unbelievably complicated*. Either way, there are very few people who really understand the whole codebase.

On the hardware side (and while we're throwing out baseless accusations) it would be awfully nice to take a second look at the [Intel Secure Key](#) (formerly Bull Mountain) integrated random number generators that most Intel processors will be getting shortly. Even if there's no problem, it's going to be an awfully hard sell pushing these internationally after today's news.

### Which standards?

From my point of view this is probably the most interesting and worrying part of today's leak. Software is almost always broken, but standards -- in theory -- get read by everyone. It should be extremely difficult to weaken a standard without someone noticing. And yet the Guardian and NYT stories are extremely specific that the NSA has been weakening standards.

The Guardian specifically calls out the [National Institute of Standards and Technology](#) (NIST) for a standard they published in 2006. Cryptographers have always had complicated feelings about NIST, and that's mostly because NIST has a complicated relationship with the NSA.

Here's the problem: the NSA ostensibly has both a *defensive* and an *offensive* mission. The defensive mission is pretty simple: it's to make sure US information systems don't get *pwned*. A substantial portion of that mission is accomplished through fruitful collaboration with NIST, which helps to promote data security standards such as the [Federal Information Processing Standards](#) (FIPS) and [NIST Special Publications](#).

I said cryptographers have complicated feelings about NIST, and that's because we all know that the NSA has the power to use NIST for good as well as evil. Up until today there's been no real evidence of malice, despite some [occasional glitches](#) -- and [compelling evidence that at least one NIST cryptographic standard could have contained a backdoor](#). But now maybe we'll have to re-evaluate that relationship. As utterly crazy as it may seem.



### Attack of the week: Cross-VM side-channel attacks

It's been a busy week for crypto flaws. So busy in fact, that I'm totally overwhelmed and paralyzed trying to write about them. It...

### Surviving a bad RNG

A couple of weeks ago I wrote a long post about random number generation, which I find to be one of the most fascinating subjects in crypto...



### Let's talk about ZRTP

Source: Zooko. I just checked the date on my last post and it seems that I haven't blogged in nearly a month. Believe me, this is...



### A bad couple of years for the cryptographic token industry

SafeNet eToken PRO Anywhere There was a time just a few weeks ago when it seemed like the CRYPTO 2012 accepted papers list might not be...

### Is the cryptopocalypse nigh?

I've been traveling a bit over the past couple of weeks, so I haven't had much of a chance to keep up on blogging. One consequence i...

### My Blog List

**Schneier on Security**  
The NSA Is Breaking Most Encryption on the Internet  
15 hours ago

**Bristol Cryptography Blog**  
Leakage-Resilient Symmetric Encryption via Re-Keying  
22 hours ago

**ellipticnews**  
ECC 2013 Conference  
2 days ago

**Shtetl-Optimized**  
Firewalls  
1 week ago

**The MPC Lounge**  
Workshop on Applied Multi-Party Computation (February 20-21, 2014, MSR Redmond)  
5 weeks ago

**root labs rdist**  
Keeping skills current in a changing world  
3 months ago

### Cryptanalysis



Unfortunately, we're highly dependent on NIST standards, ranging from [pseudo-random number generators](#) to [hash functions](#) and [ciphers](#) to the [specific elliptic curves](#) we use in SSL/TLS. While the possibility of a backdoor in any of these components does seem remote, it's going to be an absolute nightmare to rule it out.

### Which people?

Probably the biggest concern in all this is the evidence of collaboration between the NSA and unspecified 'telecom providers'. We already know that the major US (and international) telecom carriers [routinely assist the NSA in collecting data](#) from fiber-optic cables. But all this data is no good if it's encrypted.

While software compromises and weak standards can help the NSA deal with some of this, by far the easiest way to access encrypted data is to simply ask for -- or steal -- the keys. This goes for something as simple as [cellular encryption](#) (protected by a single key database at each carrier) all the way to SSL/TLS which is (most commonly) protected with a few relatively short RSA keys.

The good *and bad* thing is that as the nation hosting the largest number of popular digital online services (like Google, Facebook and Yahoo) many of those critical keys are located right here on US soil. Simultaneously, the people communicating with those services -- *i.e.*, the 'targets' -- may be foreigners. Or they may be US citizens. Or you may not know *who they are* until you scoop up and decrypt all of their traffic and run it for keywords.

Which means there's a circumstantial case that the NSA and GCHQ are either directly accessing [Certificate Authority](#) keys\* or else actively stealing keys from US providers, possibly (or probably) without executives' knowledge. This only requires a small number of people with physical or electronic access to servers, so it's quite feasible.\*\* The one reason I would have ruled it out a few days ago is because it seems so obviously immoral if not illegal, and moreover a huge threat to the checks and balances that the NSA allegedly has to satisfy in order to access specific users' data via programs such as [PRISM](#).

To me, the existence of this program is probably the least unexpected piece of all the news today. Somehow it's also the most upsetting.

### So what does it all mean?

I honestly wish I knew. Part of me worries that the whole security industry will talk about this for a few days, then we'll all go back to our normal lives without giving it a second thought. I hope we don't, though. Right now there are too many unanswered questions to just let things lie.

The most likely short-term effect is that there's going to be a lot less trust in the security industry. And a whole lot less trust for the US and its software exports. Maybe this is a good thing. We've been saying for years that you can't trust closed code and unsupported standards.

Even better, these revelations may also help to spur a whole burst of new research and re-designs of cryptographic software. We've also been saying that even *open* code like OpenSSL needs more expert eyes. Unfortunately there's been little interest in this, since the clever researchers in our field view these problems as 'solved' and thus somewhat uninteresting.

What we learned today is that they're solved all right. Just not the way we thought.

### Notes:

\* I had omitted the Certificate Authority route from the original post due to an oversight -- thanks to Kenny Patterson for pointing this out -- but I still think this is a less viable attack for *passive* eavesdropping (that does not involve actively running a [man in the middle attack](#)). And it seems that much of the interesting eavesdropping here is passive.

\*\* The major exception here is Google, which deploys Perfect Forward Secrecy for many of its connections, so key theft would not work here. To deal with this the NSA would have to subvert the software or break the encryption in some other way.

Posted by [Matthew Green](#) at 11:27 PM

+39 Recommend this on Google

8 comments:

SSL/TLS broken again – A weakness in the RC4 stream cipher  
5 months ago

Chargen  
Flint 1.1.0 Available  
3 years ago

### Subscribe

 Posts  
 Comments

### Followers

Join this site  
with Google  
Friend  
Connect

### Members (77)



Already a member? [Sign in](#)

### Blog Archive

- ▼ 2013 (16)
  - ▼ September (1)
    - [On the NSA](#)
  - ▶ August (1)
  - ▶ July (1)
  - ▶ June (2)
  - ▶ May (2)
  - ▶ April (2)
  - ▶ March (2)
  - ▶ February (3)
  - ▶ January (2)
- ▶ 2012 (48)
- ▶ 2011 (39)



**Anonymous** September 6, 2013 at 12:10 AM

Matthew,

Thanks for this (as ever) excellent overview of the issues.

Another "route to private keys" that is plausible, and supported by one of the diagrams in the Guardian article, is asking CAs to provide them for key pairs that they have generated on behalf of users/website.

Regards

Kenny Paterson

[Reply](#)

[Replies](#)



**Josef** September 6, 2013 at 1:24 AM

That's why you should ALWAYS create the key yourself and only send a CSR to the CA!



**Alaric Snell-Pym** September 6, 2013 at 3:18 AM

I presume it would be practical for the NSA to demand that CAs sign a CSR provided by the NSA that has, for example, mail.google.com in the CN field (or to demand the CA's private keys so they can just sign them themselves). Using such a cert for a man in the middle would run the risk of being noticed by highly savvy users who check the certificate fingerprints they receive against ones obtained through other channels, but I'm not sure if anybody does that...



**Matthew Green** September 6, 2013 at 3:21 AM

Yes, absolutely. I should have mentioned this in the first place. However the limitation of CA impersonation is that it only works for active (MITM) attacks. The interesting question here seems to be how the NSA/GCHQ obtain plaintext from fiber taps, which seem more difficult to actively attack at any kind of. But I've updated the post to mention this.

---

[Reply](#)



**Anonymous** September 6, 2013 at 12:33 AM

Is there possible relationship between "introducing weaknesses into security standards" mentioned in Guardian and this W3C Web Cryptography API problems?

You wrote about that WC-API before: <http://blog.cryptographyengineering.com/2012/12/the-anatomy-of-bad-idea.html>

[Reply](#)



**Anonymous** September 6, 2013 at 3:11 AM

Can you please elaborate on this: "Google [...] deploys Perfect Forward Secrecy for many of its connections"? Thanks!

[Reply](#)

[Replies](#)



**Anonymous** September 6, 2013 at 3:20 AM

<http://googleonlinesecurity.blogspot.com/2011/11/protecting-data-for-long-term-with.html>

---

[Reply](#)



**Pawel Krawczyk** September 6, 2013 at 3:38 AM

In 90's NSA made the mistake of trying to backdoor things openly - through mandating that in law. That obviously didn't work, and it couldn't work in country like pre-WTC USA or UK. So they switched back to good old HUMINT, that was seemingly not very much respected by the techies, and it seems to work much better. And that's actually something that you would expect an electronic intelligence agency to do, because that's what they are paid for. I'm pretty sure everyone - Russia, France, China - do it exactly the same way, with the difference that in countries like Russia or China it's much easier, as the security industry is saturated with people in uniforms. Plus citizen control is much weaker, and there's virtually no whistleblowing.

[Reply](#)

**Comment as:**

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)