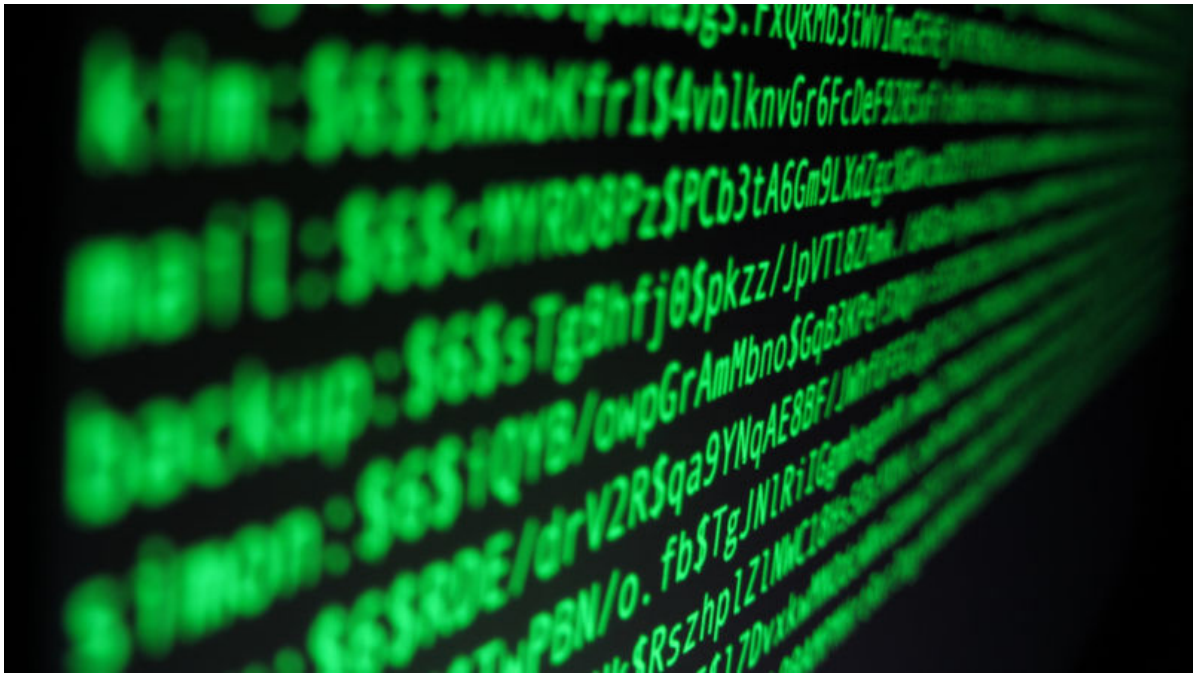
Op-ed: I’m throwing in the towel on PGP, and I work in security

Filippo Valsorda - Dec 10, 2016 2:00 pm UTC

opsec is still hard —

“If you need to securely contact me... DM me asking for my Signal number.”



Filippo Valsorda is an engineer on the Cloudflare Cryptography team, where he's deploying and helping design TLS 1.3, the next revision of the protocol implementing HTTPS. He also created a Heartbleed testing site in 2014. This post originally appeared on his blog and is re-printed with his permission.

After years of wrestling with GnuPG with varying levels of enthusiasm, I came to the conclusion that it's just not worth it, and I'm giving up—at least on the concept of long-term PGP keys. This editorial is not about the `gpg` tool itself, or about tools at all. Many others have already written about that. It's about the long-term PGP key model—be it secured by Web of Trust, fingerprints or Trust on First Use—and how it failed me.

Trust me when I say that I tried. I went through all the setups. I used Enigmail. I had offline master keys on a dedicated Raspberry Pi with short-lived subkeys. I wrote custom tools to make handwritten paper backups of offline keys (which I'll publish sooner or later). I had YubiKeys. Multiple. I spent days designing my public PGP policy.

I traveled two hours by train to meet the closest Biglumber user in Italy to get my first signature in the strong set. I have a signature from the most connected key in the set. I went to key-signing parties in multiple continents. I organized a couple.

I have the arrogance of saying that I understand PGP. In 2013 I was dissecting the packet format to brute force short IDs. I devised complex silly systems to make device subkeys tie to both my personal and company master keys. I filed usability and security issues in GnuPG and its various distributions.

All in all, I should be the perfect user for PGP: competent, enthusiast, embedded in a similar community. *But it just didn't work.*

First, there's the adoption issue others talked about extensively. I get, at most, two encrypted e-mails a year.

Then, there's the UX problem: easy crippling mistakes; messy keyserver listings from years ago; "I can't read this e-mail on my phone" or "on the laptop;" "I left the keys I never use on the other machine."

But the real issues, I realized, are more subtle. I never felt confident in the security of my long-term keys. The more time passed, the more I would feel uneasy about any specific key. Yubikeys would get exposed to hotel rooms. Offline keys would sit in a far away drawer or safe. Vulnerabilities would be announced. USB devices would get plugged in.

A long-term key is as secure as the minimum common denominator of your security practices over its lifetime. *It's the weak link.*

Worse, long-term key patterns, like collecting signatures and printing fingerprints on business cards, discourage practices that would otherwise be obvious hygiene: rotating keys often, having different keys for different devices, compartmentalization. Such practices actually encourage expanding the attack surface by making backups of the key.

We talk about Pets vs. Cattle in infrastructure; those concepts would apply just as well to keys! If I suspect I'm compromised, I want to be able to toss the laptop and rebootstrap with minimum overhead. The worst outcome possible for a scheme is making the user stick with a key that has a suspicion of compromise, because the cost of rotating would be too high.

And all this for what gain?

"Well, of course, long-term trust."

Yeah, about that. I never, ever, ever successfully used the WoT to validate a public key. And remember, I have a well-linked key. I haven't done a formal study, but I'm almost positive that everyone that used PGP to contact me has, or would have done (if asked), one of the following:

- pulled the best-looking key from a keyserver, most likely not even over TLS
- used a different key if replied with "this is my new key"
- re-sent the e-mail unencrypted if provided an excuse like "I'm traveling"

Travel in particular is hostile to long-term keys, making this kind of fresh start impractical.

Moreover, I'm not even sure there's an attacker that long-term keys make sense against. Your average adversary probably can't MitM Twitter DMs (which means you can use them to exchange fingerprints opportunistically, while still protecting your privacy). The Mossad will do Mossad things to your machine, whatever key you use.

Finally, these days I think I care much more about forward secrecy, deniability, and ephemerality than I do about ironclad trust. Are you sure you can protect that long-term key forever? Because when an attacker decides to target you and succeeds, they won't have access just from that point forward; they'll have access to all your past communications, too. And that's ever more relevant.

Moving forward

I'm not dropping to plaintext. Quite the opposite. But I won't be maintaining any public long-term key.

Mostly I'll use Signal or WhatsApp, which offer vastly better endpoint security on iOS, ephemerality, and smoother key rotation.

If you need to securely contact me, your best bet is to DM me asking for my Signal number. If needed we can decide an appropriate way to compare fingerprints.

If we meet in person and need to set up a secure channel, we will just exchange a secret passphrase to use with what's most appropriate: OTR, Pond, Ricochet.

If it turns out we really need PGP, we will set up some ad-hoc keys, more *à la* Operational PGP. Same for any signed releases or canaries I might maintain in the future.

To exchange files, we will negotiate Magic Wormhole, OnionShare, or ad-hoc PGP keys over the secure channel we already have. The point is not to avoid the `gpg` tool, but the PGP key management model.

If you really need to cold-contact me, I might maintain a Keybase key, but no promises. I like rooting trust in your social profiles better since it makes key rotation much more natural and is probably how most people know me anyway.

I'm also not dropping YubiKeys. I'm very happy about my new YubiKey 4 with touch-to-operate, which I use for SSH keys, password storage, and machine bootstrap. But these things are one hundred percent under my control.

About my old keys and transitioning

I broke the offline seal of all my keys. I don't have reason to believe they are compromised, but you should stop using them now.

Below are detached signatures for the Markdown version of this document from all keys I could still find.

In the coming weeks I'll import all signatures I received, make all the signatures I promised, and then publish revocations to the keyservers. I'll rotate my Keybase key. Eventually, I'll destroy the private keys.

See you on Signal. (Or Twitter.)

Giving up on PGP.md
Giving up on PGP.md.B8CC58C51CAEA963.asc
Giving up on PGP.md.C5C92C16AB6572C2.asc
Giving up on PGP.md.54D93CBC8AA84B5A.asc
Giving up on PGP.md.EBF01804BCF05F6B.asc [*coming once I recover the passphrase from another country*]

Note: I expect the "Moving forward" section to evolve over time, as tools come and go. The signed `.md` file won't change, an unauthenticated `.diff` will appear below for verification convenience.