
CLOUDFLARE
post-
quantum
fancy
crypto-
graphy
Bas Westerbaan
Cloudflare Research

👏 First: thank you!

Post-quantum key agreement is used at a huge scale today.
(Signal, iMessage, webservers, browsers)

(credit: Bill Jacobus)

In 2010 practically all cryptography with any real-world deployment or standardisation are boring* primitives such as plain signatures, hashes, symmetric encryption, public-key encryption, DH.

* An aside: boring is a compliment.

Most standardised foundational cryptography is not boring enough yet.

(Sharp edges with AES-GCM. Cofactors. New security properties of signatures / KEMs. And so on …)

… and boring is hard enough!

Primitive	Examples	Migration to PQC
Symmetric ciphers	AES, SHA3	⭐ Already PQ.
Key agreement / PKE	X25519, ML-KEM	🙂 Doable. Actual progress.
Signatures	Ed25519, ML-DSA	😣 Painful and being delayed.
Fancy cryptography	Blind signatures, PAKEs, ZKPs, …	😵‍💫 Mixed bag. Mostly terrible.

Only in the past few years have we seen the slow real-world adoption and even slower standardisation of what Google's Sophie Schmieg calls fancy* cryptographic primitives.

(* Also a compliment.)

To name a few

Anonymous credentials and zero-knowledge proofs in Signal's private group system. Oblivious PAKEs in WhatsApp's encrypted backups, and regular ones in Magic Wormhole. Unlinkable tokens in Apple Private Relay (blind signatures), Privacy Pass (OPRF), and Dutch CoronaCheck app (Idemix). Attribute-Based Encryption in Cloudflare's GeoKDL. Private set intersection with blinding for password protection in Chrome.

(We've started to crowdsource a list here.)

Very few are standardised

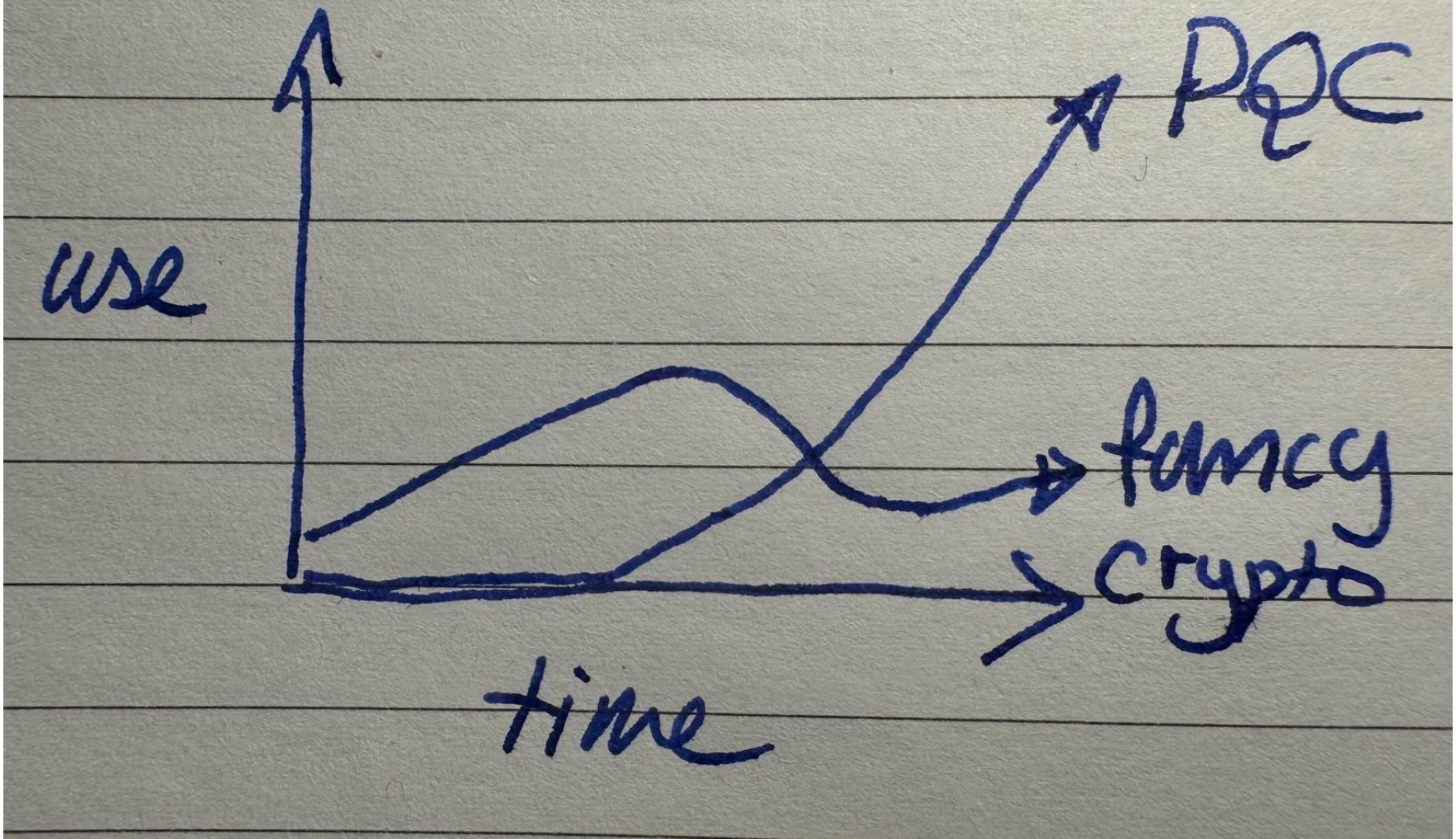
- RFC 9474 RSA Blind signatures
- RFC 9382 SPAKE2
- RFC 9497 OPRF

In progress of standardisation

- OPAQUE
- Privacy pass
- Curves with pairings
- Threshold cryptography

My worry

(Please forgive me the placeholder illustration on the next slide)


use
PQC
fancy
crypto
time

Before thinking about post-quantum fancy cryptography,
I think it's helpful to reflect…

Why isn’t fancy cryptography used more often today?

Performance? Certainly a big concern.

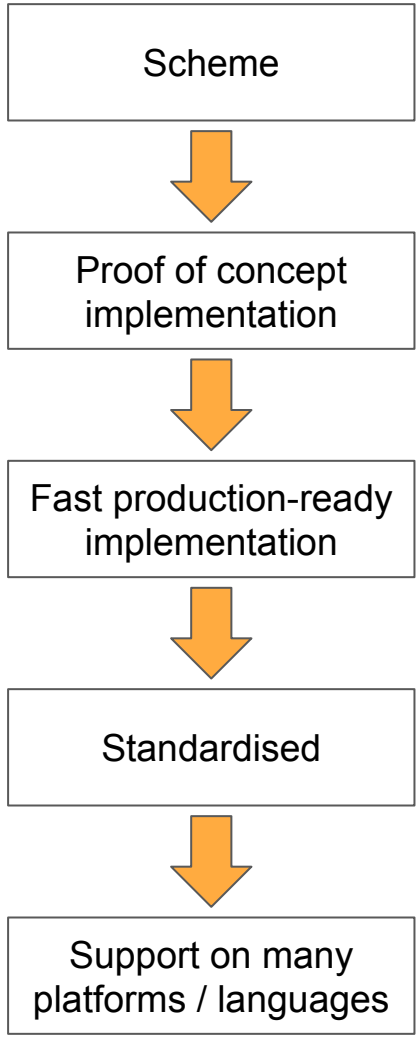
Complexity of implementation, specification, security properties and trade-offs.

Could it be useful to my problem?

Should it be useful to my problem?

Is it useful to my problem? Brave single-party deployment.

Multi-party deployment



Lattice based ZKP

Groth16

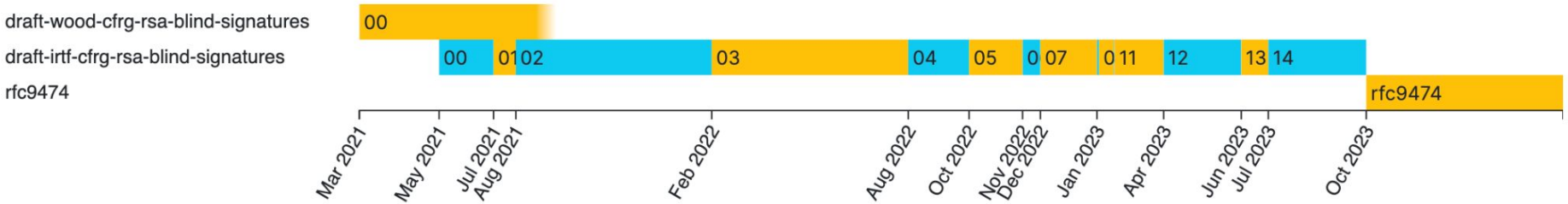
RSA blind signatures

RSA Blind Signatures

RFC 9474

Status | Email expansions | History

This RFC was published on the Internet Research Task Force (IRTF) stream. This RFC is **not endorsed by the IETF** and has **no formal standing** in the IETF standards process.

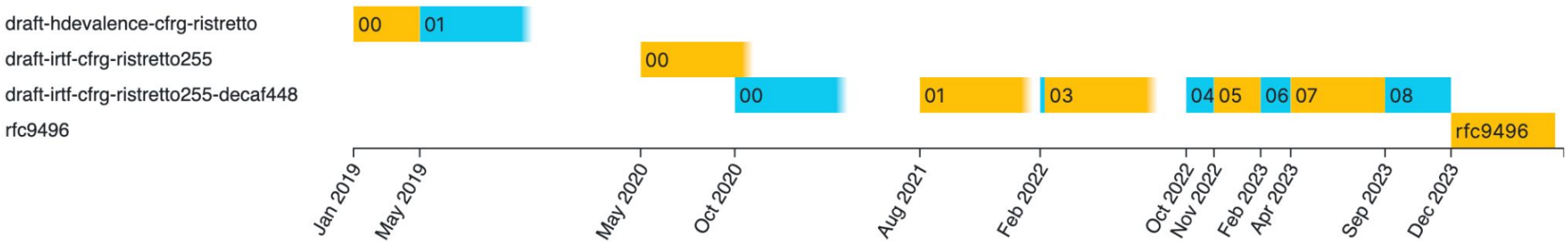


The ristretto255 and decaf448 Groups

RFC 9496

Status | Email expansions | History

This RFC was published on the Internet Research Task Force (IRTF) stream. This RFC is **not endorsed by the IETF** and has **no formal standing** in the IETF standards process.

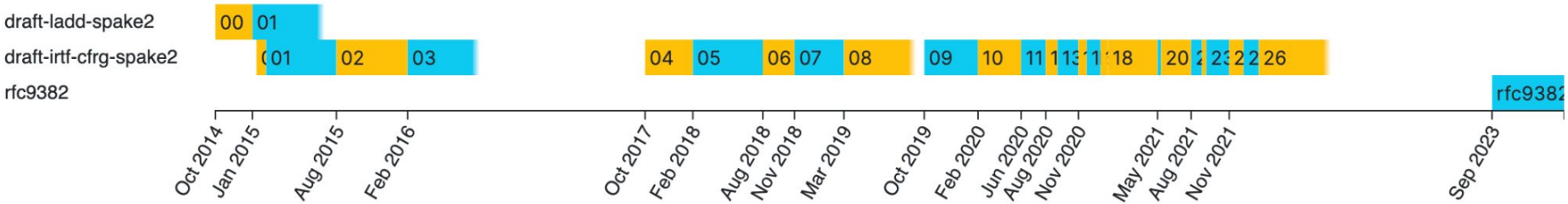


SPAKE2, a Password-Authenticated Key Exchange

RFC 9382

Status | Email expansions | History

This RFC was published on the Internet Research Task Force (IRTF) stream. This RFC is **not endorsed by the IETF** and has **no formal standing** in the IETF standards process.



Boring cryptography is hard to avoid: it solves relatively easy-to-explain problems well that mostly cannot be solved in any other way.

We know fancy cryptography can solve problems that cannot be solved in any other way, but they're often subtle, and…

Fancy cryptography in practice competes with solutions using trusted third parties, trusted execution environments, policy measures, etc — or frankly, not solving the issue at all.

On a more positive note…

Fancy cryptography and their use cases have a discoverability problem.

I am sure more fancy cryptography would be used today, if industry knew better what's possible.

Industry tells academia what they care about by what systems they've deployed. What's left on the table, is what they would like to solve.

Academia communicates with industry what's possible by telling them what cryptographic primitives they've designed. But these are rarely tailored to the them unknown use cases.

Crypto primitives as interface

Pro

Modular analysis / design.

Clear target for effort.

Deduplicate work.

Easier to communicate / discover.

Con

Full solution not as efficient as it could be: much worse than with boring crypto.

Use cases don’t map cleanly to primitives.

Distracts effort from the application.

Take aways?

Complexity often is a bigger bottleneck than performance.

Industry needs to get better at standardising fancy crypto.

Use cases are complex and benefit from tailoring: helpful to look at the full use case instead of the primitives.

Your thoughts?