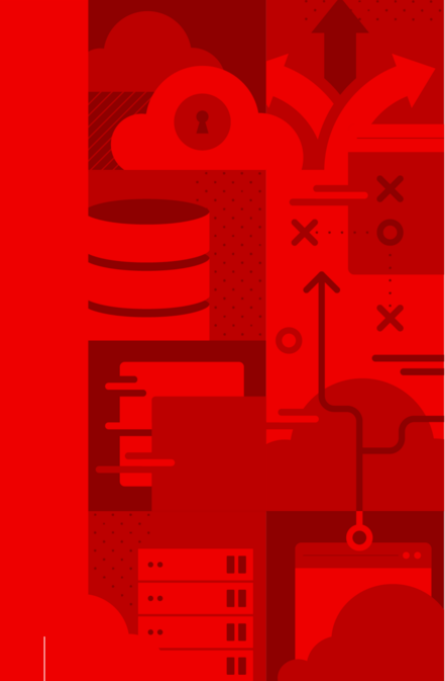


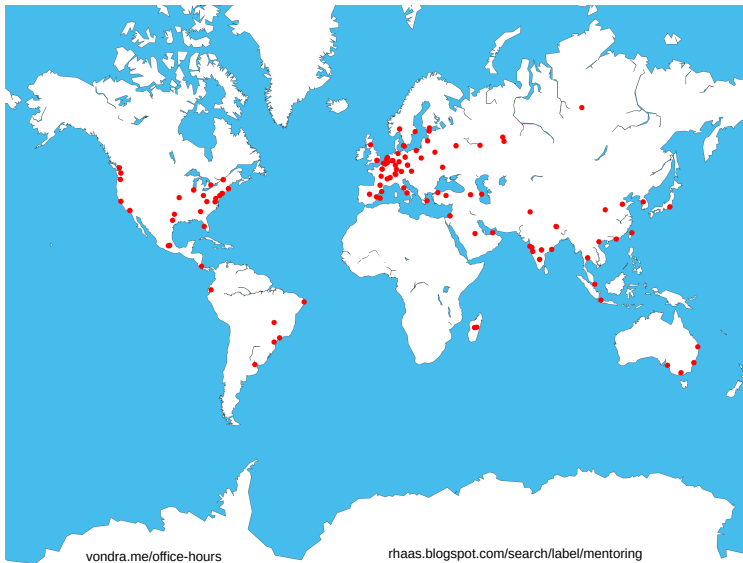


Post Quantum Cryptography in PostgreSQL

Why, how and when?

Dmitrii Dolgov, Filip Janus

20-10-2026





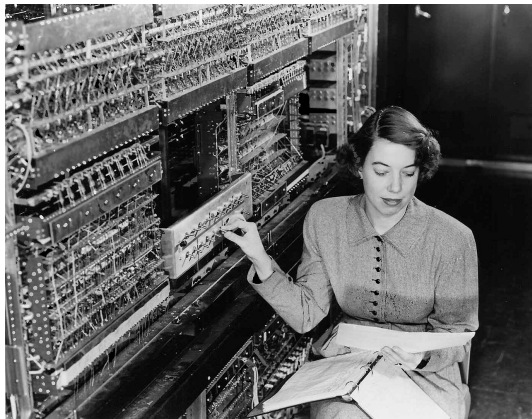
Intro

PostgreSQL almost supports PQC



Quantum Computing

Work of J. Mauchly and J. P. Eckert, a.k.a **Von Neumann architecture.**



Jean F. Hall, AVIDAC, 1953 (Argonne National Laboratory)

Not the only one...

Analog computing

Optical/Photonic computing

DNA computing

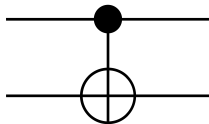
Quantum computing

Kogge, Peter M. The Zen of Exotic Computing. Society for Industrial and Applied Mathematics, 2022.

Quantum computing

$$|\psi\rangle, |\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Spin, Ion trap, Photon, Topological



Mohamed M. Sabry Aly, Anupam Chattopadhyay. Emerging Computing: From Devices to Systems. Springer Singapore, 2023.

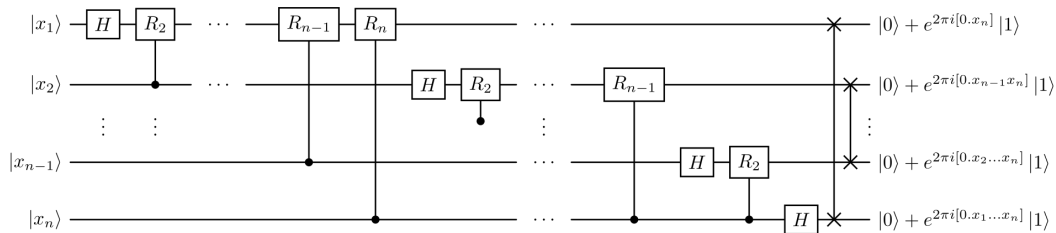
Factorization

The security of RSA is related to the difficulty of factoring the product of two large prime numbers.

The largest publicly known (2026) factored RSA number had 862 bits. Currently recommended number of bits is 2048.

Cormen, Thomas H., et al. Introduction to algorithms. Vol. 2. Cambridge: MIT press, 1990.

Shor's algorithm



Quantum Fourier transform (QFT), the quantum part of Shor's algorithm, @Sjoerd Terlouw.

Current state of the art

Ref.	Qubits (10^6)	Runtime (d)
Van Meter et al. 2009	6500	410
Jones et al. 2010	620	10
Fowler et al. 2012	1000	1.1
O'Gorman et al. 2017	230	3.7
Gheorghiu et al. 2019	170	1
Gidney et al. 2019	20	0.31
Gidney et al. 2025	1	5
Webster et al. 2026	0.1	30

Current state of the art

	2026	2028	2033
IBM	120	1080	2000
IONQ	12	1600	80000

<https://www.ibm.com/roadmaps/quantum/>
<https://www.ionq.com/roadmap>

Replication of Quantum Factorisation Records with an 8-bit Home Computer, an Abacus, and a Dog

Peter Gutmann, University of Auckland

`pgut001@cs.auckland.ac.nz`

Stephan Neuhaus, Zürcher Hochschule für

Angewandte Wissenschaften

`neut@zhaw.ch`

March 2025, last update November 2025

1 Abstract

This paper presents implementations that match and, where possible, exceed current quantum factorisation records using a VIC-20 8-bit home computer from 1981, an abacus, and a dog. We hope that this work will inspire future efforts to match any further quantum factorisation records, should they arise.



PQC

NIST IR 8547

Name	Deprecated	Disallowed
ECDSA (112)	2030	2035
ECDSA (≥ 128)		2035
RSA (112)	2030	2035
RSA (≥ 128)		2035
ECDH (112)	2030	2035
ECDH (≥ 128)		2035

Moody, Dustin, et al. Transition to post-quantum cryptography standards. NIST Internal or Interagency Report (NISTIR) 8547, 2024.

"Looking forward, I agree with the initial public draft of the NIST internal report on the transition to post-quantum cryptography standards: vulnerable systems should be **deprecated after 2030** and **disallowed after 2035**. Not because I expect sufficiently large quantum computers to exist by 2030, but because I prefer security to not be contingent on progress being slow."

Gidney, Craig. "How to factor 2048 bit RSA integers with less than a million noisy qubits." arXiv preprint arXiv:2505.15917 (2025).

EU

2026: First steps, transition roadmap

2030: High-risk use cases are completed

2035: Medium-risk use cases are completed

EU PQC Workstream. "A coordinated implementation roadmap for the transition to Post-Quantum Cryptography",
<https://digital-strategy.ec.europa.eu/en/node/13885/> (2025).

FIPS 203

ECDH -> ML-KEM, shared secret key

FIPS 204

ECDSA -> ML-DSA, digital signature

”In fact, Signal and Apple already put in production various Signal-based variants with certain levels of **hybrid security**: PQXDH (only on the initial handshake), and PQ3 (on the entire protocol), by adding a PQ-ratchet to the DR protocol.”

Dodis, Yevgeniy, et al. "Triple ratchet: A bandwidth efficient hybrid-secure signal protocol." Annual International Conference on the Theory and Applications of Cryptographic Techniques. Cham: Springer Nature Switzerland, 2025.



OpenSSL

Changes between 3.4 and 3.5.0 [8 Apr 2025]:

- * ML-KEM as specified in FIPS 203.
- * Add ML-DSA as specified in FIPS 204.

PostgreSQL 18

```
# Minimum required OpenSSL version is 1.1.1
AC_DEFINE(OPENSSL_API_COMPAT, [0x10101000L],
    [Define to the OpenSSL API version in use.
     This avoids deprecation warnings from
     newer OpenSSL versions.] )
```

PostgreSQL 18

`ssl_ecdh_curve -> ssl_groups`

`SSL_CTX_set_tmp_ecdh -> SSL_CTX_set1_groups_list`

`# postgresql.conf`

`ssl_groups="X25519MLKEM768:X25519:prime256v1"`

https://www.postgresql.org/message-id/flat/tencent_27D1768E49DE4D7186BE2F7112930C332C06%40qq.com

How to verify?

```
openssl s_client  
  -starttls postgres  
  -connect postgres.host:5432  
  -groups X25519MLKEM768 -brief
```

```
CONNECTION ESTABLISHED  
Protocol version: TLSv1.3  
Negotiated TLS1.3 group: X25519MLKEM768
```

How to verify?

Transport Layer Security

TLSv1 Record Layer: Handshake Protocol: Client Hello

Handshake Protocol: Client Hello

Handshake Type: Client Hello (1)

[...]

How to verify?

Extension: supported_groups (len=28)

Type: supported_groups (10)

Length: 28

Supported Groups List Length: 26

Supported Groups (13 groups)

Supported Group: X25519MLKEM768 (0x11ec)

Supported Group: SecP256r1MLKEM768 (0x11eb)

[...]

How to verify?

Extension: key_share (len=**1258**) X25519MLKEM768, x25519
Type: key_share (**51**)
Length: **1258**
Key Share extension
Client Key Share Length: **1256**
Key Share Entry: Group: X25519MLKEM768
Group: X25519MLKEM768 (**4588**)
Key Exchange Length: **1216**
[...]

How to verify?

Transport Layer Security

TLSv1.2 Record Layer: Handshake Protocol: Server Hello

Handshake Protocol: Server Hello

Handshake Type: Server Hello (2)

Extension: key_share (len=1124) X25519MLKEM768

Type: key_share (51)

Length: 1124

Key Share extension

Key Share Entry: Group: X25519MLKEM768

Group: X25519MLKEM768 (4588)

Key Exchange Length: 1120

[...]

Multiple certificates?

```
ssl_cert_files = 'server-rsa.crt, server-ecdsa.crt'  
ssl_key_files  = 'server-rsa.key, server-ecdsa.key'
```

<https://www.postgresql.org/message-id/flat/0cabf744-6d58-4bc4-817a-413c804cf61d%40redhat.com>



Channel Binding

ML-DSA + Channel binding

ERROR: could not find digest for NID UNDEF

https://www.postgresql.org/message-id/flat/CAFjYY%2BJCCQeh03nzVG6Rs9MUgU_kOvhMbNaaS6kn_c4CcAZkTg%40mail.gmail.com

RFC 5056

Channel binding: the process of establishing that no man-in-the-middle exists between two end-points that have been authenticated at one network layer but are using a secure channel at a lower network layer.

RFC 5929: tls-server-end-point

single hash function => digest from the certificate

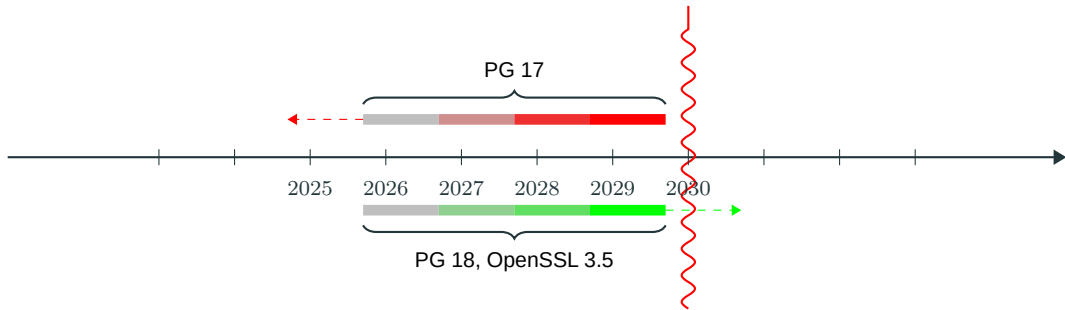
no unique hash function (e.g. ML-DSA) => ?

Errata-ID: 9041

"If no unique suitable hash function could be identified from the certificate's signatureAlgorithm, then use the identity function."



Summary



Questions?

✉ @erthalion@fosstodon.org

✉ ddolgov at redhat dot com

✉ fjanus at redhat dot com