Code-based post-quantum cryptography

D. J. Bernstein
University of Illinois at Chicago

“Once the enormous energy boost that quantum computers are expected to provide hits the street, most encryption security standards—and any other standard based on computational difficulty—will fall, experts believe.”

(Magiq’s web site, 2008; the “experts” aren’t named)

Is cryptography dead?

Imagine:
15 years from now
someone announces
successful construction
of a large quantum computer.

New York Times headline:
"INTERNET CRYPTOGRAPHY
KILLED BY PHYSICISTS."

Users panic.

What happens to cryptography?

RSA: Dead.

RSA: Dead.

DSA: Dead.

ECDSA: Dead.

RSA: Dead.

DSA: Dead.

ECDSA: Dead.

ECC in general: Dead.

HECC in general: Dead.

RSA: Dead.

DSA: Dead.

ECDSA: Dead.

ECC in general: Dead.

HECC in general: Dead.

Buchmann–Williams: Dead.

Class groups in general: Dead.

RSA: Dead.

DSA: Dead.

ECDSA: Dead.

ECC in general: Dead.

HECC in general: Dead.

Buchmann–Williams: Dead.

Class groups in general: Dead.

“They’re all dead, Dave.”

RSA: Dead.

DSA: Dead.

ECDSA: Dead.

ECC in general: Dead.

HECC in general: Dead.

Buchmann–Williams: Dead.

Class groups in general: Dead.

"They're all dead, Dave."

But we have other types of cryptographic systems!

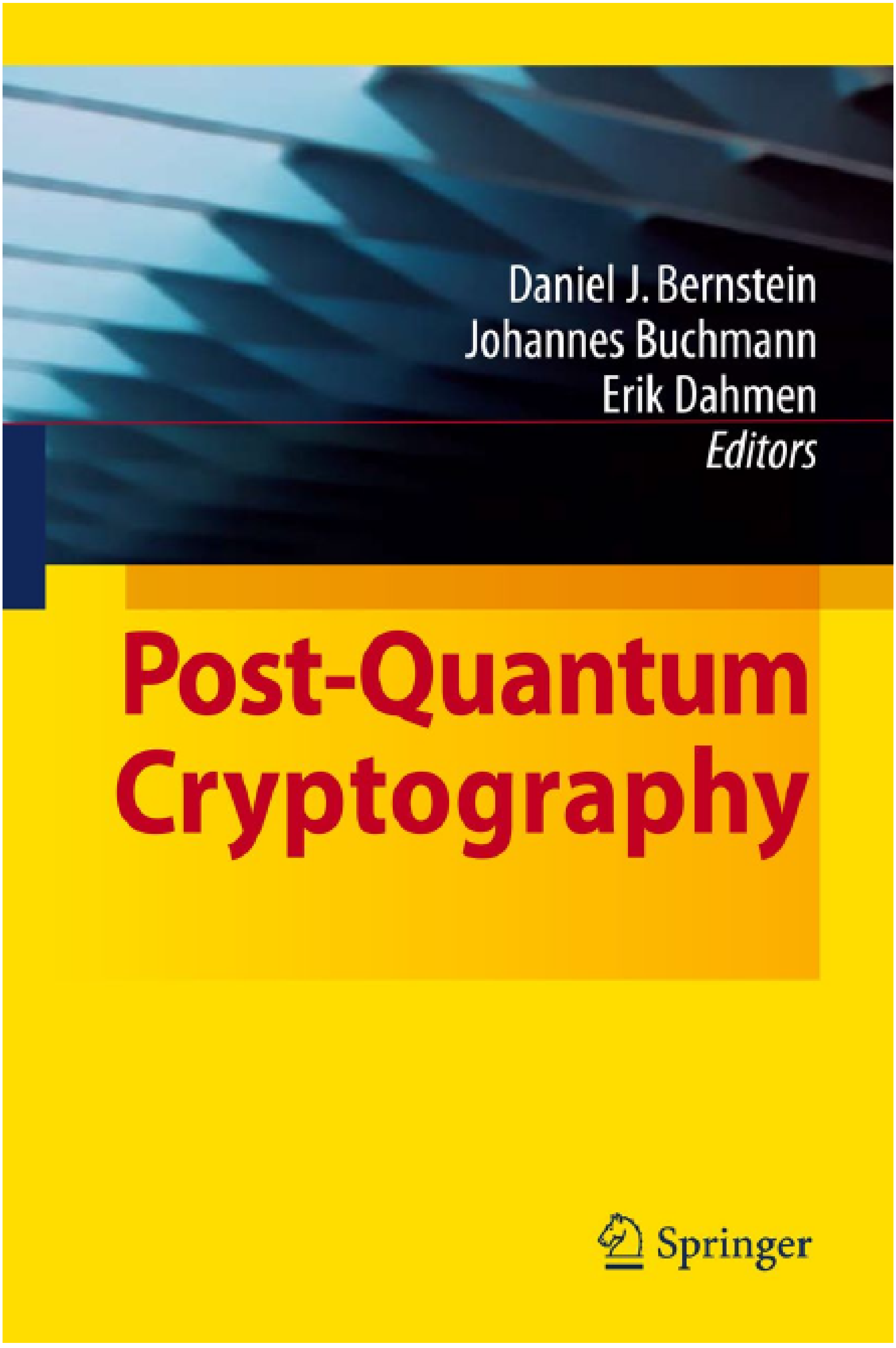
Hash-based cryptography.
Example: 1979 Merkle hash-tree public-key signature system.

Code-based cryptography.
Example: 1978 McEliece hidden-Goppa-code public-key encryption system.

Lattice-based cryptography.
Example: 1998 “NTRU.”

Multivariate-quadratic-equations cryptography.
Example:
1996 Patarin “$\text{HFE}^{\text{v}-}$”
public-key signature system.

Secret-key cryptography.
Example: 1998 Daemen–Rijmen “Rijndael” cipher, aka “AES.”



Daniel J. Bernstein
Johannes Buchmann
Erik Dahmen
Editors

Post-Quantum Cryptography

Springer

Bernstein: "Introduction to post-quantum cryptography."

Hallgren, Vollmer: "Quantum computing."

Buchmann, Dahmen, Szydlo: "Hash-based digital signature schemes."

Overbeck, Sendrier: "Code-based cryptography."

Micciancio, Regev: "Lattice-based cryptography."

Ding, Yang: "Multivariate public key cryptography."

The McEliece cryptosystem

Receiver's public key: "random" 500×1024 matrix K over $\mathbf{F}_2$. Specifies linear $\mathbf{F}_2^{1024} \to \mathbf{F}_2^{500}$.

Messages suitable for encryption: 1024-bit strings of weight 50; i.e., $\{m \in \mathbf{F}_2^{1024} :$

$$\#\{i : m_i = 1\} = 50\}.$$

Encryption of m is $Km \in \mathbf{F}_2^{500}$.

Can use m as secret AES key to encrypt much more data.

Attacker, by linear algebra, can easily work backwards from Km to *some* $v \in \mathbf{F}_2^{1024}$ such that $Kv = Km$.

i.e. Attacker finds *some* element $v \in m + \mathrm{Ker}K$. Note that $\#\mathrm{Ker}K \geq 2^{524}$.

Attacker wants to decode v: to find element of Ker$K$ at distance only 50 from v. Presumably unique, revealing m.

But decoding isn't easy!

Information-set decoding

Choose random size-500 subset $S \subseteq \{1, 2, 3, \ldots, 1024\}$.

For typical K: Good chance that $\mathbf{F}_2^S \hookrightarrow \mathbf{F}_2^{1024} \xrightarrow{K} \mathbf{F}_2^{500}$ is invertible.

Hope $m \in \mathbf{F}_2^S$; chance $\approx 2^{-53}$. Apply inverse map to Km, revealing m if $m \in \mathbf{F}_2^S$.

If $m \notin \mathbf{F}_2^S$, try again. $\approx 2^{80}$ operations overall.

Various improvements:
1988 Lee–Brickell;
1988 Leon;
1989 Stern;
1990 van Tilburg;
1994 Canteaut–Chabanne;
1998 Canteaut–Chabaud;
1998 Canteaut–Sendrier.
2^{68} Alpha cycles.

2008 Bernstein–Lange–Peters:
further improvements;
2^{58} Core 2 Quad cycles;
carried out successfully!

1988 Lee–Brickell idea:
Hope that $m + e \in \mathbf{F}_2^S$
for some weight-2 vector e.
Reuse one matrix inversion
for all choices of e.

1989 Stern idea:
Hope that $m + e + e' \in \mathbf{F}_2^S$
for low-weight vectors e, e'.
Search for collision between
function of e, function of e'.

2008 Bernstein–Lange–Peters:
more reuse, optimization, etc.

Modern McEliece

Easily rescue system by using a larger public key: “random” $(n/2) \times n$ matrix K over $\mathbf{F}_2$. e.g., 1800×3600.

Larger weight: $\approx n/(2 \lg n)$. e.g. $m \in \mathbf{F}_2^{3600}$ of weight 150.

All known attacks scale badly: roughly $2^{n/(2 \lg n)}$ operations. For much more precise analysis see 2009 Bernstein–Lange–Peters–van Tilborg.

Receiver secretly generates public key K with a hidden Goppa-code structure that allows fast decoding.

Namely: $K = SHP$ for secret $(n/2) \times (n/2)$ invertible matrix S, $(n/2) \times n$ Goppa matrix H, $n \times n$ permutation matrix P.

Detecting this structure seems even more difficult than attacking random K.

Goppa codes

Fix $q \in \{8, 16, 32, \ldots\}$;
$t \in \{2, 3, \ldots, \lfloor (q-1)/\lg q \rfloor\}$;
$n \in \{t \lg q + 1, t \lg q + 2, \ldots, q\}$.
e.g. $q = 1024$, $t = 50$, $n = 1024$.
or $q = 4096$, $t = 150$, $n = 3600$.

Receiver's matrix H is
the parity-check matrix
for the classical (genus-0)
irreducible length-n degree-t
binary Goppa code defined by
a monic degree-t irreducible
polynomial $g \in \mathbf{F}_q[x]$ and
distinct $a_1, a_2, \ldots, a_n \in \mathbf{F}_q$.

. . . which means: $H =$

$$\begin{pmatrix} \frac{1}{g(a_1)} & \cdots & \frac{1}{g(a_n)} \\ \frac{a_1}{g(a_1)} & \cdots & \frac{a_n}{g(a_n)} \\ \vdots & \ddots & \vdots \\ \frac{a_1^{t-1}}{g(a_1)} & \cdots & \frac{a_n^{t-1}}{g(a_n)} \end{pmatrix}.$$

View each element of $\mathbf{F}_q$ here as a column in $\mathbf{F}_2^{\lg q}$.

Then $H : \mathbf{F}_2^n \to \mathbf{F}_2^{t \lg q}$.

More useful view: Consider the map $m \mapsto \sum_i m_i/(x - a_i)$ from $\mathbf{F}_2^n$ to $\mathbf{F}_q[x]/g$.

H is the matrix for this map where $\mathbf{F}_2^n$ has standard basis and $\mathbf{F}_q[x]/g$ has basis $\lfloor g/x \rfloor$, $\lfloor g/x^2 \rfloor$, $\ldots$, $\lfloor g/x^t \rfloor$.

One-line proof: In $\mathbf{F}_q[x]$ have

$$\frac{g - g(a_i)}{x - a_i} = \sum_{j \ge 0} a_i^j \left\lfloor g/x^{j+1} \right\rfloor.$$

Decoding Goppa codes

1975 Patterson: Given Hm,
can quickly find m
if weight of m is $\leq t$.

Given ciphertext $Km = SHPm$:
receiver computes HPm
by applying secret S^{-1};
decodes H to obtain Pm
by Patterson's algorithm;
computes message m
by applying secret P^{-1}.

Patterson input is $r \in \mathbf{F}_q[x]/g$ having form $\sum_i m_i/(x - a_i)$ where $m \in \mathbf{F}_2^n$ has weight $\le t$. Output will be m.

If $r = 0$, output 0 and stop.

If $r \neq 0$:
Lift $\sqrt{r^{-1} - x}$ from $\mathbf{F}_q[x]/g$ to $s \in \mathbf{F}_q[x]$ of degree $< t$.

Consider lattice $L \subseteq \mathbf{F}_q[x]^2$ generated by $(s, 1)$ and $(g, 0)$.

Define length of (α, β) as norm of $\alpha^2 + x\beta^2$.

Find a minimum-length nonzero vector $(\alpha_0, \beta_0) \in L$.

Monic part of $\epsilon_0 = \alpha_0^2 + x\beta_0^2$
is exactly $\prod_{i:m_i=1}(x - a_i)$.
Factor ϵ_0 and print m.

Why this works:
Define $\epsilon = \prod_{i:m_i=1}(x - a_i)$.
Write ϵ as $\alpha^2 + x\beta^2$ in $\mathbf{F}_q[x]$.
Have $\epsilon'/\epsilon = r$ in $\mathbf{F}_q[x]/g$
so $\beta^2/(\alpha^2 + x\beta^2) = 1/(s^2 + x)$
so $s = \alpha/\beta$ in $\mathbf{F}_q[x]/g$;
i.e., $(\alpha, \beta) \in L$.
Volume of L forces
$(\alpha, \beta) \in (\alpha_0, \beta_0)\mathbf{F}_q[x]$
so $\epsilon = \text{square} \cdot \epsilon_0$;
ϵ is squarefree so square $\in \mathbf{F}_q$. $\blacksquare$

What if Patterson is used for m having weight $> t$?

Volume argument fails.
$(\alpha, \beta) \notin (\alpha_0, \beta_0)\mathbf{F}_q[x]$.

But can compute short basis $(\alpha_0, \beta_0), (\alpha_1, \beta_1)$ of L.

Then ϵ is a linear combination of $\epsilon_0 = \alpha_0^2 + x\beta_0^2$
and $\epsilon_1 = \alpha_1^2 + x\beta_1^2$.
Coefficients are small squares;
"small" depends on weight of m.

Divisors in residue classes

Want all divisors of n in $u + v\mathbf{Z}$, given positive integers u, v, n with $\gcd\{v, n\} = 1$.

Easy if $v \geq n^{1/2}$.

1984 Lenstra: polynomial-time algorithm for $v \geq n^{1/3}$.

1997 Konyagin–Pomerance: polynomial-time algorithm for $v \geq n^{3/10}$.

1998 Coppersmith–Howgrave-Graham–Nagaraj: polynomial-time algorithm for $v \geq n^{1/4+\epsilon}$.

2000 Boneh: can view same algorithm as a list-decoding algorithm for CRT codes.

Function-field analogue is famous 1999 Guruswami–Sudan algorithm for list decoding of Reed–Solomon codes.

Can build grand unified picture of “Coppersmith-type” algorithms and “Sudan-type” algorithms. See, e.g., my survey paper “Reducing lattice bases to find small-height values of univariate polynomials.”

2008 Bernstein:

Tweak parameters
in the same algorithm
to find all divisors of n that are
linear combinations of u, v
with small coprime coefficients.

2008 Bernstein:

Tweak parameters
in the same algorithm
to find all divisors of n that are
linear combinations of u, v
with small coprime coefficients.

Apply to the Goppa situation:
analogous algorithm finds all
divisors of $\prod_i (x - a_i)$ that are
linear combinations of ϵ_0, ϵ_1
with small coprime coefficients.

Compared to Patterson,
pushes allowable weight of m
up to $\approx t + t^2/n$.

New algorithm assumes that ϵ_1 is coprime to $\prod_i (x - a_i)$.

Easy to achieve by adding a small multiple of ϵ_0 to ϵ_1.

... unless $n = q$ and ϵ_1/ϵ_0 is a permutation function. Can this happen to Patterson?

I don't know any examples. Weil forces rather large degree: can show that the curve

$$\frac{\epsilon_0(x)\epsilon_1(y) - \epsilon_1(x)\epsilon_0(y)}{x - y} = 0$$

has no points over $\mathbf{F}_q$.

Many other current topics
in code-based cryptography.

e.g. 2009 Misoczki–Barreto:
Hide quasi-dyadic Goppa code
as quasi-dyadic public key.
Key length only $b^{1+o(1)}$.
Encryption time $b^{\lg 3+o(1)}$.
Decryption time $b^{\lg 3+o(1)}$.

2009 Bernstein: easy tweak
to Misoczki–Barreto algorithms,
reducing time to $b^{1+o(1)}$.