

A brief survey  
of 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 “HFE<sup>v-</sup>”

public-key signature system.

## **Secret-key cryptography.**

Example: 1998 Daemen–Rijmen

“Rijndael” cipher, aka “AES.”

Nobody has figured out a way to apply Shor's algorithm to any of these systems.

Grover's algorithm does have some applications, but cryptographers can easily compensate by scaling up somewhat.

“Maybe there’s a better attack!”

This was already a familiar risk before quantum computers.

This is why the community puts tremendous effort into cryptanalysis.

Results of cryptanalysis:

Some systems are killed.

Some systems need larger keys.

Some systems inspire confidence.

1978 RSA paper mentions Schroeppele's "linear sieve" for factorization. Need  $b^{2+o(1)}$ -bit RSA key for  $b$ -bit security.

1988 Pollard et al.: "Number-field sieve." Need  $b^{3+o(1)}$ -bit RSA key for  $b$ -bit security.

Many improvements since then, but still  $b^{3+o(1)}$  ... against classical computers.

1994 Shor: Need intolerably large RSA key for  $b$ -bit security against quantum computers.

1978 McEliece paper  
mentions a decoding algorithm.  
Need code dimension  
 $b^{1+o(1)}$  for  $b$ -bit security.

Many improvements since then,  
but still dimension  $b^{1+o(1)}$   
against classical computers.

Still dimension  $b^{1+o(1)}$   
against quantum computers.

If McEliece security is so good,  
why are we still using RSA?

Answer: McEliece key is huge.

Pre-quantum cryptography:

Cryptographers design systems to scramble and unscramble data.

RSA, McEliece, AES, many more.

Cryptanalysts break some systems using  $< 2^b$  classical operations.

Tools: NFS, LLL, F4, etc.

Unbroken systems:

RSA with  $b^{3+o(1)}$  bits,

McEliece with dimension  $b^{1+o(1)}$ ,

AES if  $b \leq 256$ , etc.

Algorithm designers and implementors find the fastest unbroken systems.

Post-quantum cryptography:

Cryptographers design systems to scramble and unscramble data.

RSA, McEliece, AES, many more.

Cryptanalysts break some systems using  $< 2^b$  quantum operations.

Tools: NFS, LLL, F4, etc.

*plus* Shor, Grover, etc.

Unbroken systems:

McEliece with dimension  $b^{1+o(1)}$ ,

AES if  $b \leq 128$ , etc.

Algorithm designers and implementors find the fastest unbroken systems.



## A hash-based signature system

Standardize a 256-bit hash function  $H$ .

Signer's public key: 512 strings  $y_1[0], y_1[1], \dots, y_{256}[0], y_{256}[1]$ , each 256 bits.

Total: 131072 bits.

Signature of a message  $m$ :

256-bit strings  $r, x_1, \dots, x_{256}$

such that the bits  $(h_1, \dots, h_{256})$

of  $H(r, m)$  satisfy

$$y_1[h_1] = H(x_1), \dots,$$

$$y_{256}[h_{256}] = H(x_{256}).$$

Signer's secret key:

512 independent uniform

random 256-bit strings

$x_1[0], x_1[1], \dots, x_{256}[0], x_{256}[1]$ .

Signer computes

$y_1[0], y_1[1], \dots, y_{256}[0], y_{256}[1]$

as  $H(x_1[0]), H(x_1[1]), \dots,$

$H(x_{256}[0]), H(x_{256}[1])$ .

To sign  $m$ :

generate uniform random  $r$ ;

$H(r, m) = (h_1, \dots, h_{256})$ ;

reveal  $(r, x_1[h_1], \dots, x_{256}[h_{256}])$ ;

discard remaining  $x$  values;

refuse to sign more messages.

This is the “Lamport–Diffie one-time signature system.”

How to sign  
more than one message?

Easy answer: “Chaining.”

Signer expands  $m$  to include  
a newly generated public key  
that will sign next message.

More advanced answers  
(Merkle et al.)

scale logarithmically with the  
number of messages signed.

# A code-based encryption system

Receiver's public key:

$1800 \times 3600$  bit matrix  $K$ .

Messages suitable for encryption:

3600-bit strings of "weight 150";

i.e., 3600-bit strings

with exactly 150 nonzero bits.

Encryption of  $m$

is 1800-bit string  $Km$ .

Attacker, by linear algebra,  
can easily work backwards  
from  $Km$  to *some*  $v$   
such that  $Kv = Km$ .

Huge number of choices of  $v$ .  
Finding weight-150 choice  
(“syndrome-decoding  $K$ ”)  
seems extremely difficult  
for most choices of  $K$ .

Best attacks? See next talk.

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

Detecting this structure seems even more difficult than syndrome-decoding random  $K$ .

Receiver starts with secret monic degree-150 irreducible polynomial  $g \in \mathbf{F}_{4096}[x]$  and distinct  $\alpha_1, \dots, \alpha_{3600} \in \mathbf{F}_{4096}$ .

“Patterson’s algorithm” syndrome-decodes the matrix

$$H = \begin{pmatrix} \frac{1}{g(\alpha_1)} & \dots & \frac{1}{g(\alpha_{3600})} \\ \frac{\alpha_1}{g(\alpha_1)} & \dots & \frac{\alpha_{3600}}{g(\alpha_{3600})} \\ \vdots & \ddots & \vdots \\ \frac{\alpha_1^{149}}{g(\alpha_1)} & \dots & \frac{\alpha_{3600}^{149}}{g(\alpha_{3600})} \end{pmatrix} .$$

Receiver also has  
a secret invertible  
 $1800 \times 1800$  matrix  $S$   
and a secret  $3600 \times 3600$   
permutation matrix  $P$ .

Receiver's public key  $K$   
is the product  $SHP$ .

Given ciphertext  $Km = SHPm$ :  
receiver computes  $HPm$ ;  
decodes  $H$  to obtain  $Pm$ ;  
computes  $m$ .



This is 1986 Niederreiter variant of McEliece's original system.

Reducing  $K$  to "systematic form" reduces space to 3211248 bits.

Many other improvements.

Lattice-based cryptography:  
similar; more complicated;  
maybe more attractive!

See tomorrow's talk.

## An MQ signature system

Signer's public key:

polynomials  $P_1, \dots, P_{300}$

$\in \mathbf{F}_2[w_1, \dots, w_{600}]$ .

Extra requirements

on each of these polynomials:

degree  $\leq 2$ , no squares;

i.e., linear combination of

$1, w_1, \dots, w_{600},$

$w_1 w_2, w_1 w_3, \dots, w_{599} w_{600}.$

Overall 54090300 bits.

Signature of  $m$ :

a 300-bit string  $r$  and

values  $w_1, \dots, w_{600} \in \mathbf{F}_2$

such that  $H(r, m) =$

$(P_1(w_1, \dots, w_{600}), \dots,$

$P_{300}(w_1, \dots, w_{600}))$ .

Only 900 bits!

Verifying a signature uses

one evaluation of  $H$  and

millions of bit operations

to evaluate  $P_1, \dots, P_{300}$ .

Main challenge for attacker:

find bits  $w_1, \dots, w_{600}$

producing specified outputs

$(P_1(w_1, \dots, w_{600}), \dots,$

$P_{300}(w_1, \dots, w_{600}))$ .

Random guess: on average,  
only  $2^{-300}$  chance of success.

“XL” etc.: fewer operations,  
but still not practical.

Signer generates public key  
with secret “HFE<sup>v-</sup>” structure.

Standardize a degree-450  
irreducible polynomial  $\varphi \in \mathbf{F}_2[t]$ .

Define  $L = \mathbf{F}_2[t]/\varphi$ .

Critical step in signing:  
finding roots of a  
secret polynomial in  $L[x]$   
of degree at most 300.

Secret polynomial is chosen with all nonzero exponents of the form  $2^i + 2^j$  or  $2^i$ . (So degree  $\leq 288$ .)

If  $x_0, x_1, \dots, x_{449} \in \mathbf{F}_2$  and  
 $x = x_0 + x_1 t + \dots + x_{449} t^{449}$  then  
 $x^2 = x_0 + x_1 t^2 + \dots + x_{449} t^{898}$ ,  
 $x^4 = x_0 + x_1 t^4 + \dots + x_{449} t^{1796}$ ,  
etc.

In general,  $x^{2^i + 2^j}$   
is a quadratic polynomial  
in the variables  $x_0, \dots, x_{449}$ .

Signer's secret key:

invertible  $600 \times 600$  matrix  $S$ ;

$300 \times 450$  matrix  $T$  of rank 300;

$Q \in L[x, v_1, v_2, \dots, v_{150}]$ .

Each term in  $Q$

has one of the forms

$lx^{2^i+2^j}$  with  $l \in L$ ,  $2^i < 2^j$ ,

$2^i + 2^j \leq 300$ ;

$lx^{2^i}v_j$  with  $l \in L$ ,  $2^i \leq 300$ ;

$lv_iv_j$ ;

$lx^{2^i}$ ;

$lv_j$ ;

$l$ .

To compute public key:

Compute  $S(w_1, \dots, w_{600}) = (x_0, \dots, x_{449}, v_1, \dots, v_{150})$ .

In  $L[w_1, \dots, w_{600}]$

compute  $x = \sum x_i t^i$

and  $y = Q(x, v_1, v_2, \dots, v_{150})$

modulo  $w_1^2 - w_1, \dots, w_{600}^2 - w_{600}$ .

Write  $y = y_0 + \dots + y_{449} t^{449}$

with  $y_i \in \mathbf{F}_2[w_1, \dots, w_{600}]$ .

Compute  $(P_1, \dots, P_{300}) =$

$T(y_0, y_1, \dots, y_{449})$ .



Sign by working backwards.

Given values  $(P_1, \dots, P_{300})$ , invert  $T$  to obtain values  $(y_0, \dots, y_{449})$ .  
 $2^{150}$  choices; randomize.

Choose  $(v_1, \dots, v_{150})$  randomly.  
Substitute into  $Q(x, v_1, \dots, v_{150})$   
to obtain  $Q(x) \in L[x]$ .

Solve  $Q(x) = y$  for  $x \in L$ .

If several roots, randomize.

If no roots, start over.

Invert  $S$  to obtain signature.

This is an “HFE<sup>v-</sup>” example.

“HFE”: “Hidden Field Equation”

$$Q(x) = y.$$

“-”: publish only 300 equations instead of 450.

“v”: “vinegar” variables

$$v_1, \dots, v_{150}.$$

State-of-the-art attack

breaks a simplified system with  
0 vinegar variables, 1 term in  $Q$ .

Can build MQ systems  
in many other ways.

## Preparing for the future

When someone announces a large quantum computer we'll switch to McEliece etc.

Why worry about the switch *now*?

Answer 1: We need time to improve efficiency.

Answer 2: We need time to build confidence.

Answer 3: We need time to improve usability.

Have you implemented  
a public-key system?

Send your software to eBATS:  
ECRYPT Benchmarking of  
Asymmetric Systems.

Now integrated into eBACS:  
ECRYPT Benchmarking of  
Cryptographic Systems.

<http://bench.cr.yp.to>

2003.09 Bernstein, sci.crypt:  
“I’m thinking about  
publishing a paper  
on post-quantum cryptography.  
This isn’t too early  
to start planning ahead  
for the very real possibility  
of quantum computers.”

2004.10 Buchmann et al.,  
“Post-Quantum Signatures”:  
“We would like to thank  
Dan Bernstein for inventing  
the notion ‘post-quantum  
cryptography.’ ”

2004.10 Silverberg to Bernstein:  
Hey, let's run a workshop.

2004.10 Bernstein to Silverberg:  
How about a workshop on  
post-quantum cryptography?

Independently, late 2004:  
Wolf proposes workshop on  
post-quantum cryptography.

Independently, late 2004:  
Ding proposes workshop on  
“Cryptology in the  
quantum computer era.”

2005.05 Lange manages to herd  
Bernstein, Ding, Nguyen, Wolf,  
et al.; ECRYPT starts  
organizing PQCrypto 2006.

2006.05: PQCrypto 2006  
in Leuven, Belgium.

2008.10: PQCrypto 2008  
here in Cincinnati.

End of 2008: Survey book!



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.”