Cryptography for the paranoid

Daniel J. Bernstein
(University of Illinois at Chicago,
Technische Universiteit Eindhoven)

Based on joint work with:

Tanja Lange
(Technische Universiteit Eindhoven)

Christiane Peters
(Danmarks Tekniske Universitet)

Paranoia

"They're out to get us."

Paranoia

"They're out to get us."

Who's out to get us?

Paranoia

“They’re out to get us.”

Who’s out to get us?

“The government.
That other government.
Every government.
And these corporations
making money off everything.
It’s a conspiracy, man.”

Paranoia

“They’re out to get us.”

Who’s out to get us?

“The government.
That other government.
Every government.
And these corporations
making money off everything.
It’s a conspiracy, man.”

Hmmm.
What exactly are they doing?

Cryptographic paranoia

"They're monitoring *everything*
we do on the Internet.
And they're *changing* packets
and faking *web pages* in transit
without our even noticing.
And they have huge armies of
computers analyzing everything."

Cryptographic paranoia

"They're monitoring *everything*
we do on the Internet.
And they're *changing* packets
and faking *web pages* in transit
without our even noticing.
And they have huge armies of
computers analyzing everything."

Um, okay.
Have you considered encryption?

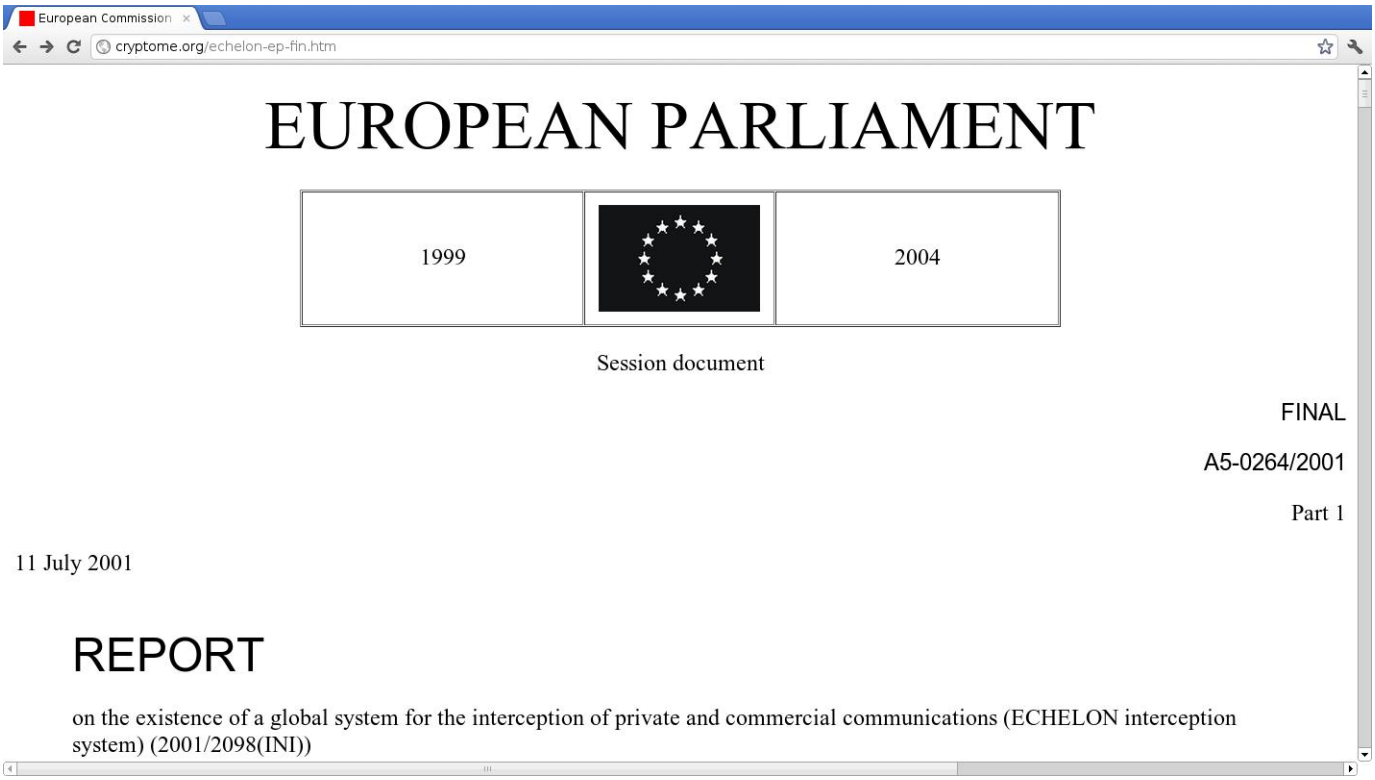
"They're *recording* everything. Even if they don't understand it today, they'll keep looking at it for *years* until they understand it. They have huge armies of *mathematicians* analyzing it. And they're working on building *quantum computers*. Encryption is dead, man."

"They're *recording* everything.
Even if they don't understand it
today, they'll keep looking at it
for *years* until they understand it.
They have huge armies of
mathematicians analyzing it.
And they're working on
building *quantum computers*.
Encryption is dead, man."

Hmmm.
Time to look at some facts.

Are they really monitoring everything?

Are they really monitoring everything?


European Commission

cryptome.org/echelon-ep-fin.htm

EUROPEAN PARLIAMENT

1999		2004

Session document

FINAL

A5-0264/2001

Part 1

11 July 2001

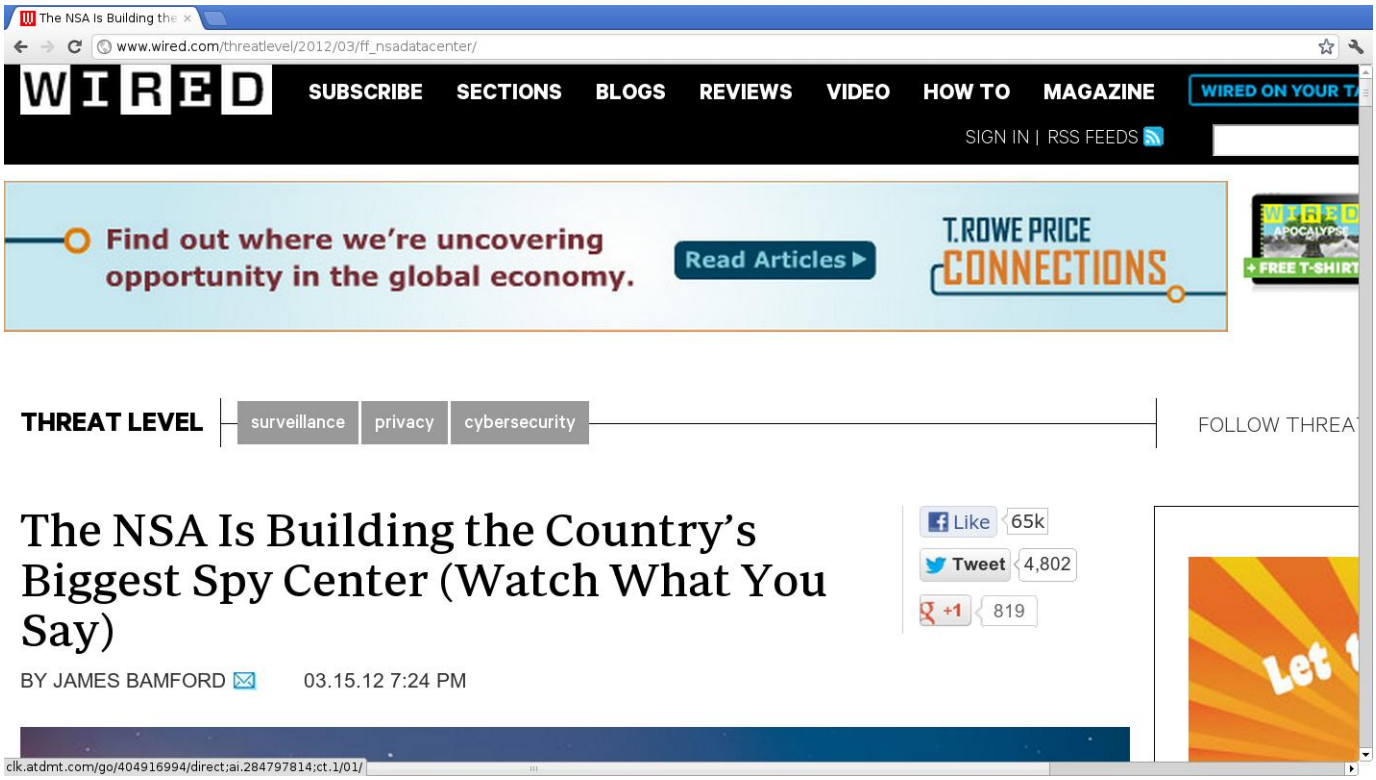
REPORT

on the existence of a global system for the interception of private and commercial communications (ECHELON interception system) (2001/2098(INI))

European Parliament: “That a global system for intercepting communications exists . . . is no longer in doubt”; “probably” this system violates European Convention on Human Rights.

Huge armies of computers analyzing everything?

Huge armies of computers analyzing everything?



New NSA data center in Utah:

$2 billion to construct;

65-megawatt power substation.

If technology is standard,

should be $\approx 2^{87}$ bit ops/year.

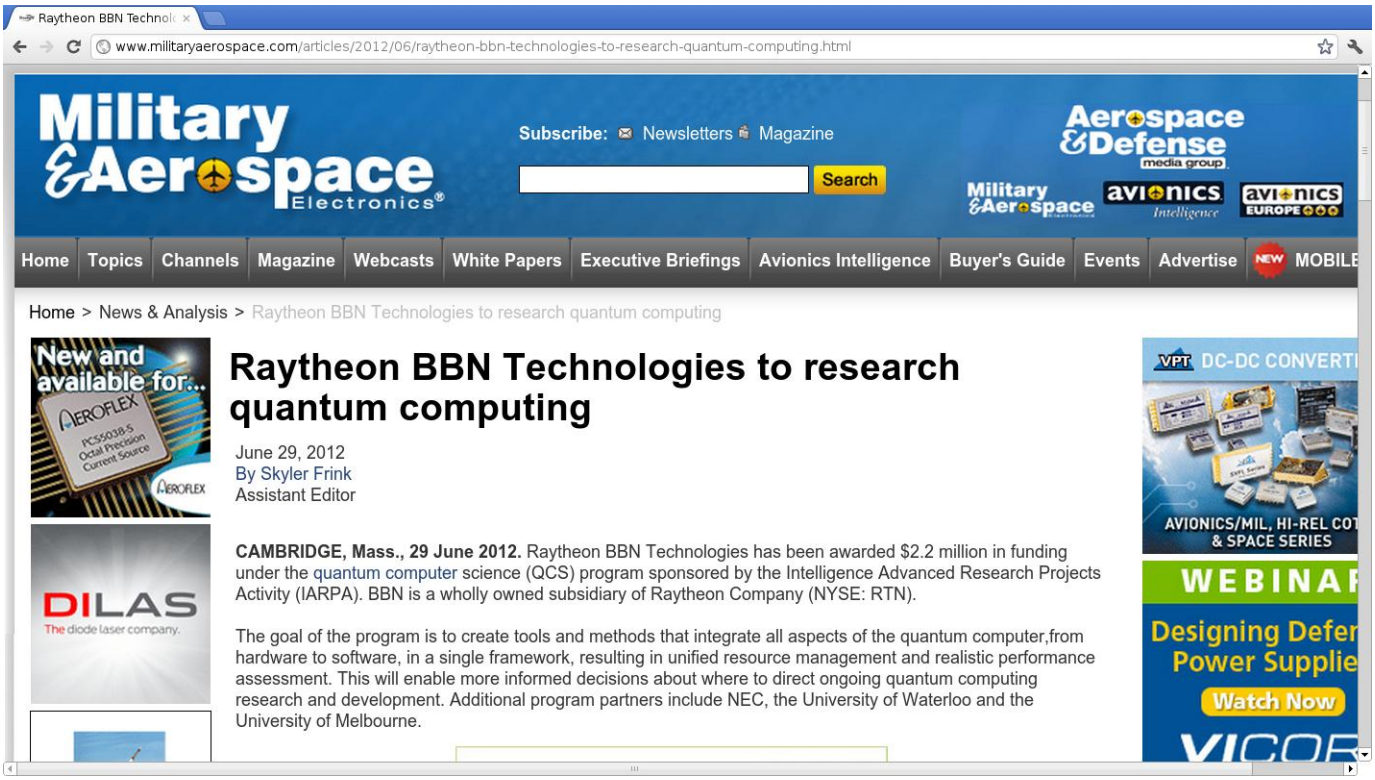
Huge armies of mathematicians trying to cryptanalyze everything?

Huge armies of mathematicians trying to cryptanalyze everything?

NSA job advertisement: "We are the largest employer of mathematicians in the country."

Working on building quantum computers?

Working on building quantum computers?



$2.2 million to Raytheon: one of many publicly announced quantum-computing grants from government agencies.

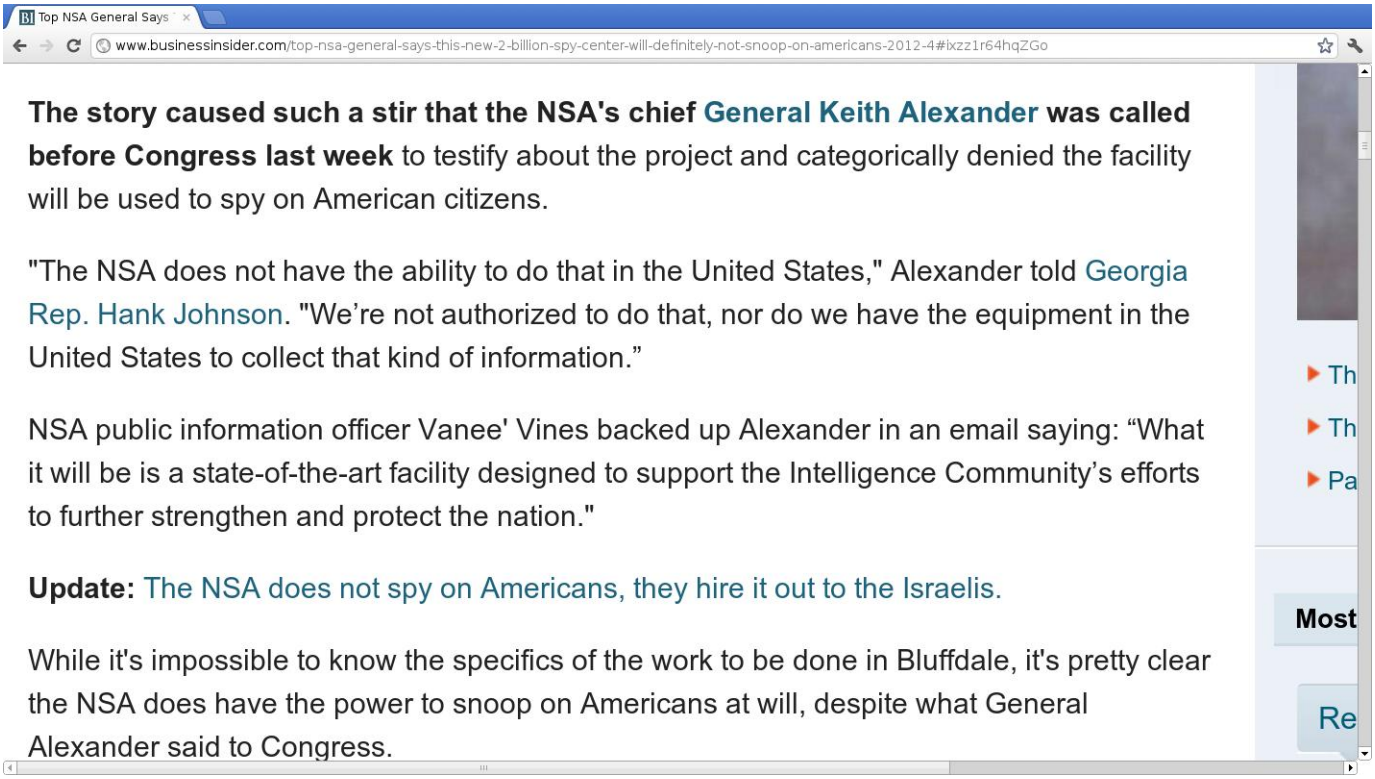
None of this justifies paranoia!

None of this justifies paranoia!

The U.S. government is a transparent, trustworthy government.

None of this justifies paranoia! The U.S. government is a transparent, trustworthy government.

U.S. government admits building the Utah data center, but says it isn't targeting Americans.



U.S. government admitted espionage operations in Europe, but said it was fighting bribery.

U.S. government admitted espionage operations in Europe, but said it was fighting bribery.

1994 example from EP report: Airbus bribed various Saudis for a $6 billion contract; NSA intercepted the faxes, exposed the bribery; MD won the contract.

U.S. government admitted espionage operations in Europe, but said it was fighting bribery.

1994 example from EP report: Airbus bribed various Saudis for a $6 billion contract; NSA intercepted the faxes, exposed the bribery; MD won the contract.

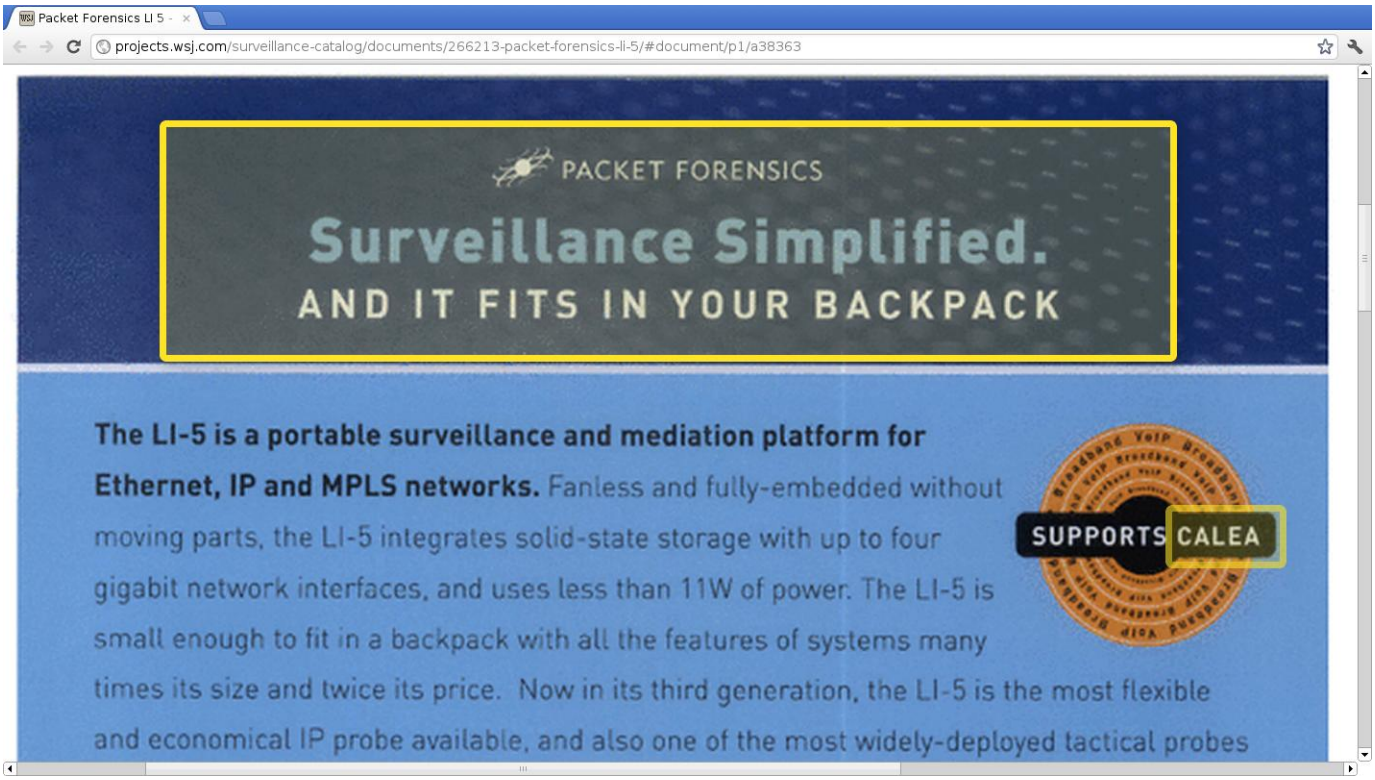
U.S. government admitted wiretapping 1960s protesters such as Martin Luther King, Jr., but said that of course it wouldn't do that sort of thing any more.

But what about
other attackers that
aren't as friendly and pure
as the U.S. government?

But what about
other attackers that
aren't as friendly and pure
as the U.S. government?

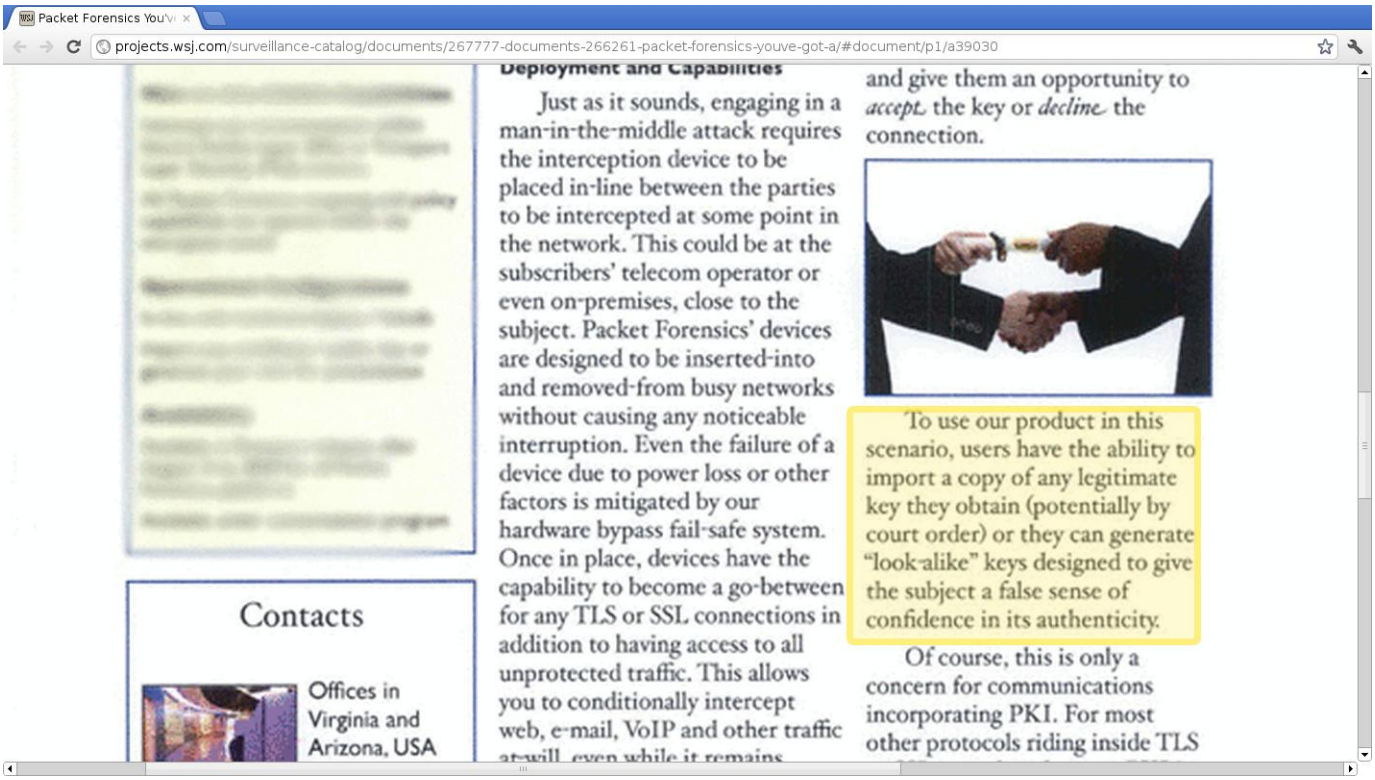
EFF: "successful man-in-the-middle attacks against hundreds of thousands of Internet users inside and outside of Iran".

Fancy attack tools are available to anyone willing to pay for them.



“Surveillance simplified.
And it fits in your backpack.”

. . . including easy-to-use tools
to modify web pages in transit.



" . . . man-in-the-middle attack
. . . designed to give the subject
a false sense of confidence
in its authenticity".

2011.10 Wall Street Journal:
“A U.S. company that
makes Internet-blocking gear
acknowledges that Syria has been
using at least 13 of its devices
to censor Web activity there.”

2011.10 Wall Street Journal:
“A U.S. company that
makes Internet-blocking gear
acknowledges that Syria has been
using at least 13 of its devices
to censor Web activity there.”

2012.02: Trustwave (one of the
SSL CAs trusted by your browser)
admits selling a transparent
HTTPS interception box
to a private company.

Cryptography for the paranoid

1994 Schneier "Applied Cryptography": "There are two kinds of cryptography in this world: cryptography that will stop your kid sister from reading your files, and cryptography that will stop major governments from reading your files.
This book is about the latter."

Cryptography for the paranoid

1994 Schneier "Applied Cryptography": "There are two kinds of cryptography in this world: cryptography that will stop your kid sister from reading your files, and cryptography that will stop major governments from reading your files. This book is about the latter."

2012: We now think that major governments can break almost everything in the book!

Problem #1:
Cryptanalytic breakthroughs.

Some systems are vulnerable
to very fast attacks
that were publicly announced
after the book appeared.

Problem #1:
Cryptanalytic breakthroughs.

Some systems are vulnerable
to very fast attacks
that were publicly announced
after the book appeared.

Paranoid approach:
Pay attention to cryptanalysis.
Use systems already subjected
to extensive public cryptanalysis,
minimizing risk of big speedups.
(Much easier now than in 1994.)

Problem #2:
Attackers doing $\gg 2^{80}$ bit ops.

e.g. Utah data center has
enough power to break many
RSA-1024 keys every year.
Botnets have similar power.
Far beyond public computations.

Problem #2:
Attackers doing $\gg 2^{80}$ bit ops.

e.g. Utah data center has
enough power to break many
RSA-1024 keys every year.
Botnets have similar power.
Far beyond public computations.

Paranoid approach: Look at
total computer power of
human race, extrapolate by years.
$\Rightarrow$ Aim for at least 2^{128}.

Problem #3:

Attackers who have access to big quantum computers.

Problem #3:
Attackers who have access to
big quantum computers.

Not just a future problem!
Attacker records everything;
eventually (10 years from now?)
builds quantum computer;
applies quantum computer
to the recorded traffic.

Problem #3:
Attackers who have access to
big quantum computers.

Not just a future problem!
Attacker records everything;
eventually (10 years from now?)
builds quantum computer;
applies quantum computer
to the recorded traffic.

Paranoid approach:
Evaluate security assuming that
attacker has quantum computer.

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.

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^{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."

Focus of this talk:
code-based cryptography.

Extensive analysis of McEliece cryptosystem since 1978.
Cryptanalytic progress has had only small effect on key size (and CPU time) for 2^{128} security.
Confidence-inspiring!

Focus of this talk:
code-based cryptography.

Extensive analysis of McEliece cryptosystem since 1978.
Cryptanalytic progress has had only small effect on key size (and CPU time) for 2^{128} security.
Confidence-inspiring!

But maybe can do even better.
We'll see some low-cost modifications to McEliece that seem to pose extra annoyances for cryptanalysts.

Outside scope of this talk:

Encrypt with RSA-16384
and codes *and* lattices
in case one idea is broken?

Outside scope of this talk:

Encrypt with RSA-16384
and codes *and* lattices
in case one idea is broken?

Or use same resources to
encrypt with much larger codes?

Outside scope of this talk:

Encrypt with RSA-16384
and codes *and* lattices
in case one idea is broken?

Or use same resources to
encrypt with much larger codes?

Also use physical techniques:
locked-briefcase cryptography,
quantum key distribution, etc.?
Very expensive, hard to secure,
but maybe not totally obsolete.

Outside scope of this talk:

Encrypt with RSA-16384
and codes *and* lattices
in case one idea is broken?

Or use same resources to
encrypt with much larger codes?

Also use physical techniques:
locked-briefcase cryptography,
quantum key distribution, etc.?
Very expensive, hard to secure,
but maybe not totally obsolete.

Security beyond cryptography?
PKI, buffer overflows, ...

The McEliece cryptosystem

(with 1986 Niederreiter speedup)

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 $Km \in \mathbf{F}_2^{500}$.

Use hash of (m, Km)
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 $\mathrm{Ker}K$
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.

Bad estimate by McEliece: $\approx 2^{64}$.

Long history, many improvements:
1962 Prange; 1981 Omura;
1988 Lee–Brickell; 1988 Leon;
1989 Krouk; 1989 Stern;
1989 Dumer;
1990 Coffey–Goodman;
1990 van Tilburg; 1991 Dumer;
1991 Coffey–Goodman–Farrell;
1993 Chabanne–Courteau;
1993 Chabaud;
1994 van Tilburg;
1994 Canteaut–Chabanne;
1998 Canteaut–Chabaud;
1998 Canteaut–Sendrier.
$\approx 2^{70}$ cycles.

2008 Bernstein–Lange–Peters:
further improvements;
$\approx 2^{60}$ cycles;
carried out successfully!

More recent literature:
2009 Bernstein–Lange–
Peters–van Tilborg;
2009 Bernstein;
2009 Finiasz–Sendrier;
2010 Bernstein–Lange–Peters;
2011 May–Meurer–Thomae;
2011 Becker–Coron–Joux;
2012 Becker–Joux–May–Meurer.

Modern McEliece

Easily rescue system by using a larger public key: "random" $\approx(n/2) \times n$ 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. Also 2009 Bernstein: $2^{n/(4 \lg n)}$ quantum.

How does the receiver
decode these errors, anyway?
Why weight $n/(2 \lg n)$?

Outline of answer:
Receiver has a secret,
a fast decoding algorithm D.
Receiver generates K as a
random (or systematic) matrix
with $\mathrm{Ker} K = \{\text{outputs of } D\}$.

Let's look at the details.
Why do we get $n/(2 \lg n)$ errors?
Why is it hard for attacker to
work backwards from K to D?

Reed–Solomon codes

Fix a prime power q.

Write $\alpha_1, \alpha_2, \ldots, \alpha_q$
for the elements of $\mathbf{F}_q$
in a standard order.

Fix an integer t with $0 \leq t < q$.

$$\{(f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_q)) : f \in \mathbf{F}_q[x], \deg f < q - t\}$$

is the (q, t) Reed–Solomon code.

(1960 Reed–Solomon,
described differently)

This is a "$[q, q-t, t+1]_q$" code:
it is a $(q-t)$-dimensional
$\mathbf{F}_q$-subspace of $\mathbf{F}_q^q$;
it has minimum distance $t+1$.

1960 Peterson:
$q^{O(1)}$ arithmetic ops
to correct $\lfloor t/2 \rfloor$ errors.

1968 Berlekamp: $O(q^2)$.
Modern view: Reduce
a 2-dimensional lattice basis.

1976 Justesen,
independently 1977 Sarwate:
$q(\lg q)^{2+o(1)}$. Modern view:
fast lattice-basis reduction.

Receiver builds secret decoder
by starting from RS decoder,
choosing defenses to add.

Several interesting defenses:

- Scaling.
- Permutation.
- Puncturing.
- $\mathbf{F}_q$-subcodes.
- Subfield.
- Wildness.
- List decoding.
- Increased genus.

Scaling

Scaling a code $C \subseteq \mathbf{F}_q^n$
by $(\beta_1, \ldots, \beta_n) \in (\mathbf{F}_q^*)^n$
produces $\{(\beta_1 c_1, \ldots, \beta_n c_n) :$
$(c_1, \ldots, c_n) \in C\}$.

Same length, dimension, distance.

To decode scaled code:

divide, decode C, multiply.

Scaled RS code:

$\{(\beta_1 f(\alpha_1), \ldots, \beta_q f(\alpha_q)) :$
$f \in \mathbf{F}_q[x],\ \deg f < q - t\}$.

Permutation

Permuting a code $C \subseteq \mathbf{F}_q^n$
by a permutation π of $\{1, \ldots, n\}$
produces $\{(c_{\pi(1)}, \ldots, c_{\pi(n)}) :$
$(c_1, \ldots, c_n) \in C\}$.

Same length, dimension, distance.

To decode permuted code:
unpermute, decode C, permute.

Permuted scaled RS code:
$\{(\beta_1 f(\alpha_1), \ldots, \beta_q f(\alpha_q)) :$
$f \in \mathbf{F}_q[x],\ \deg f < q - t\}$
where $\alpha_1, \alpha_2, \ldots, \alpha_q$ are
the elements of $\mathbf{F}_q$ in any order.

Puncturing

Puncturing a code $C \subseteq \mathbf{F}_q^n$
at position 1 produces
$\{(c_2, \ldots, c_n) :$
$(c_1, c_2, \ldots, c_n) \in C\}$.
Similarly can puncture at
any subset of $\{1, \ldots, n\}$.

Generalized RS code = punctured
permuted scaled RS code:
$\{(\beta_1 f(\alpha_1), \ldots, \beta_n f(\alpha_n)) :$
$f \in \mathbf{F}_q[x], \deg f < n - t\}$
where $\alpha_1, \alpha_2, \ldots, \alpha_n$ are
distinct elements of $\mathbf{F}_q$.

This is an $[n, n-t, t+1]_q$ code (assuming $0 \leq t < n \leq q$).
Most RS decoders easily generalize to GRS decoders.

This is an $[n, n-t, t+1]_q$ code (assuming $0 \leq t < n \leq q$).
Most RS decoders easily generalize to GRS decoders.

"Look at all these secrets! Attacker can't search through all the possibilities."

This is an $[n, n-t, t+1]_q$ code (assuming $0 \leq t < n \leq q$).
Most RS decoders easily generalize to GRS decoders.

"Look at all these secrets!
Attacker can't search through all the possibilities."

But it turns out that the structure isn't hidden well enough.
1992 Sidelnikov–Shestakov broke scaling+permutation+puncturing in polynomial time.

How the attack works:

K allows attacker to
generate random codewords.
Attacker is also free to
add more linear constraints.

Attacker generates a random
shortened codeword: a codeword
with 0 in last $n-t-1$ coordinates.

This codeword has the form
$(\beta_1 f(\alpha_1), \ldots, \beta_n f(\alpha_n))$ where
$\alpha_{t+2}, \ldots, \alpha_n$ are roots of f.

i.e. $(\beta_1 f(\alpha_1), \ldots, \beta_n f(\alpha_n))$
where $f = c(x-\alpha_{t+2}) \cdots (x-\alpha_n)$.
If $c = 0$, try again.

Swap $t+1$ with n: obtain
$(\beta_1 g(\alpha_1), \ldots, \beta_n g(\alpha_n))$ where
$g = d(x-\alpha_{t+1}) \cdots (x-\alpha_{n-1})$.

Divide $\beta_i f(\alpha_i)$ by $\beta_i g(\alpha_i)$ to
obtain $(c/d)(\alpha_i - \alpha_n)/(\alpha_i - \alpha_{t+1})$
for each $i \leq t$.

Guess (or presume) $\alpha_1, \alpha_{t+1}, \alpha_n$;
deduce $c/d, \alpha_2, \ldots, \alpha_t$;
similary deduce other α_i;
deduce $(\beta_1 : \beta_2 : \ldots : \beta_n)$.

$\mathbf{F}_q$-subcodes

Take a code $C \subseteq \mathbf{F}_q^n$.
Add several random linear constraints to build a random $\mathbf{F}_q$-linear subcode of C.

Same decoder, same length, slightly reduced dimension.
Eliminates polynomials such as $(x - \alpha_{t+2}) \cdots (x - \alpha_n)$.

2005 Berger–Loidreau proposed scaling+permutation+subcodes.

Scaling+permutation+puncturing +subcodes broken by 2006/2009 Wieschebrink for many/almost all parameter settings.

Basic idea: multiply
$(\beta_1 f(\alpha_1), \ldots, \beta_n f(\alpha_n))$,
$(\beta_1 g(\alpha_1), \ldots, \beta_n g(\alpha_n))$
to obtain
$(\beta_1^2 h(\alpha_1), \ldots, \beta_n^2 h(\alpha_n))$
with $h = fg$.

Apply 1992 Sidelnikov–Shestakov to h; also to f, g if h is too big.

Subfield

Assume $q = 2^m$ for simplicity.
The $\mathbf{F}_2$-subfield subcode
of $C \subseteq \mathbf{F}_q^n$ is $\mathbf{F}_2^n \cap C$.

Same decoder, same length.
Simple dimension bound:
$n - \dim_{\mathbf{F}_2}(\mathbf{F}_2^n \cap C)$
$\leq m(n - \dim_{\mathbf{F}_q} C)$.

$\mathbf{F}_2$-alternant code $=$ $\mathbf{F}_2$-subfield
subcode of GRS code:
$\{(\beta_1 f(\alpha_1), \ldots, \beta_n f(\alpha_n)) \in \mathbf{F}_2^n :$
$f \in \mathbf{F}_q[x], \deg f < n - t\}$.

$[n, \geq n - mt, \geq t + 1]_2$ code.

(1974 Helgert, independently
1975 Chien–Choy, independently
1975 Delsarte)

Drastic restriction on f.
Clear quantitative barrier to
Sidelnikov–Shestakov etc.:
$n/m - t$ equations $f(\alpha_i) = 0$
$\Rightarrow n - mt$ equations over $\mathbf{F}_2$,
typically forcing $f = 0$.

Wildness

For $g \in \mathbf{F}_q[x]$, all $g(\alpha_i) \neq 0$:
The classical binary Goppa code
$\Gamma_2(\alpha_1, \ldots, \alpha_n, g)$
is the $\mathbf{F}_2$-alternant code
with $\beta_i = g(\alpha_i)/h'(\alpha_i)$
and $t = \deg g$.

Here $h = (x - \alpha_1) \cdots (x - \alpha_n)$.

(1970 Goppa, 1971 Goppa)

Note that scaling and subfield
are prerequisites for wildness.

If g is a square
and $\sqrt{g}$ is squarefree
then $\Gamma_2(g) = \Gamma_2(\sqrt{g})$.
(1975 Sugiyama–Kasahara–
Hirasawa–Namekawa)

$[n, \geq n - m(t/2), \geq t + 1]_2$ code
where $t = \deg g$.
(alternate proof that $\Gamma_2(\sqrt{g})$ has
these parameters: 1970 Goppa)

Compared to generic β_i,
much better tradeoff between
dimension and error correction.

Generalize: improved dimension bounds for any powers in g.
(1975 Sugiyama–Kasahara–Hirasawa–Namekawa)

"BCH codes" $g = x^t$
maximize these dimension bounds.
(introduction of BCH codes
and these bounds: 1959
Hocquenghem, independently
1960 Bose–Ray-Chaudhuri)

Speculative disadvantage of wildness: somewhat special choice of β_i; maybe attacker can somehow exploit this.

Hmmm. Is this really paranoid?

Speculative disadvantage of wildness: somewhat special choice of β_i; maybe attacker can somehow exploit this.

Hmmm. Is this really paranoid?

Gigantic advantage of wildness: for same code length and same code dimension, use *twice as many errors*, drastically slowing down ISD.

1978 McEliece used scaling+ permutation+subfield+wildness.

Didn't puncture: $n = q = 2^m$.

Chose rate $\approx 1/2$:
$m(t/2) \approx n/2$, i.e., $n \approx mt$.
(Now well known: this rate is suboptimal; rate 0.8 is better.)

Corrected $t/2$ errors;
i.e., $n/(2 \lg n)$ errors.

2010 Bernstein–Lange–Peters: generalize beyond $\mathbf{F}_2$; obtain better security for (e.g.) $\mathbf{F}_{11}$.

"Support splitting" algorithm (2000 Sendrier) finds permutation if everything else is known.

Can attack McEliece by applying support splitting to each possibility for g.

This is much slower than ISD: too many possibilities for g. But immediately breaks scaling+ permutation+subfield+wildness with, e.g., BCH codes $g = x^t$.

New challenge: break
scaling+permutation+puncturing
+subcode+subfield+wildness
for BCH codes.

Slightly better parameters
than original McEliece system.

Puncturing seems to stop
support splitting.
Subcodes also seem to stop
support splitting.
Subfields stop other attacks.

Clearly more paranoid:
scaling+permutation+puncturing
+subcode+subfield+wildness
with random Goppa codes.

Support splitting
now has three obstacles:
guessing the puncturing;
guessing the subcode;
guessing g.

No disadvantages compared to
original McEliece system.

List decoding

1997 Sudan: in poly time
decode many RS codes
beyond $\lfloor t/2 \rfloor$ errors.

1998 Guruswami–Sudan:
up to big-field Johnson bound.

2000 Koetter–Vardy:
up to $\mathbf{F}_2$ Johnson bound,
when errors are in $\mathbf{F}_2$.

Can go beyond this bound:
see, e.g., 2011 Bernstein.

Speed of list decoding
is an active research area.

Clearly practical to correct
at least a few extra errors.
This makes ISD much slower.

No change in code.
No disadvantages other than
decoding time.

List decoding can produce
multiple codewords, but
"CCA2 conversion" automatically
selects the right codeword.

Increased genus (AG codes)

1980 Goppa generalized RS codes to AG codes: similar parameters but pushing length beyond q.

Extensive subsequent work on AG decoding algorithms.

Increased genus (AG codes)

1980 Goppa generalized RS codes to AG codes: similar parameters but pushing length beyond q.

Extensive subsequent work on AG decoding algorithms.

1996 Janwa–Moreno proposed replacing RS codes in McEliece with AG codes of higher genus.

Increased genus (AG codes)

1980 Goppa generalized RS codes to AG codes: similar parameters but pushing length beyond q.

Extensive subsequent work on AG decoding algorithms.

1996 Janwa–Moreno proposed replacing RS codes in McEliece with AG codes of higher genus.

Several followup attacks; very bad reputation.

This reputation is undeserved.

The successful attacks are
on AG *without subfields*.
We use RS with subfields;
also use AG with subfields!

Moving to higher genus
is clearly a helpful step:
adds to difficulty of ISD
and of many other attacks.

Best option seems to be scaling
+permutation+puncturing+subcode
+subfield+wildness+list decoding
+increased genus.