# The first 10 years of Curve25519

Daniel J. Bernstein

University of Illinois at Chicago &
Technische Universiteit Eindhoven

---

2005.05.19: Seminar talk; design+software close to done.

2005.09.15: Software online.

2005.09.20: Invited talk at ECC.

2005.11.15: Paper online; submitted to PKC 2006.



Abstract: "This paper explains the design and implementation of a high-security elliptic-curve-Diffie-Hellman function achieving record-setting speeds: e.g., 832457 Pentium III cycles (with several side benefits: free key compression, free key validation, and state-of-the-art timing-attack protection), more than twice as fast as other authors' results at the same conjectured security level (with or without the side benefits)."

t 10 years of Curve25519

. Bernstein

ty of Illinois at Chicago &
che Universiteit Eindhoven

.19: Seminar talk;
software close to done.

.15: Software online.

.20: Invited talk at ECC.

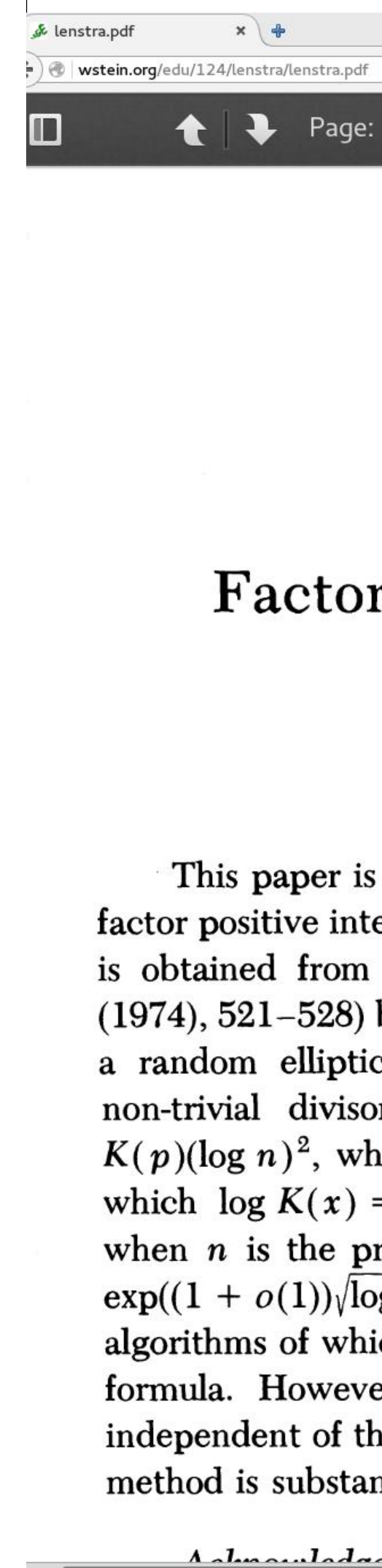
.15: Paper online;
ed to PKC 2006.



Abstract: "This paper explains the design and implementation of a high-security elliptic-curve-Diffie-Hellman function achieving record-setting speeds: e.g., 832457 Pentium III cycles (with several side benefits: free key compression, free key validation, and state-of-the-art timing-attack protection), more than twice as fast as other authors' results at the same conjectured security level (with or without the side benefits)."



Elliptic-c


lenstra.pdf
wstein.org/edu/124/lenstra/lenstra.pdf
Page:

Factor

This paper is
factor positive inte
is obtained from
(1974), 521–528) l
a random elliptic
non-trivial diviso
$K(p)(\log n)^2$, wh
which $\log K(x)$ =
when $n$ is the pr
$\exp((1 + o(1))\sqrt{\log$
algorithms of whi
formula. Howeve
independent of th
method is substan

of Curve25519

n

is at Chicago &
siteit Eindhoven

nar talk;
lose to done.

vare online.

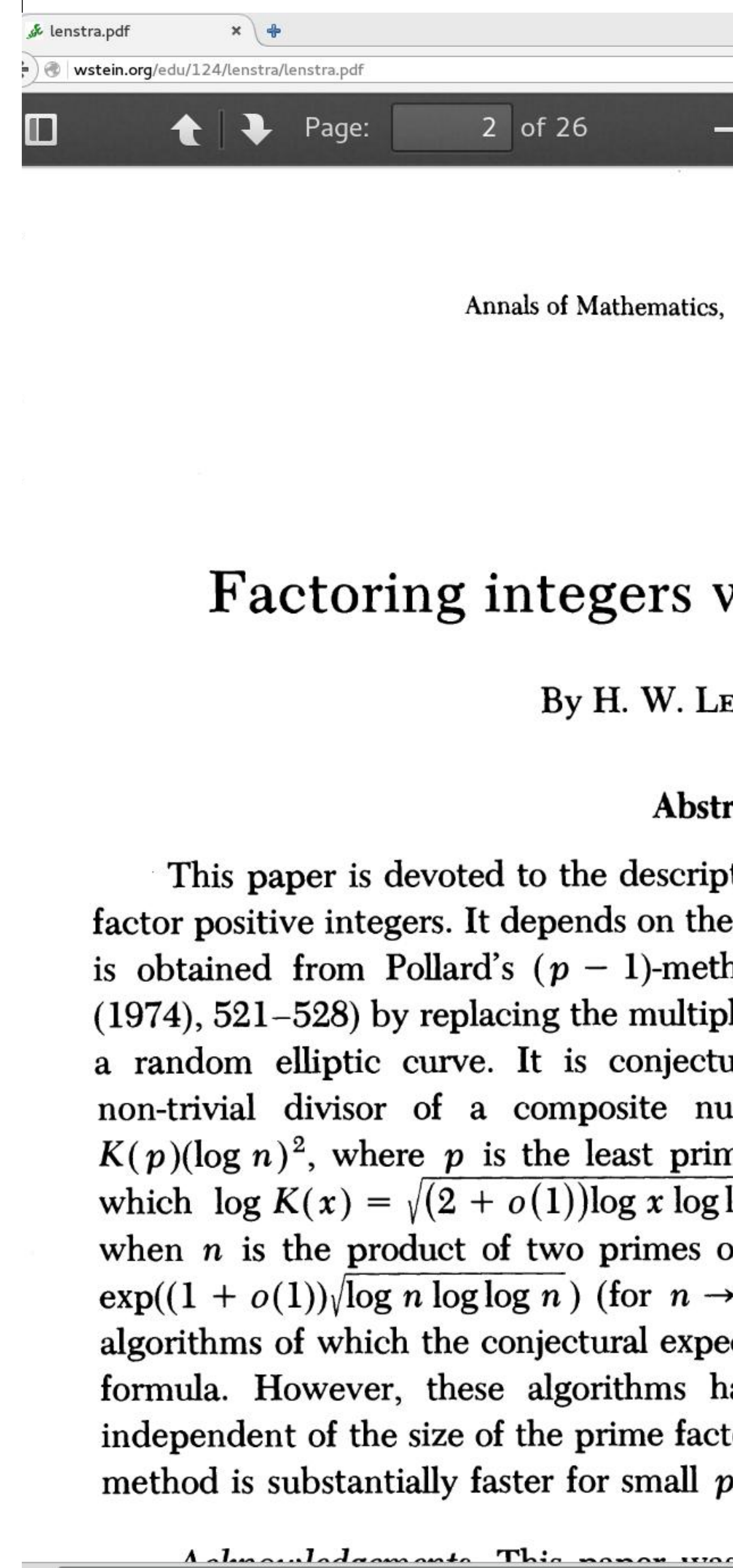
ed talk at ECC.

r online;
2006.



Abstract: "This paper explains the design and implementation of a high-security elliptic-curve-Diffie-Hellman function achieving record-setting speeds: e.g., 832457 Pentium III cycles (with several side benefits: free key compression, free key validation, and state-of-the-art timing-attack protection), more than twice as fast as other authors' results at the same conjectured security level (with or without the side benefits)."

Elliptic-curve com



5519

ago &
hoven

ne.

.

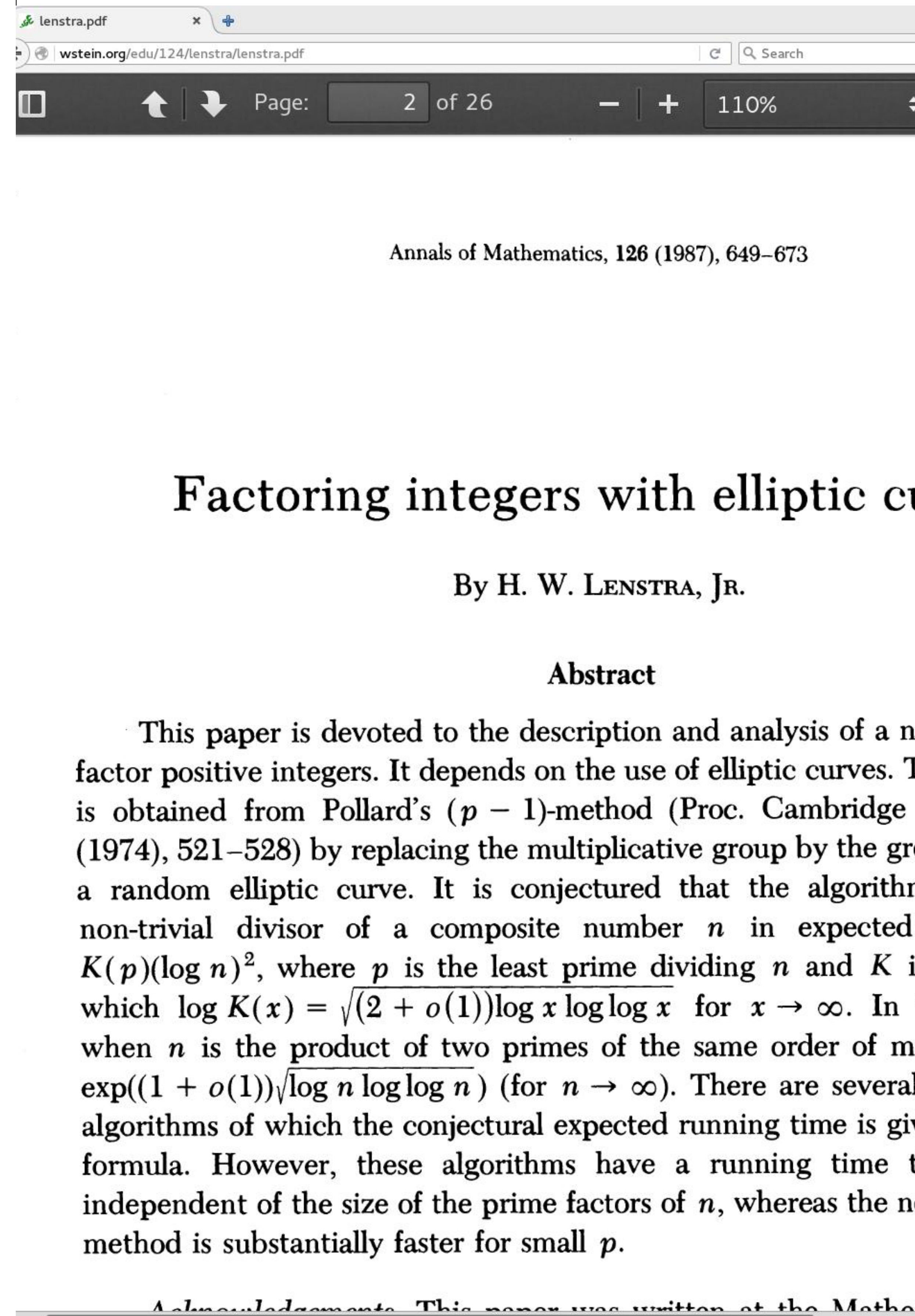
ECC.



Abstract: "This paper explains the design and implementation of a high-security elliptic-curve-Diffie-Hellman function achieving record-setting speeds: e.g., 832457 Pentium III cycles (with several side benefits: free key compression, free key validation, and state-of-the-art timing-attack protection), more than twice as fast as other authors' results at the same conjectured security level (with or without the side benefits)."



Elliptic-curve computations



Annals of Mathematics, 126 (1987), 649–673

Factoring integers with elliptic cu

By H. W. Lenstra, Jr.

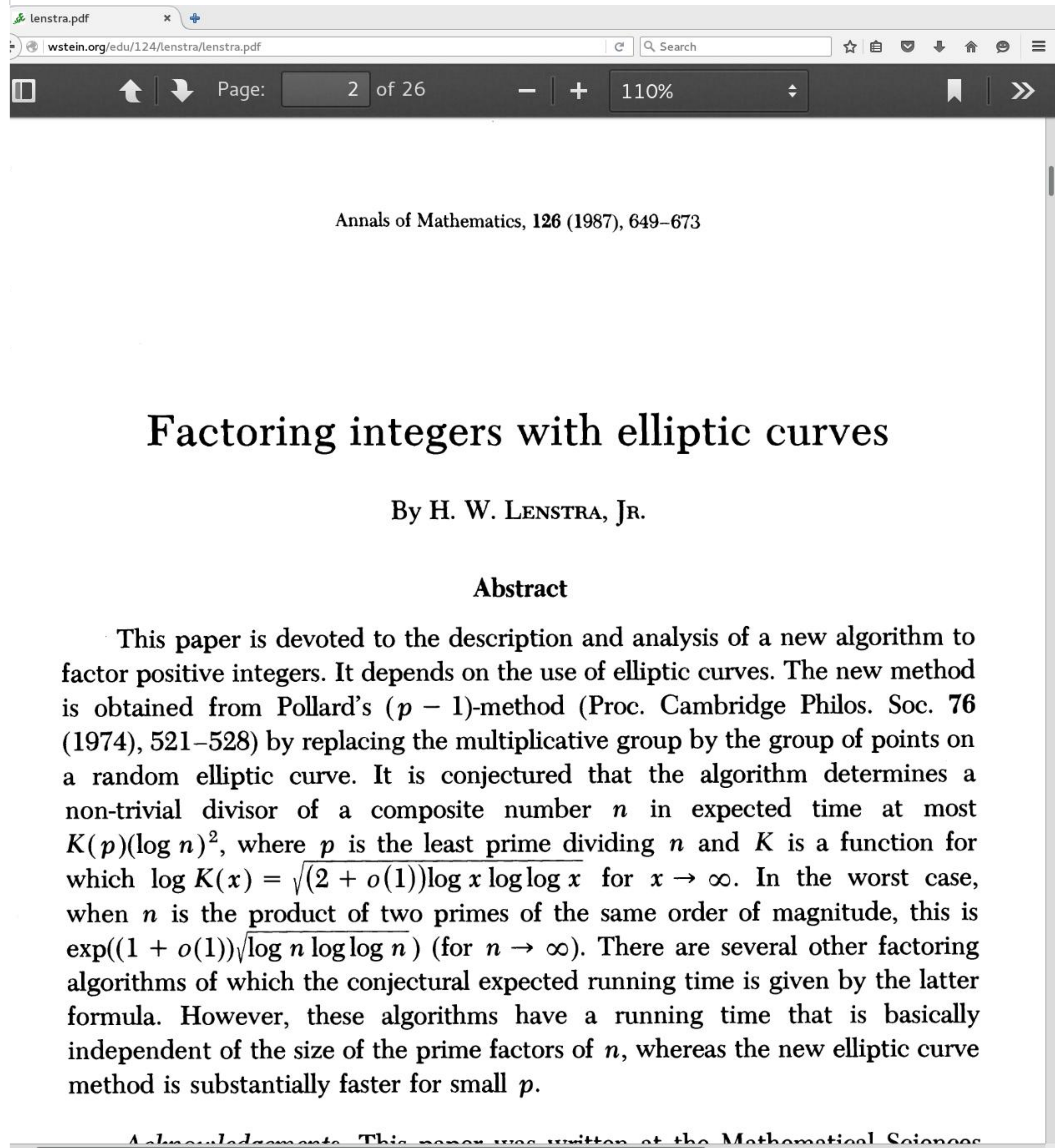
Abstract

This paper is devoted to the description and analysis of a n
factor positive integers. It depends on the use of elliptic curves. T
is obtained from Pollard's $(p-1)$-method (Proc. Cambridge
(1974), 521–528) by replacing the multiplicative group by the gr
a random elliptic curve. It is conjectured that the algorithm
non-trivial divisor of a composite number $n$ in expected
$K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ i
which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In
when $n$ is the product of two primes of the same order of m
$\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several
algorithms of which the conjectural expected running time is giv
formula. However, these algorithms have a running time t
independent of the size of the prime factors of $n$, whereas the ne
method is substantially faster for small $p$.

Abstract: "This paper explains the design and implementation of a high-security elliptic-curve-Diffie-Hellman function achieving record-setting speeds: e.g., 832457 Pentium III cycles (with several side benefits: free key compression, free key validation, and state-of-the-art timing-attack protection), more than twice as fast as other authors' results at the same conjectured security level (with or without the side benefits)."



## Elliptic-curve computations




Annals of Mathematics, **126** (1987), 649–673

# Factoring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

This paper is devoted to the description and analysis of a new algorithm to factor positive integers. It depends on the use of elliptic curves. The new method is obtained from Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76** (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm determines a non-trivial divisor of a composite number $n$ in expected time at most $K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ is a function for which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case, when $n$ is the product of two primes of the same order of magnitude, this is $\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several other factoring algorithms of which the conjectural expected running time is given by the latter formula. However, these algorithms have a running time that is basically independent of the size of the prime factors of $n$, whereas the new elliptic curve method is substantially faster for small $p$.

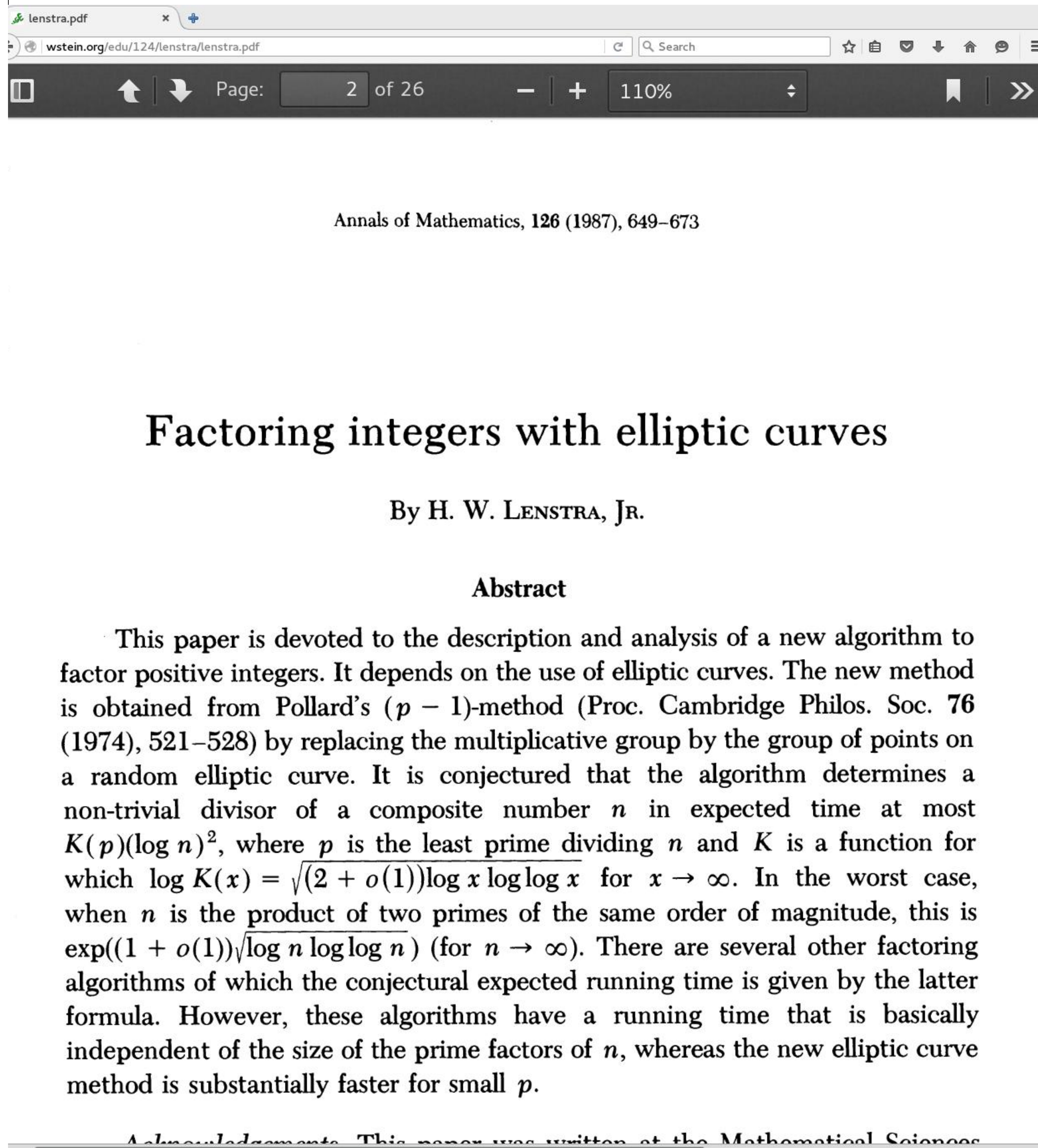
*Acknowledgements.* This paper was written at the Mathematical Sciences

t: "This paper explains
gn and implementation
n-security elliptic-curve-
ellman function
g record-setting speeds:
2457 Pentium III cycles
veral side benefits:
compression, free key
on, and state-of-the-art
ttack protection),
an twice as fast as other
results at the same
red security level (with
ut the side benefits)."



# Elliptic-curve computations






Annals of Mathematics, **126** (1987), 649–673

## Factoring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

This paper is devoted to the description and analysis of a new algorithm to factor positive integers. It depends on the use of elliptic curves. The new method is obtained from Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76** (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm determines a non-trivial divisor of a composite number $n$ in expected time at most $K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ is a function for which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case, when $n$ is the product of two primes of the same order of magnitude, this is $\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several other factoring algorithms of which the conjectural expected running time is given by the latter formula. However, these algorithms have a running time that is basically independent of the size of the prime factors of $n$, whereas the new elliptic curve method is substantially faster for small $p$.

1987 (di
ECM, th
of factor

1985 Bo
Kilian, 1
Chudnov
elliptic-c

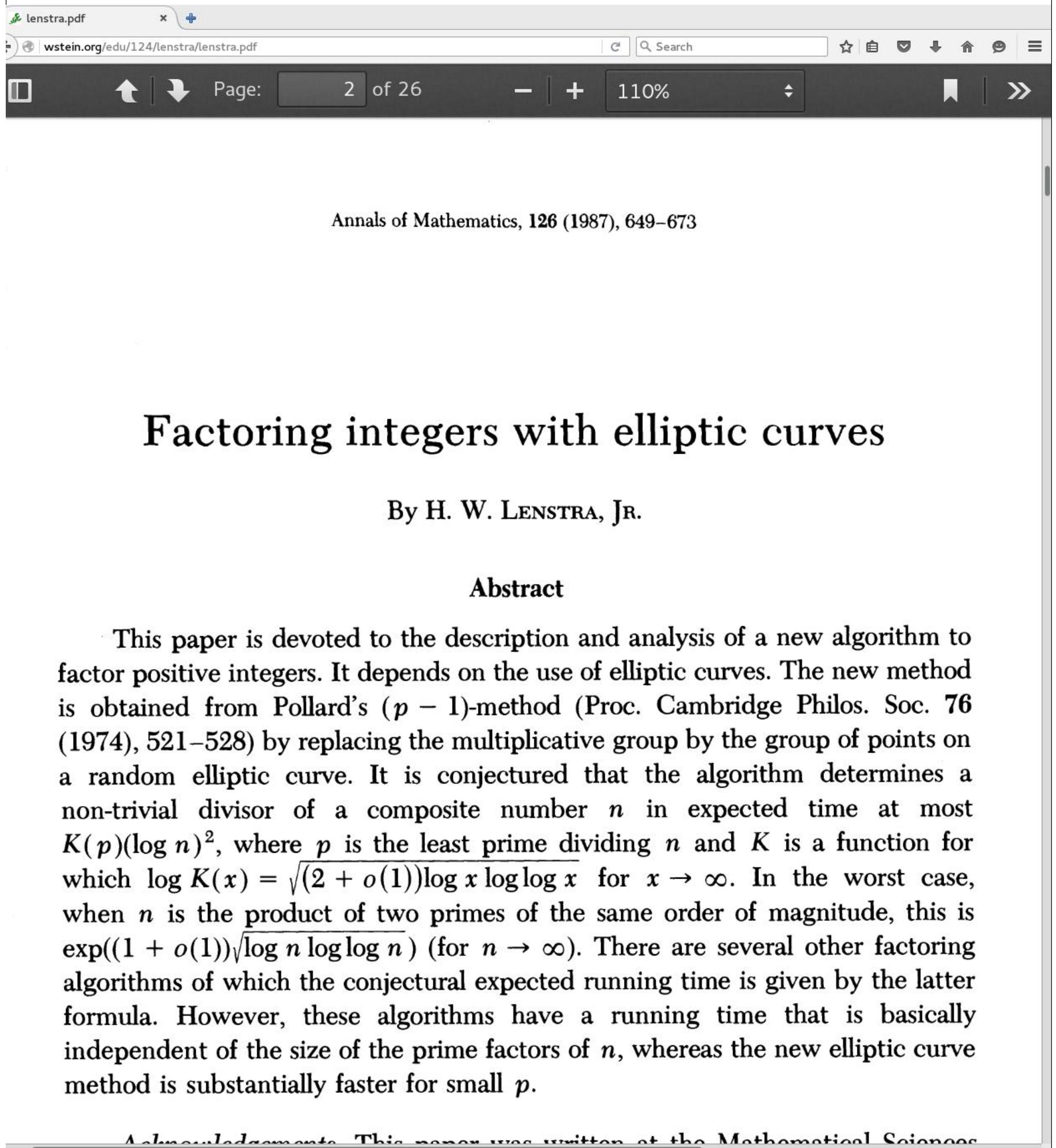
1985/6 (
and inde
1987 (di
ECC—u
to avoid

aper explains
plementation
elliptic-curve-
ction
etting speeds:
um III cycles
benefits:
on, free key
te-of-the-art
ection),
s fast as other
 the same
ty level (with
e benefits)."





## Elliptic-curve computations




Annals of Mathematics, **126** (1987), 649–673

### Factoring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

This paper is devoted to the description and analysis of a new algorithm to factor positive integers. It depends on the use of elliptic curves. The new method is obtained from Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76** (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm determines a non-trivial divisor of a composite number $n$ in expected time at most $K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ is a function for which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case, when $n$ is the product of two primes of the same order of magnitude, this is $\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several other factoring algorithms of which the conjectural expected running time is given by the latter formula. However, these algorithms have a running time that is basically independent of the size of the prime factors of $n$, whereas the new elliptic curve method is substantially faster for small $p$.

*Acknowledgements.* This paper was written at the Mathematical Sciences

1987 (distributed
ECM, the elliptic-c
of factoring intege

1985 Bosma, 1986
Kilian, 1986 Chud
Chudnovsky, 1988
elliptic-curve prima

1985/6 (distribute
and independently
1987 (distributed
ECC—use elliptic
to avoid index-calc

ins
ion
rve-

eds:
cles

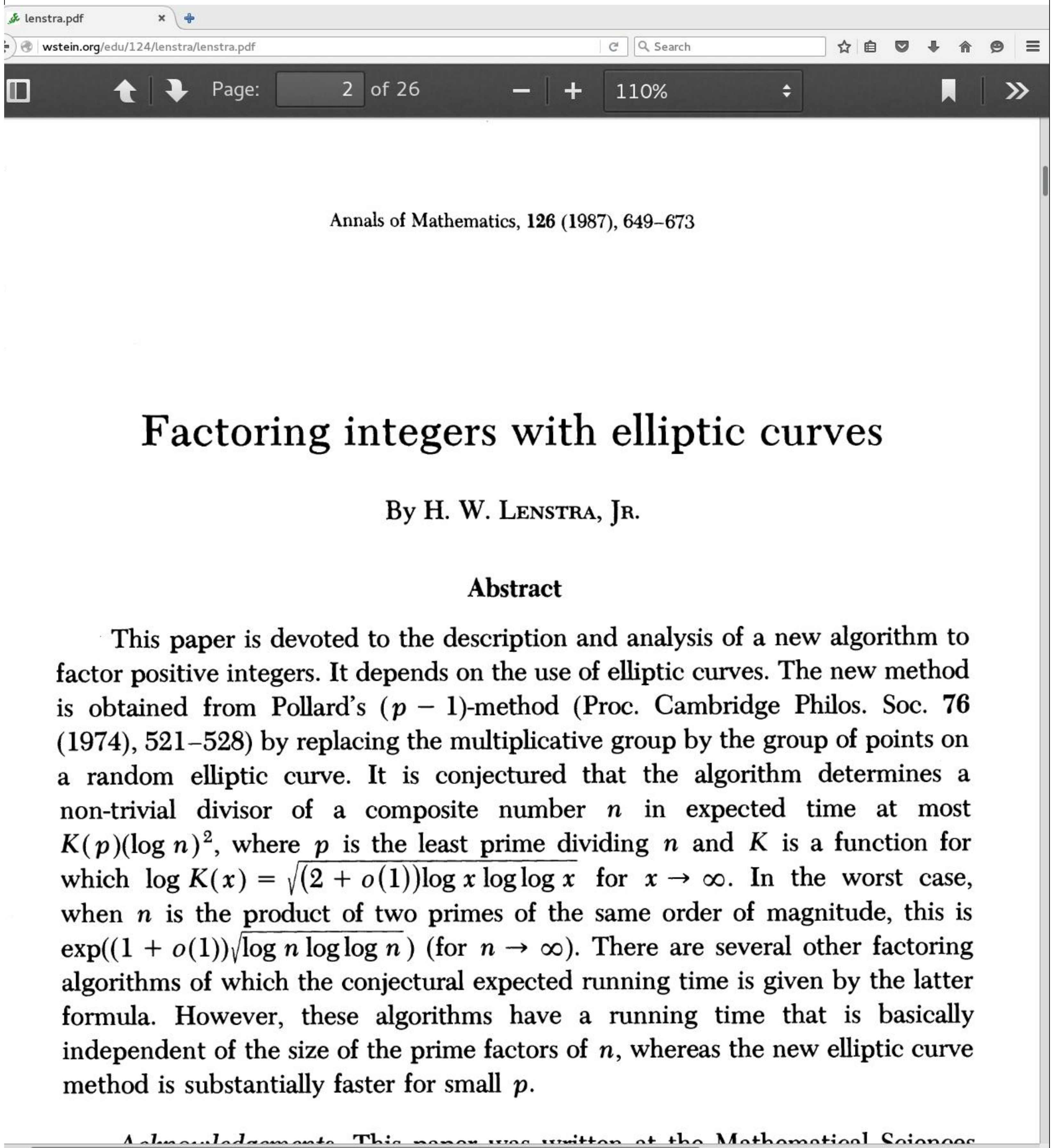
ey
art

other

vith
)."



# Elliptic-curve computations




Annals of Mathematics, **126** (1987), 649–673

## Factoring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

This paper is devoted to the description and analysis of a new algorithm to factor positive integers. It depends on the use of elliptic curves. The new method is obtained from Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76** (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm determines a non-trivial divisor of a composite number $n$ in expected time at most $K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ is a function for which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case, when $n$ is the product of two primes of the same order of magnitude, this is $\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several other factoring algorithms of which the conjectural expected running time is given by the latter formula. However, these algorithms have a running time that is basically independent of the size of the prime factors of $n$, whereas the new elliptic curve method is substantially faster for small $p$.

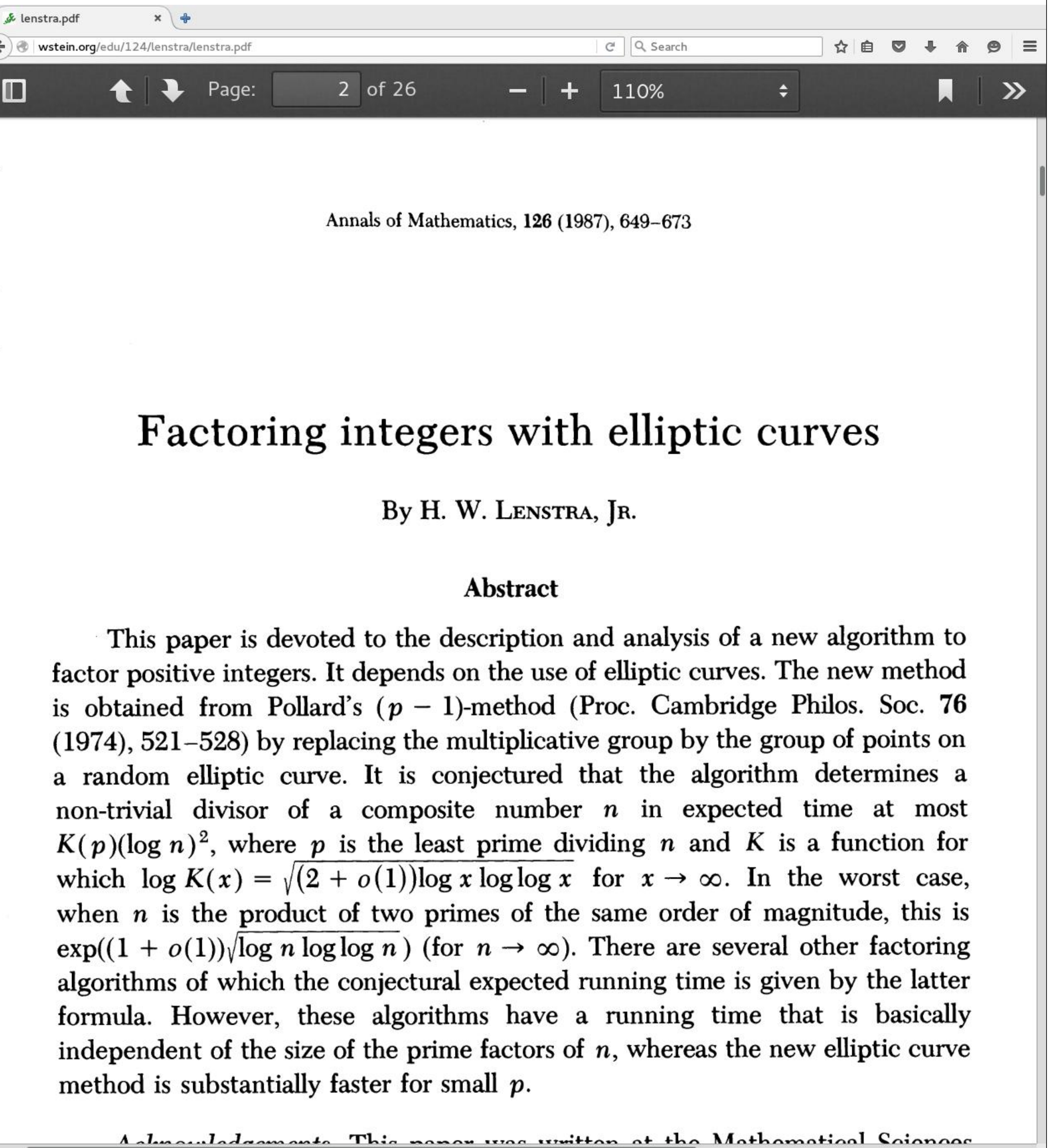


1987 (distributed 1984) Len
ECM, the elliptic-curve meth
of factoring integers.

1985 Bosma, 1986 Goldwass
Kilian, 1986 Chudnovsky–
Chudnovsky, 1988 Atkin: EC
elliptic-curve primality provi

1985/6 (distributed 1984) M
and independently
1987 (distributed 1984) Kob
ECC—use elliptic curves in
to avoid index-calculus attac

# Elliptic-curve computations


lenstra.pdf

wstein.org/edu/124/lenstra/lenstra.pdf

Search

Page: 2 of 26

110%

Annals of Mathematics, **126** (1987), 649–673

## Factoring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

This paper is devoted to the description and analysis of a new algorithm to factor positive integers. It depends on the use of elliptic curves. The new method is obtained from Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76** (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm determines a non-trivial divisor of a composite number $n$ in expected time at most $K(p)(\log n)^2$, where $p$ is the least prime dividing $n$ and $K$ is a function for which $\log K(x) = \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case, when $n$ is the product of two primes of the same order of magnitude, this is $\exp((1+o(1))\sqrt{\log n \log\log n})$ (for $n \to \infty$). There are several other factoring algorithms of which the conjectural expected running time is given by the latter formula. However, these algorithms have a running time that is basically independent of the size of the prime factors of $n$, whereas the new elliptic curve method is substantially faster for small $p$.

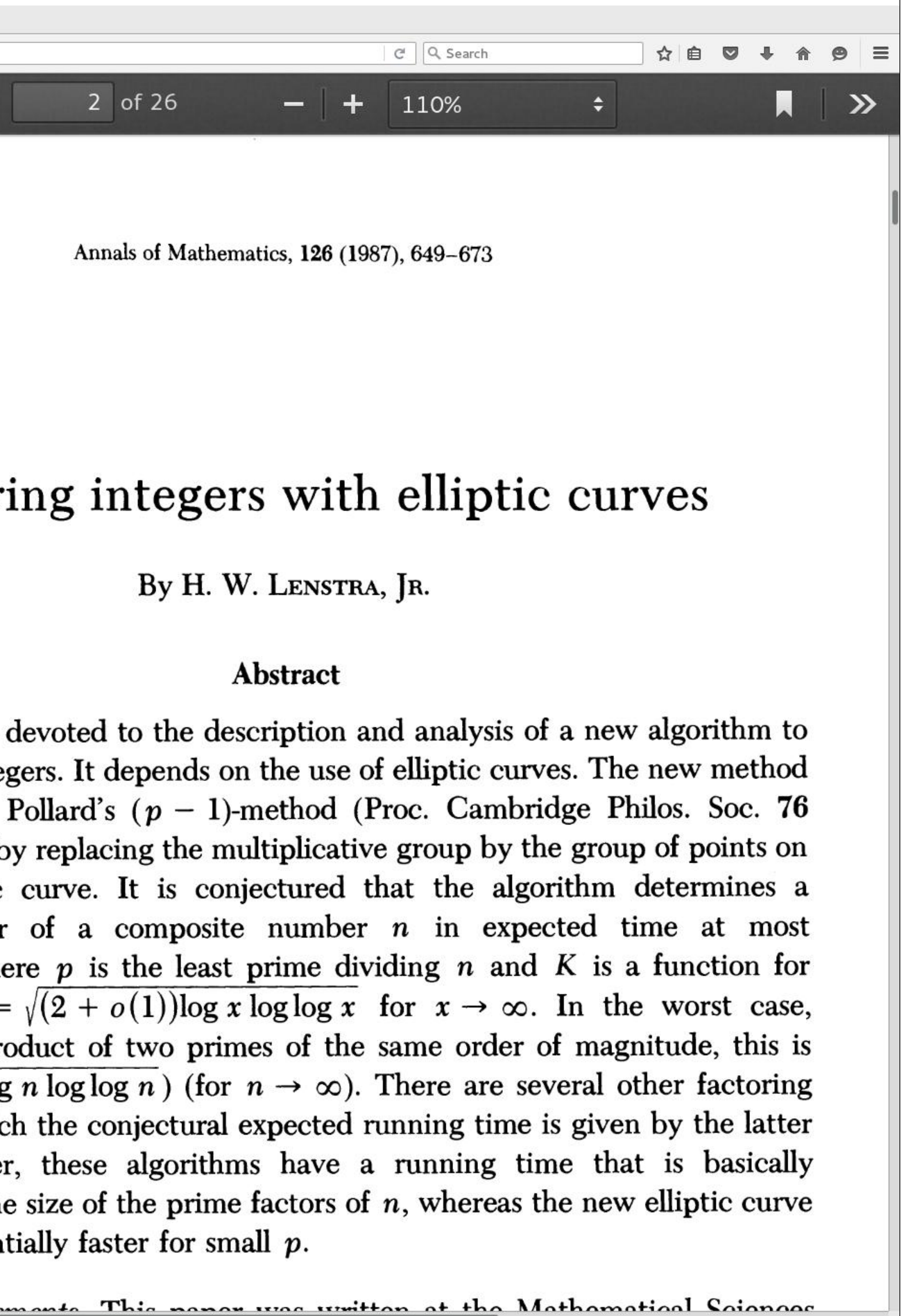


1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.

## curve computations


2 of 26
110%

Annals of Mathematics, **126** (1987), 649–673

ring integers with elliptic curves

By H. W. Lenstra, Jr.

**Abstract**

devoted to the description and analysis of a new algorithm to
egers. It depends on the use of elliptic curves. The new method
Pollard's $(p-1)$-method (Proc. Cambridge Philos. Soc. **76**
by replacing the multiplicative group by the group of points on
curve. It is conjectured that the algorithm determines a
r of a composite number $n$ in expected time at most
ere $p$ is the least prime dividing $n$ and $K$ is a function for
$= \sqrt{(2+o(1))\log x \log\log x}$ for $x \to \infty$. In the worst case,
roduct of two primes of the same order of magnitude, this is
g $n \log\log n$) (for $n \to \infty$). There are several other factoring
ch the conjectural expected running time is given by the latter
r, these algorithms have a running time that is basically
e size of the prime factors of $n$, whereas the new elliptic curve
tially faster for small $p$.



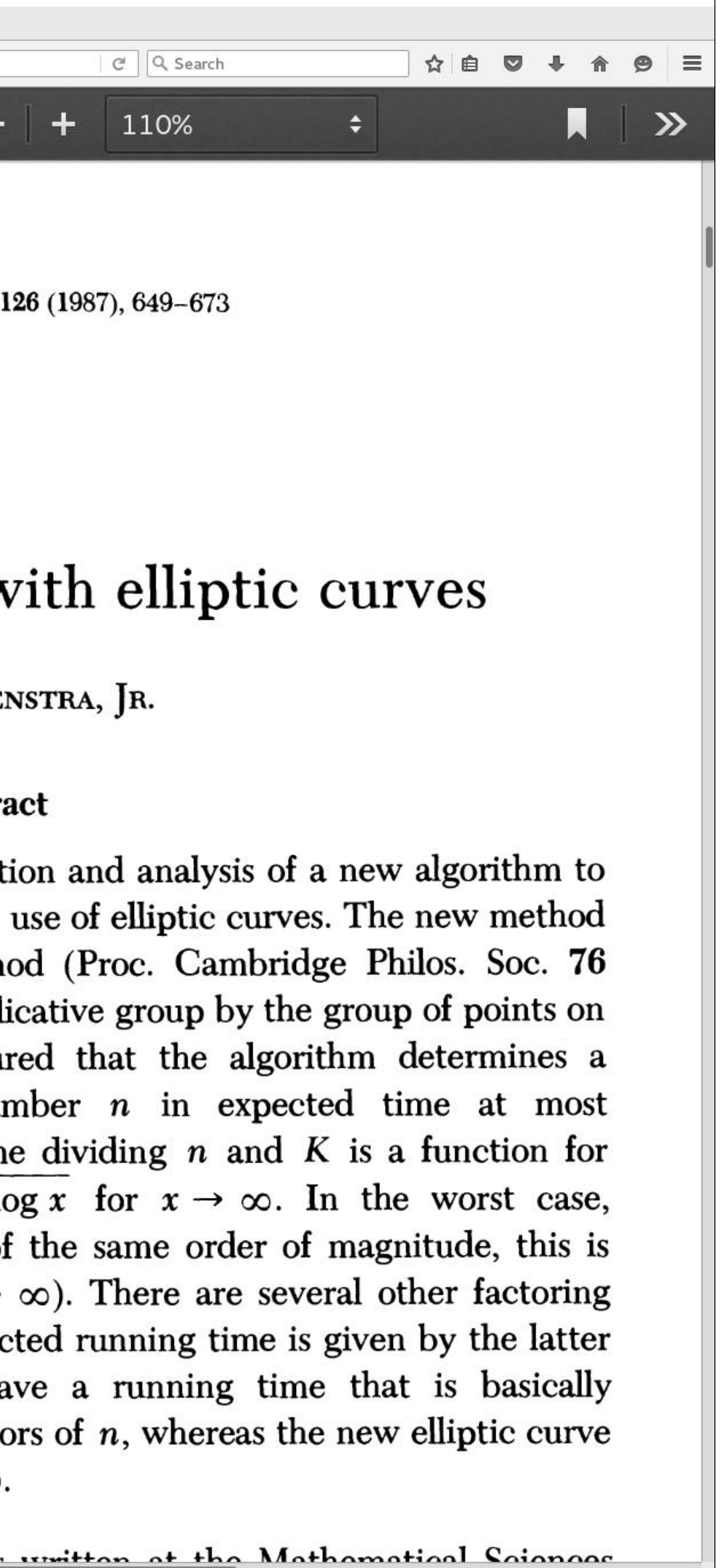
1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.

1986 Ch
for ECM
ways to
optimize

putations




126 (1987), 649–673

vith elliptic curves

ENSTRA, JR.

ract

tion and analysis of a new algorithm to
use of elliptic curves. The new method
nod (Proc. Cambridge Philos. Soc. 76
licative group by the group of points on
ured that the algorithm determines a
mber $n$ in expected time at most
ne dividing $n$ and $K$ is a function for
og $x$ for $x \to \infty$. In the worst case,
f the same order of magnitude, this is
$\infty$). There are several other factoring
cted running time is given by the latter
ave a running time that is basically
ors of $n$, whereas the new elliptic curve
).

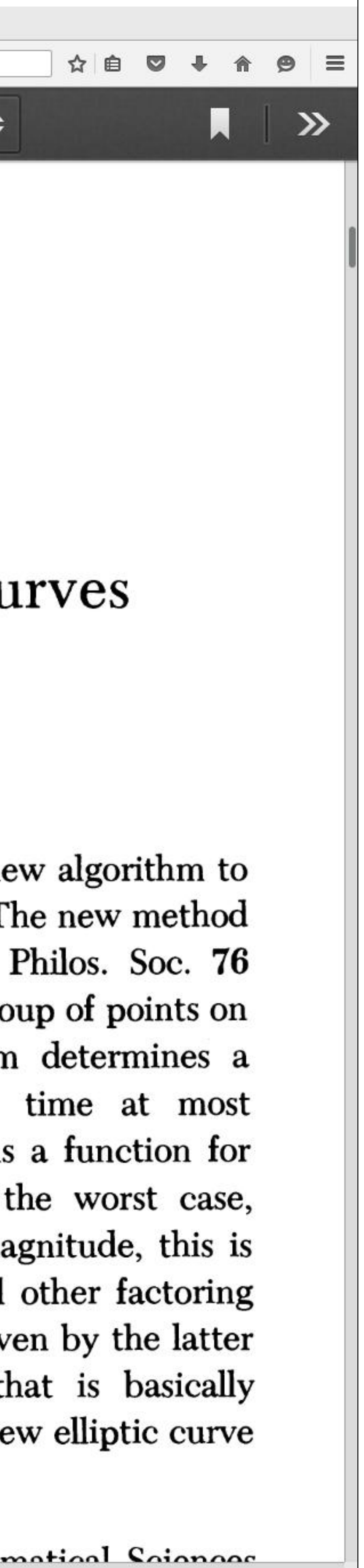


1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.

1986 Chudnovsky–
for ECM+ECPP: a
ways to represent
optimize # field o


urves

ew algorithm to
The new method
Philos. Soc. **76**
oup of points on
n determines a
time at most
s a function for
the worst case,
agnitude, this is
l other factoring
ven by the latter
that is basically
ew elliptic curve

1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.



1986 Chudnovsky–Chudnovs
for ECM+ECPP: analyze se
ways to represent elliptic cu
optimize # field operations.

1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.



1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.



1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

1987 (distributed 1984) Lenstra: ECM, the elliptic-curve method of factoring integers.

1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently
1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.



1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".

stributed 1984) Lenstra:
ne elliptic-curve method
ring integers.

osma, 1986 Goldwasser–
1986 Chudnovsky–
vsky, 1988 Atkin: ECPP,
curve primality proving.

(distributed 1984) Miller,
ependently
stributed 1984) Koblitz:
se elliptic curves in DH
index-calculus attacks.



1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".

Did Chu
actually
What ab
What ab

1984) Lenstra:
curve method
ers.

Goldwasser–
novsky–
Atkin: ECPP,
ality proving.

ed 1984) Miller,

1984) Koblitz:
curves in DH
culus attacks.





1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3+Ax^2+x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".

Did Chudnovsky a
actually recommen
What about Mont
What about paper

stra:
nod

ser–

CPP,
ng.

Miller,

olitz:
DH
cks.



1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".



Did Chudnovsky and Chudn
actually recommend this?
What about Montgomery?
What about papers after 19

1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".



Did Chudnovsky and Chudnovsky actually recommend this?
What about Montgomery?
What about papers after 1987?

1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".



Did Chudnovsky and Chudnovsky actually recommend this? What about Montgomery? What about papers after 1987?

Analyze all known options for computing $n, P \mapsto nP$ on conservative elliptic curves. Montgomery ladder is the fastest.

1986 Chudnovsky–Chudnovsky, for ECM+ECPP: analyze several ways to represent elliptic curves; optimize # field operations.

1987 Montgomery, for ECM: best speed from $y^2 = x^3 + Ax^2 + x$, preferably with $(A-2)/4$ small.

Late 1990s: ANSI/IEEE/NIST standards specify $y^2 = x^3 - 3x + b$ in Jacobian coordinates, citing Chudnovsky–Chudnovsky. Alleged motivation: "the fastest arithmetic on elliptic curves".



Did Chudnovsky and Chudnovsky actually recommend this? What about Montgomery? What about papers after 1987?

Analyze all known options for computing $n, P \mapsto nP$ on conservative elliptic curves. Montgomery ladder is the fastest.

Problem: Elliptic-curve formulas always have exceptional cases. Montgomery derives formulas for *generic* inputs; for crypto we need algorithms that *always* work.

udnovsky–Chudnovsky,
l+ECPP: analyze several
represent elliptic curves;
e # field operations.

ontgomery, for ECM:
ed from $y^2 = x^3 + Ax^2 + x$,
ly with $(A-2)/4$ small.

90s: ANSI/IEEE/NIST
ls specify $y^2 = x^3 - 3x + b$
bian coordinates,
hudnovsky–Chudnovsky.
motivation: "the fastest
tic on elliptic curves".



Did Chudnovsky and Chudnovsky actually recommend this?
What about Montgomery?
What about papers after 1987?

Analyze all known options for computing $n, P \mapsto nP$ on conservative elliptic curves.
Montgomery ladder is the fastest.

Problem: Elliptic-curve formulas always have exceptional cases.
Montgomery derives formulas for *generic* inputs; for crypto we need algorithms that *always* work.

JOURNAL OF NUMBE

Con

laws on E exists.
sisting of bihom
explicitly by Lang
there are complet
addition laws in s

THEOREM 1.
*laws on E equals*
*each of them has*

We can describ
the zero addition
call two addition

-Chudnovsky,
analyze several
elliptic curves;
perations.

, for ECM:
$^2 = x^3 + Ax^2 + x$,
$- 2)/4$ small.

/IEEE/NIST
$y^2 = x^3 - 3x + b$
nates,
–Chudnovsky.
n: “the fastest
tic curves”.





Did Chudnovsky and Chudnovsky
actually recommend this?
What about Montgomery?
What about papers after 1987?

Analyze all known options
for computing $n, P \mapsto nP$
on conservative elliptic curves.
Montgomery ladder is the fastest.

Problem: Elliptic-curve formulas
always have exceptional cases.
Montgomery derives formulas for
*generic* inputs; for crypto we need
algorithms that *always* work.

JOURNAL OF NUMBER THEORY **53**, 229–240 (199

Complete Systems of
for Ellipti

W. Bo

*Department of Pure Mathem*
*Sydney, New South W*

AN

H. W. Len

*Department of Mathematic*
*Berkeley, Califor*

laws on $E$ exists. Indeed, a complete syst
sisting of bihomogeneous polynomials
explicitly by Lange and Ruppert [2; cf. 1
there are complete systems consisting o
addition laws in such a system are neces

THEOREM 1. *The smallest cardinalit*
*laws on E equals two, and if two additio*
*each of them has bidegree* (2, 2).

We can describe all addition laws of
the zero addition law, for which *all* pa
call two addition laws *equivalent* if th

sky,
veral
rves;

:
$x^2+x$,
mall.

ST
$3x+b$

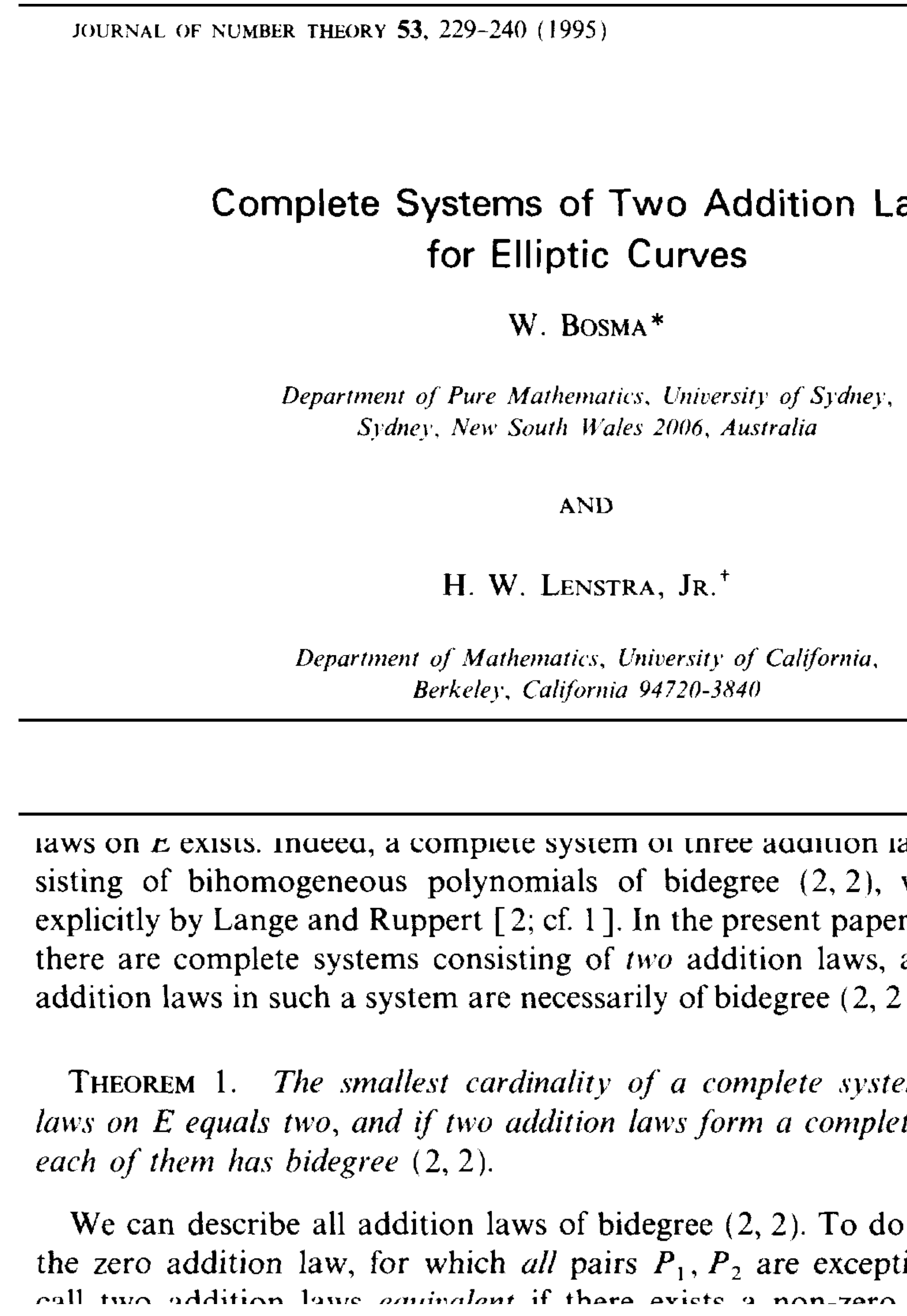
sky.
stest
’.



Did Chudnovsky and Chudnovsky actually recommend this?
What about Montgomery?
What about papers after 1987?

Analyze all known options for computing $n, P \mapsto nP$ on conservative elliptic curves.
Montgomery ladder is the fastest.

Problem: Elliptic-curve formulas always have exceptional cases.
Montgomery derives formulas for *generic* inputs; for crypto we need algorithms that *always* work.


JOURNAL OF NUMBER THEORY 53, 229–240 (1995)

Complete Systems of Two Addition La
for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney,*
*Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California,*
*Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition la
sisting of bihomogeneous polynomials of bidegree (2, 2), v
explicitly by Lange and Ruppert [2; cf. 1]. In the present paper
there are complete systems consisting of *two* addition laws, a
addition laws in such a system are necessarily of bidegree (2, 2

THEOREM 1. *The smallest cardinality of a complete syste*
*laws on E equals two, and if two addition laws form a complet*
*each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do
the zero addition law, for which *all* pairs $P_1$, $P_2$ are excepti
call two addition laws *equivalent* if there exists a non-zero

Did Chudnovsky and Chudnovsky actually recommend this?

What about Montgomery?

What about papers after 1987?

Analyze all known options for computing $n, P \mapsto nP$ on conservative elliptic curves. Montgomery ladder is the fastest.

Problem: Elliptic-curve formulas always have exceptional cases. Montgomery derives formulas for *generic* inputs; for crypto we need algorithms that *always* work.



JOURNAL OF NUMBER THEORY **53**, 229–240 (1995)

## Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1$, $P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $\lambda \in k$

dnovsky and Chudnovsky
recommend this?
bout Montgomery?
bout papers after 1987?

all known options
puting $n, P \mapsto nP$
ervative elliptic curves.
mery ladder is the fastest.

: Elliptic-curve formulas
ave exceptional cases.
mery derives formulas for
inputs; for crypto we need
ms that *always* work.

JOURNAL OF NUMBER THEORY **53**, 229–240 (1995)

Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on $E$ equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1, P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $\lambda \in k$



But wait

Crypto 1
secret br
this leak

nd Chudnovsky
nd this?
gomery?
rs after 1987?

options
$P \mapsto nP$
liptic curves.
er is the fastest.

curve formulas
tional cases.
es formulas for
crypto we need
*ways* work.



JOURNAL OF NUMBER THEORY **53**, 229–240 (1995)

Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1$, $P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $d \in k$



But wait, it's wors

Crypto 1996 Koch
secret branches af
this leaks your sec

ovsky

87?

es.

stest.

ulas

es.

as for

e need

.

JOURNAL OF NUMBER THEORY 53, 229–240 (1995)

## Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on $E$ equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1, P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $d \in k$



But wait, it's worse!

Crypto 1996 Kocher:

secret branches affect timing

this leaks your secret key.

JOURNAL OF NUMBER THEORY 53, 229–240 (1995)

## Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.†

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1, P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $d \in k$



# But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

JOURNAL OF NUMBER THEORY 53, 229–240 (1995)

## Complete Systems of Two Addition Laws for Elliptic Curves

W. BOSMA*

*Department of Pure Mathematics, University of Sydney, Sydney, New South Wales 2006, Australia*

AND

H. W. LENSTRA, JR.⁺

*Department of Mathematics, University of California, Berkeley, California 94720-3840*

laws on $E$ exists. Indeed, a complete system of three addition laws, each consisting of bihomogeneous polynomials of bidegree (2, 2), was exhibited explicitly by Lange and Ruppert [2; cf. 1]. In the present paper we show that there are complete systems consisting of *two* addition laws, and that both addition laws in such a system are necessarily of bidegree (2, 2).

THEOREM 1. *The smallest cardinality of a complete system of addition laws on E equals two, and if two addition laws form a complete system then each of them has bidegree* (2, 2).

We can describe all addition laws of bidegree (2, 2). To do this, we omit the zero addition law, for which *all* pairs $P_1$, $P_2$ are exceptional, and we call two addition laws *equivalent* if there exists a non-zero element $d \in k$

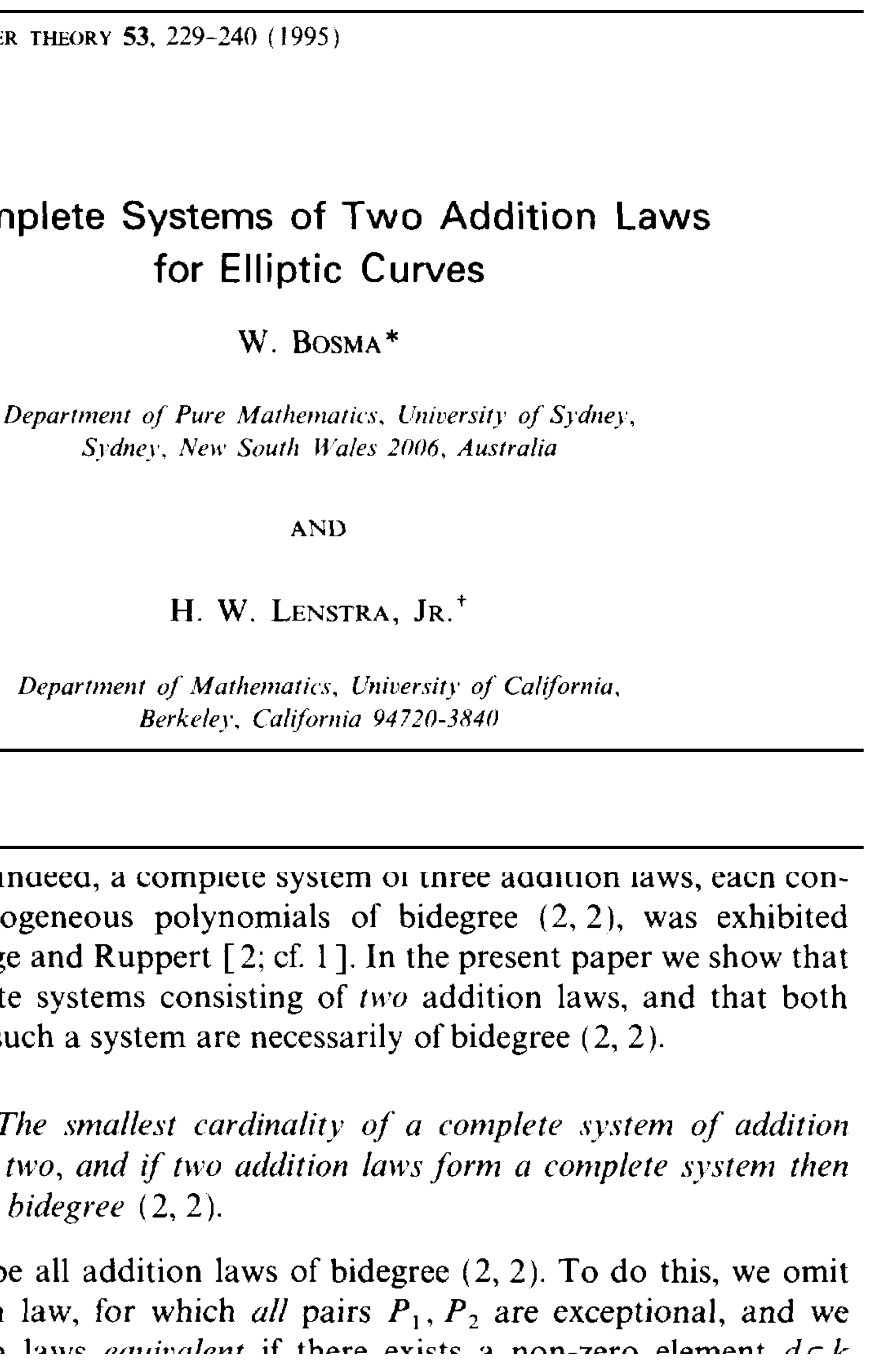


But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.


ER THEORY **53**, 229–240 (1995)

nplete Systems of Two Addition Laws
for Elliptic Curves

W. BOSMA*

Department of Pure Mathematics, University of Sydney,
Sydney, New South Wales 2006, Australia

AND

H. W. LENSTRA, JR.[†]

Department of Mathematics, University of California,
Berkeley, California 94720-3840

ndeed, a complete system of three addition laws, each con-
ogeneous polynomials of bidegree (2, 2), was exhibited
ge and Ruppert [2; cf. 1]. In the present paper we show that
te systems consisting of *two* addition laws, and that both
such a system are necessarily of bidegree (2, 2).

*The smallest cardinality of a complete system of addition two, and if two addition laws form a complete system then bidegree* (2, 2).

be all addition laws of bidegree (2, 2). To do this, we omit
law, for which *all* pairs $P_1$, $P_2$ are exceptional, and we



But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–
Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–
Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



“Guaran
load ent

5)

f Two Addition Laws
c Curves

OSMA*

*atics, University of Sydney,*
*Vales 2006, Australia*

D

STRA, JR.†

*s, University of California,*
*rnia 94720-3840*

tem of three addition laws, each con-
s of bidegree (2, 2), was exhibited
]. In the present paper we show that
of *two* addition laws, and that both
sarily of bidegree (2, 2).

*y of a complete system of addition*
*on laws form a complete system then*

bidegree (2, 2). To do this, we omit
airs $P_1$, $P_2$ are exceptional, and we



But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–
Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–
Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



"Guaranteed" cou
load entire table i


aws
aws, eacn con-
was exhibited
we show that
and that both
).
m of addition
e system then
this, we omit
ional, and we



But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–
Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–
Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



"Guaranteed" countermeasu
load entire table into cache.

But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



"Guaranteed" countermeasure:
load entire table into cache.

But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



"Guaranteed" countermeasure:
load entire table into cache.

2004.11/2005.04 Bernstein:
Timing attacks on AES.
Countermeasure isn't safe;
e.g., secret array indices can affect
timing via cache-bank collisions.
What *is* safe: kill all data flow
from secrets to array indices.

But wait, it's worse!

Crypto 1996 Kocher:
secret branches affect timing;
this leaks your secret key.

Briefly mentioned by Kocher
and by ESORICS 1998 Kelsey–Schneier–Wagner–Hall:
secret array indices can affect
timing via cache misses.

2002 Page, CHES 2003 Tsunoo–Saito–Suzaki–Shigeri–Miyauchi:
timing attacks on DES.



"Guaranteed" countermeasure:
load entire table into cache.

2004.11/2005.04 Bernstein:
Timing attacks on AES.
Countermeasure isn't safe;
e.g., secret array indices can affect
timing via cache-bank collisions.
What *is* safe: kill all data flow
from secrets to array indices.

2013 Bernstein–Schwabe
"A word of warning":
Cheaper countermeasure
recommended by Intel isn't safe.

t, it's worse!

1996 Kocher:
ranches affect timing;
s your secret key.

mentioned by Kocher
ESORICS 1998 Kelsey–
–Wagner–Hall:
rray indices can affect
ia cache misses.

ge, CHES 2003 Tsunoo–
uzaki–Shigeri–Miyauchi:
ttacks on DES.

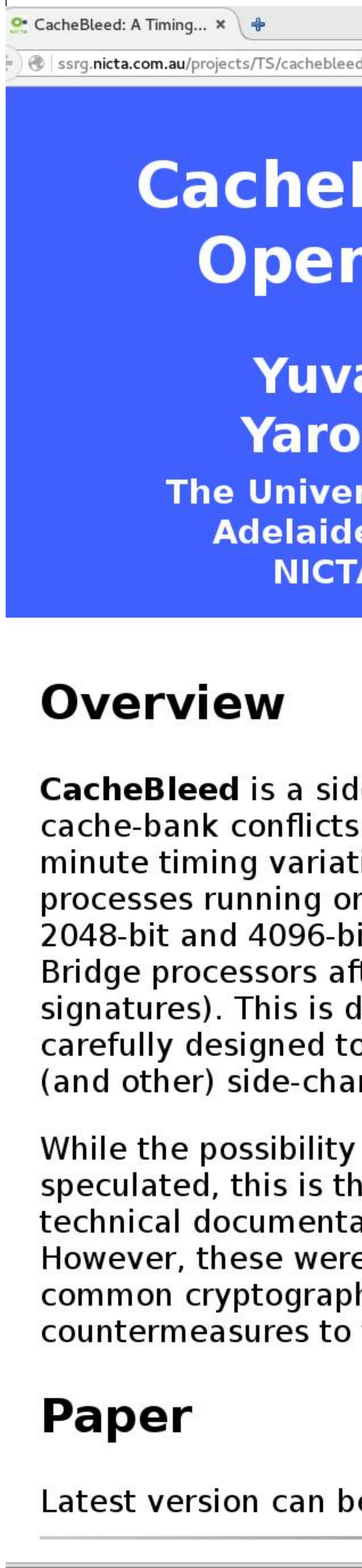


"Guaranteed" countermeasure:
load entire table into cache.

2004.11/2005.04 Bernstein:
Timing attacks on AES.
Countermeasure isn't safe;
e.g., secret array indices can affect
timing via cache-bank collisions.
What *is* safe: kill all data flow
from secrets to array indices.

2013 Bernstein–Schwabe
"A word of warning":
Cheaper countermeasure
recommended by Intel isn't safe.

2016: O



se!

er:
fect timing;
ret key.

by Kocher
1998 Kelsey–
Hall:
s can affect
nisses.

2003 Tsunoo–
eri–Miyauchi:
DES.

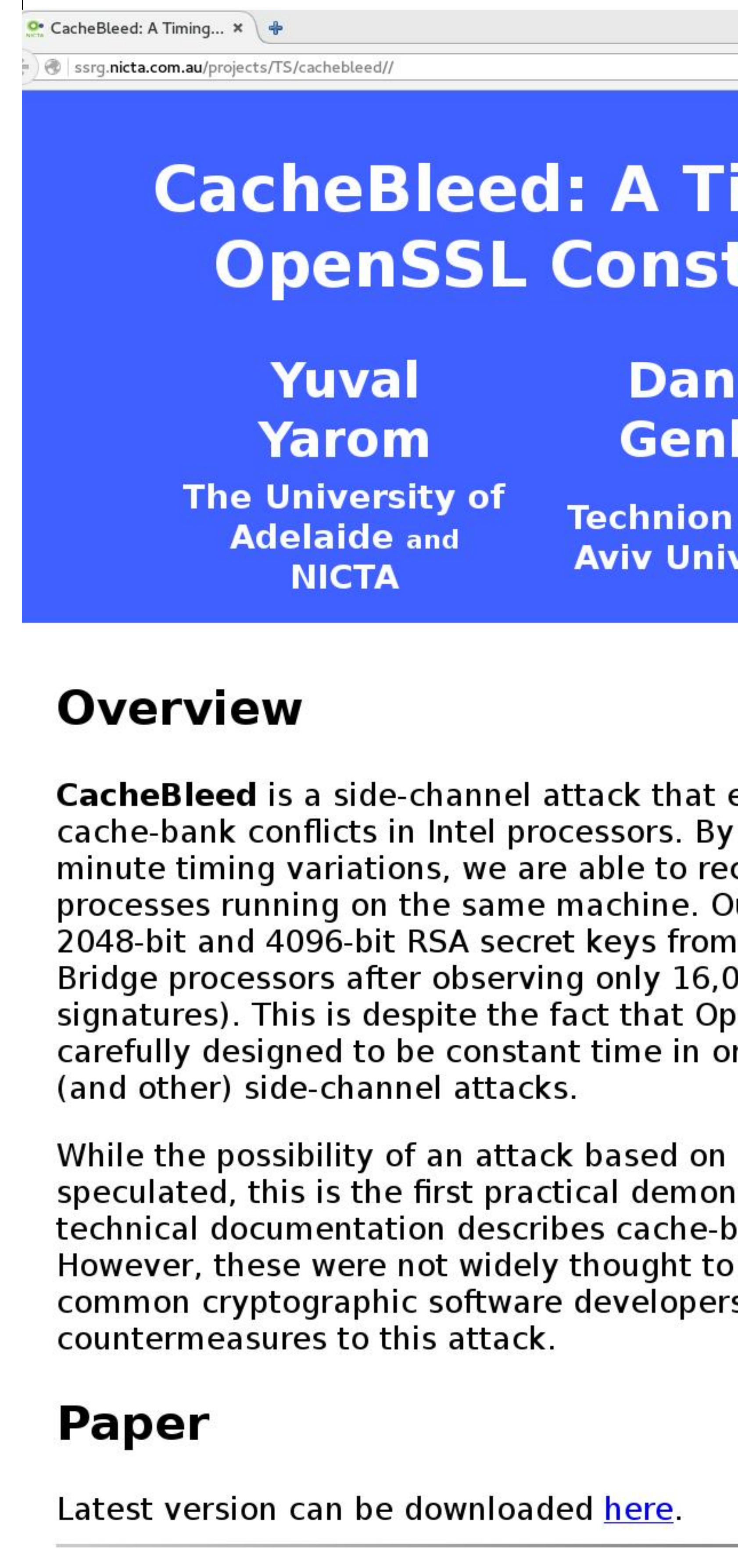


"Guaranteed" countermeasure:
load entire table into cache.

2004.11/2005.04 Bernstein:
Timing attacks on AES.
Countermeasure isn't safe;
e.g., secret array indices can affect
timing via cache-bank collisions.
What *is* safe: kill all data flow
from secrets to array indices.

2013 Bernstein–Schwabe
"A word of warning":
Cheaper countermeasure
recommended by Intel isn't safe.

2016: OpenSSL d



g;

r

ey–

ct

noo–

chi:



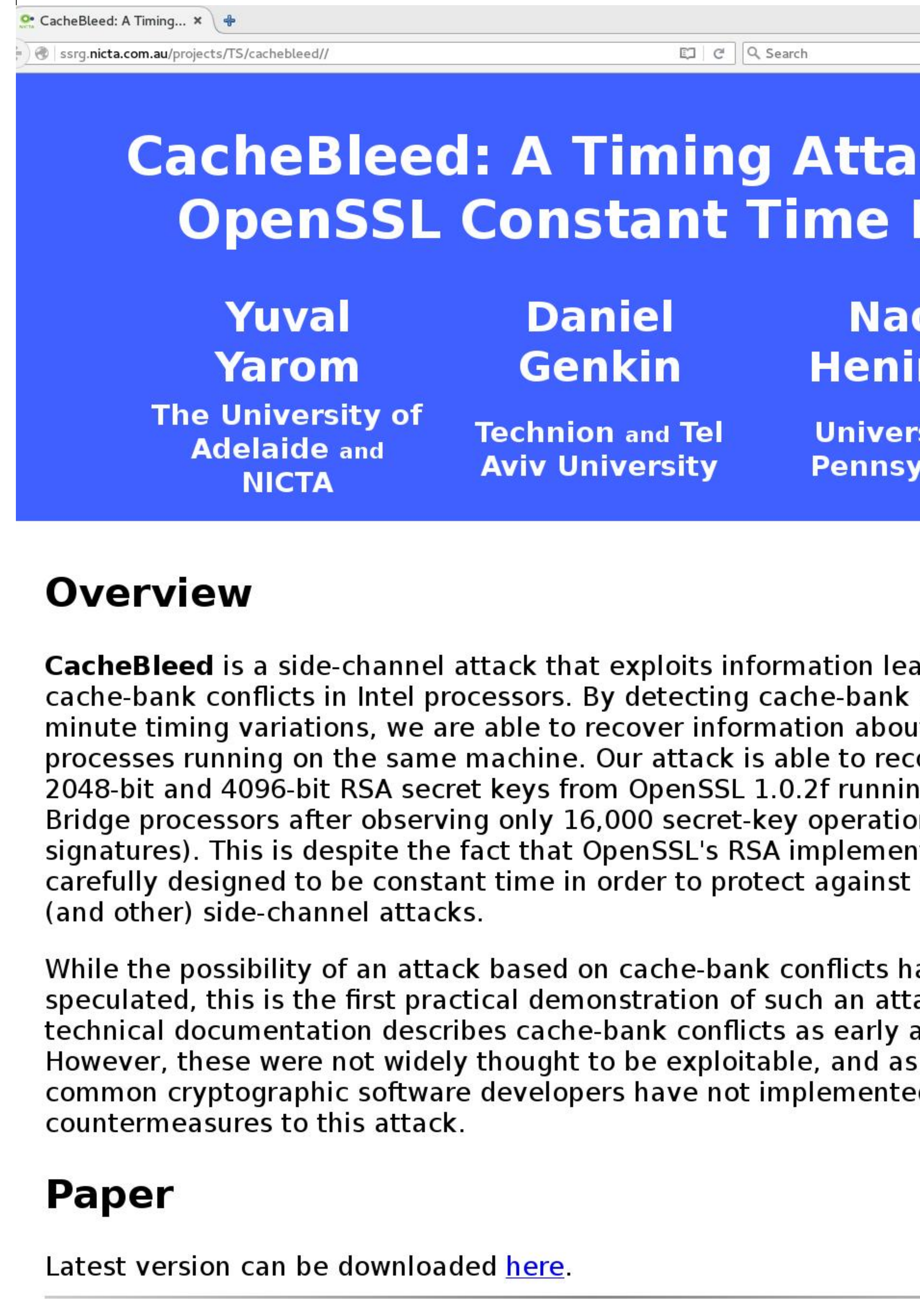
“Guaranteed” countermeasure: load entire table into cache.

2004.11/2005.04 Bernstein: Timing attacks on AES. Countermeasure isn’t safe; e.g., secret array indices can affect timing via cache-bank collisions. What *is* safe: kill all data flow from secrets to array indices.

2013 Bernstein–Schwabe “A word of warning”: Cheaper countermeasure recommended by Intel isn’t safe.



2016: OpenSSL didn’t liste



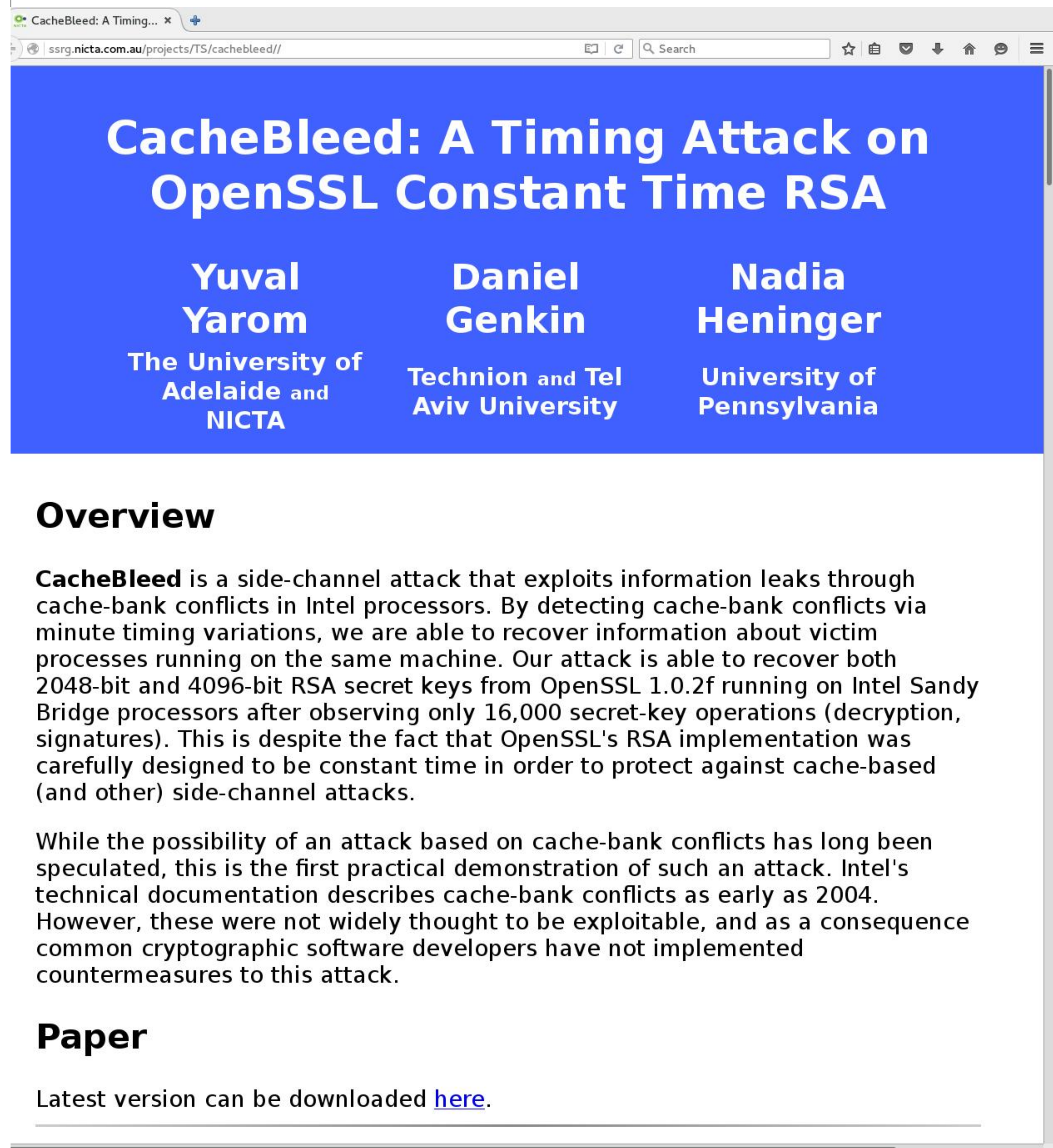
“Guaranteed” countermeasure: load entire table into cache.

2004.11/2005.04 Bernstein: Timing attacks on AES. Countermeasure isn’t safe; e.g., secret array indices can affect timing via cache-bank collisions. What *is* safe: kill all data flow from secrets to array indices.

2013 Bernstein–Schwabe “A word of warning”: Cheaper countermeasure recommended by Intel isn’t safe.



2016: OpenSSL didn’t listen.


CacheBleed: A Timing...

ssrg.nicta.com.au/projects/TS/cachebleed//

# CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom — The University of Adelaide and NICTA

Daniel Genkin — Technion and Tel Aviv University

Nadia Heninger — University of Pennsylvania

## Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

## Paper

Latest version can be downloaded here.

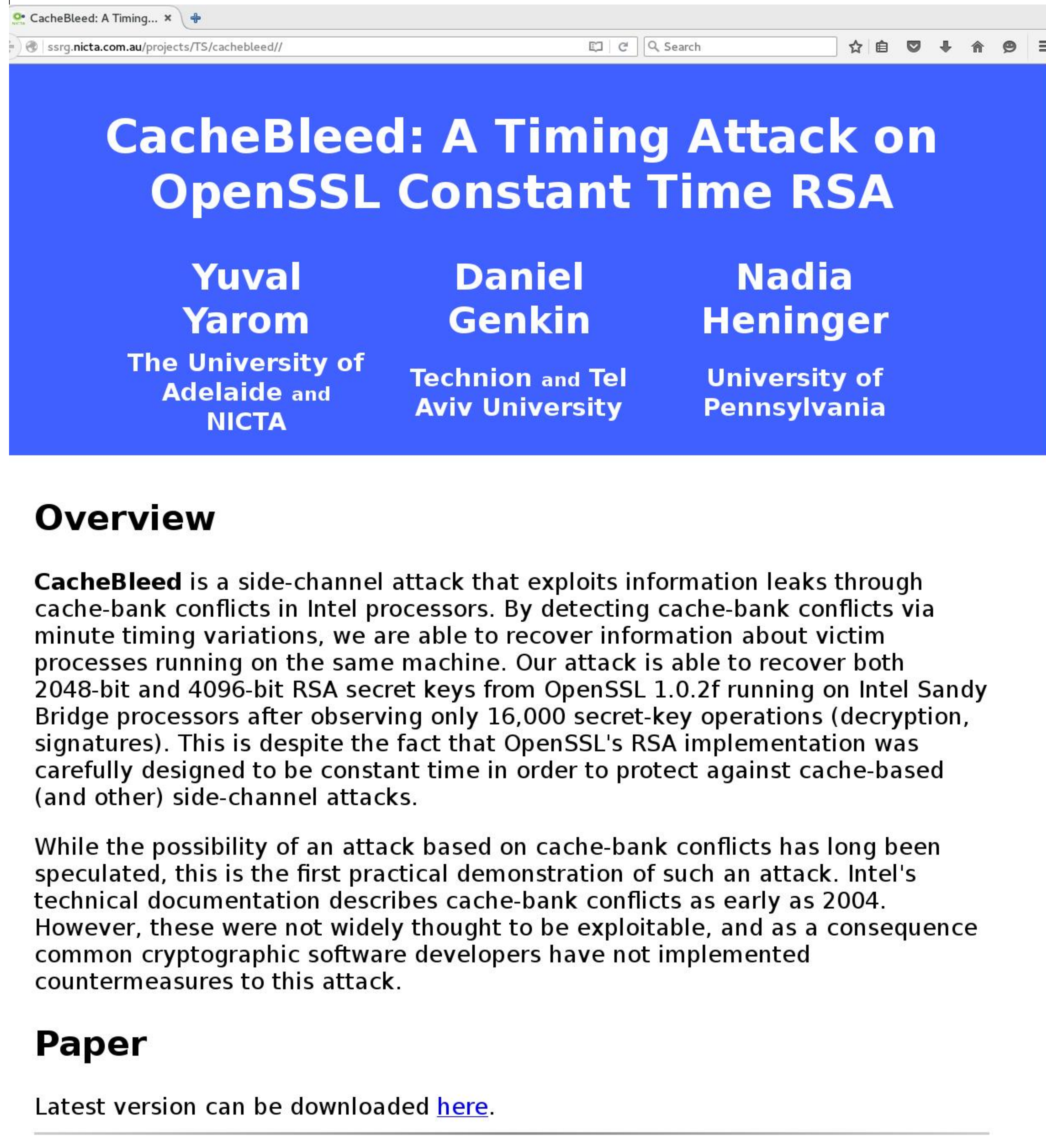
teed" countermeasure:
ire table into cache.

/2005.04 Bernstein:
attacks on AES.
measure isn't safe;
ret array indices can affect
ia cache-bank collisions.
safe: kill all data flow
crets to array indices.

rnstein–Schwabe
of warning":
countermeasure
ended by Intel isn't safe.



2016: OpenSSL didn't listen.




The Cur

Avoid "a
branches
indices,
with inp

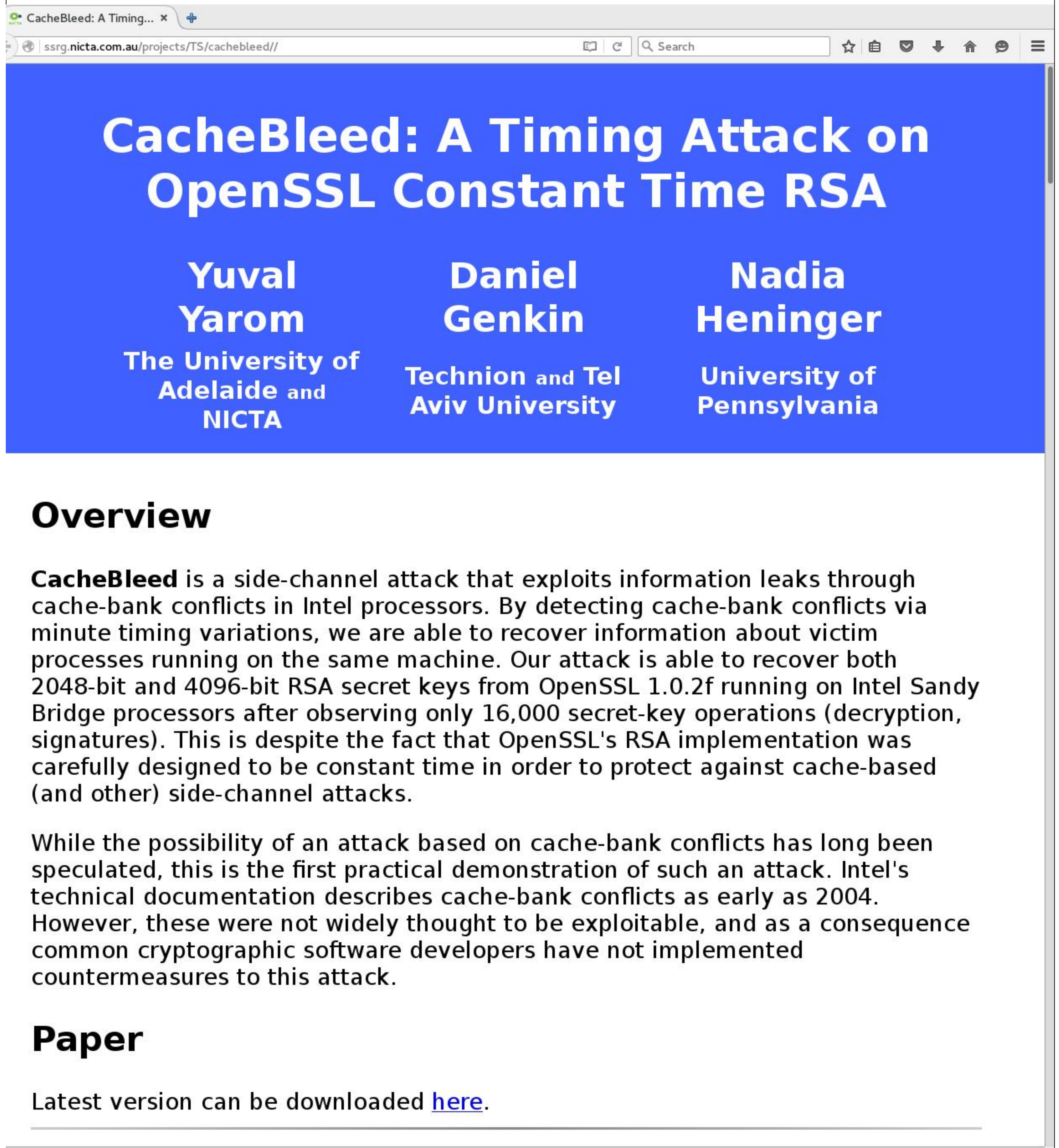
ntermeasure:
nto cache.

Bernstein:
AES.
sn't safe;
ndices can affect
bank collisions.
all data flow
ray indices.

chwabe
g":
easure
ntel isn't safe.



2016: OpenSSL didn't listen.




# CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom
The University of Adelaide and NICTA

Daniel Genkin
Technion and Tel Aviv University

Nadia Heninger
University of Pennsylvania

## Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

## Paper

Latest version can be downloaded here.

The Curve25519 p

Avoid "all input-d
branches, all input
indices, and other
with input-depend

re:

affect

ons.

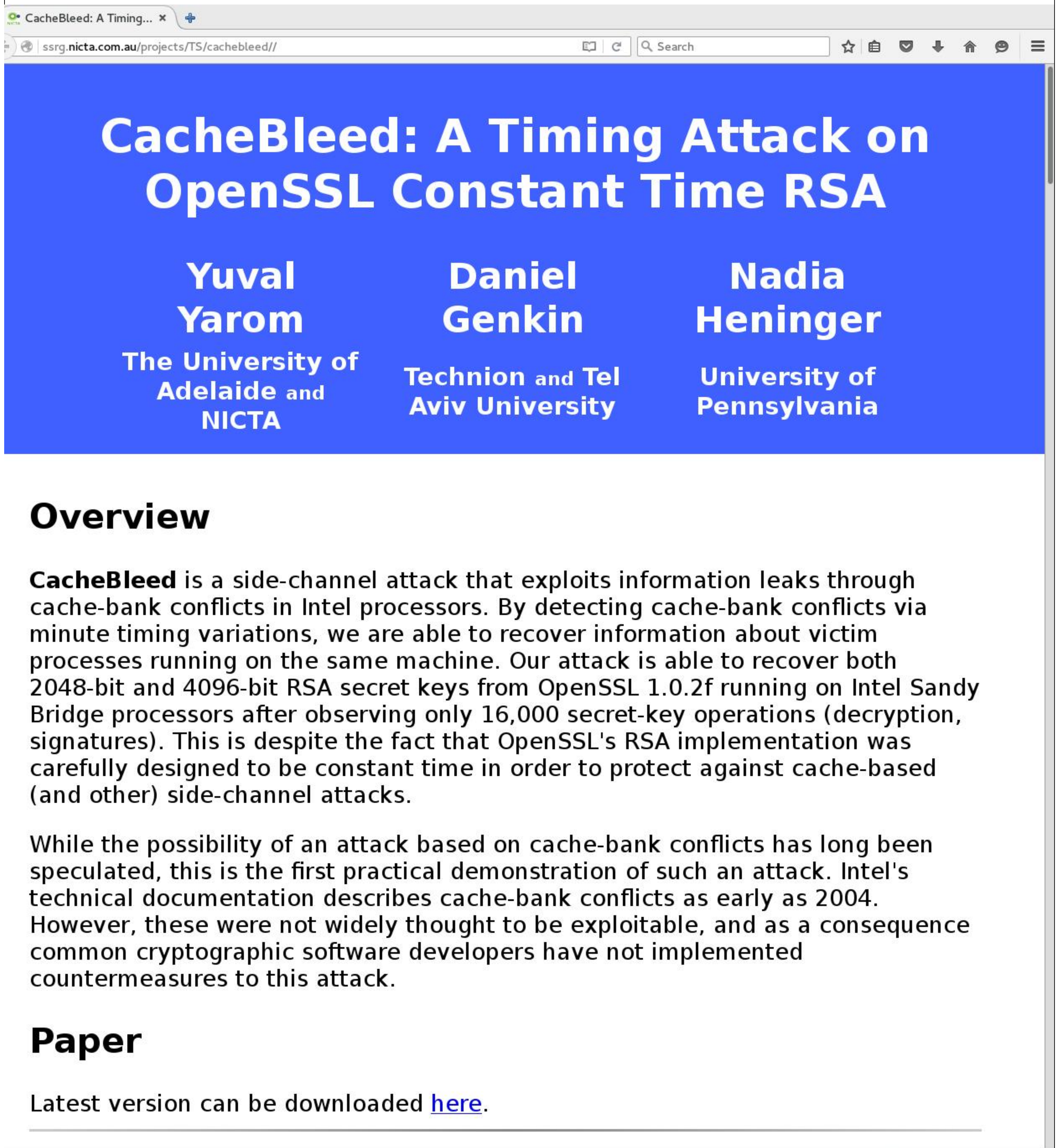
ow

.

safe.



# 2016: OpenSSL didn’t listen.




**CacheBleed: A Timing Attack on OpenSSL Constant Time RSA**

**Yuval Yarom** — The University of Adelaide and NICTA
**Daniel Genkin** — Technion and Tel Aviv University
**Nadia Heninger** — University of Pennsylvania

**Overview**

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

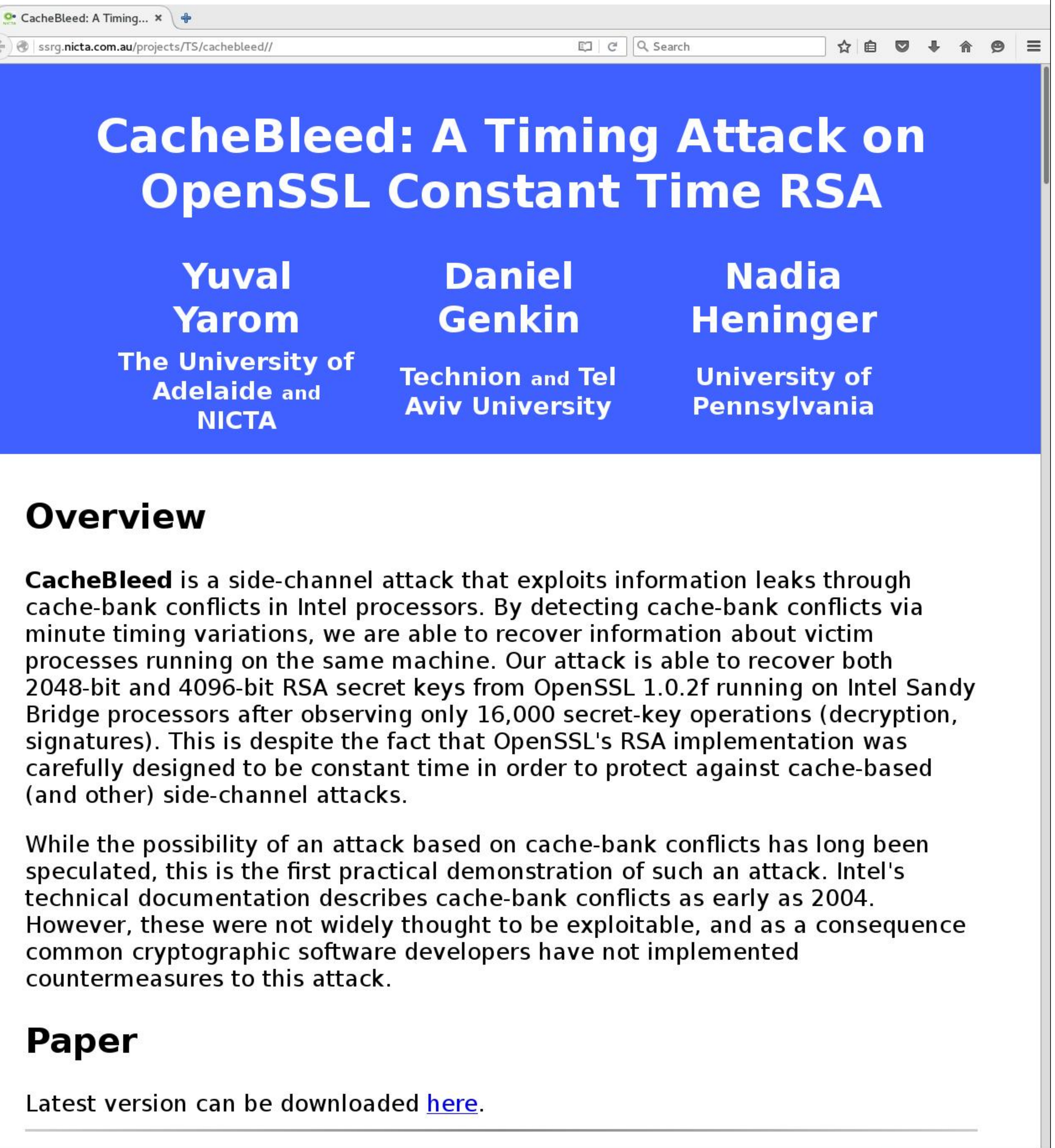
**Paper**

Latest version can be downloaded here.

# The Curve25519 paper

Avoid “all input-dependent
branches, all input-depender
indices, and other instructio
with input-dependent timing

# 2016: OpenSSL didn’t listen.

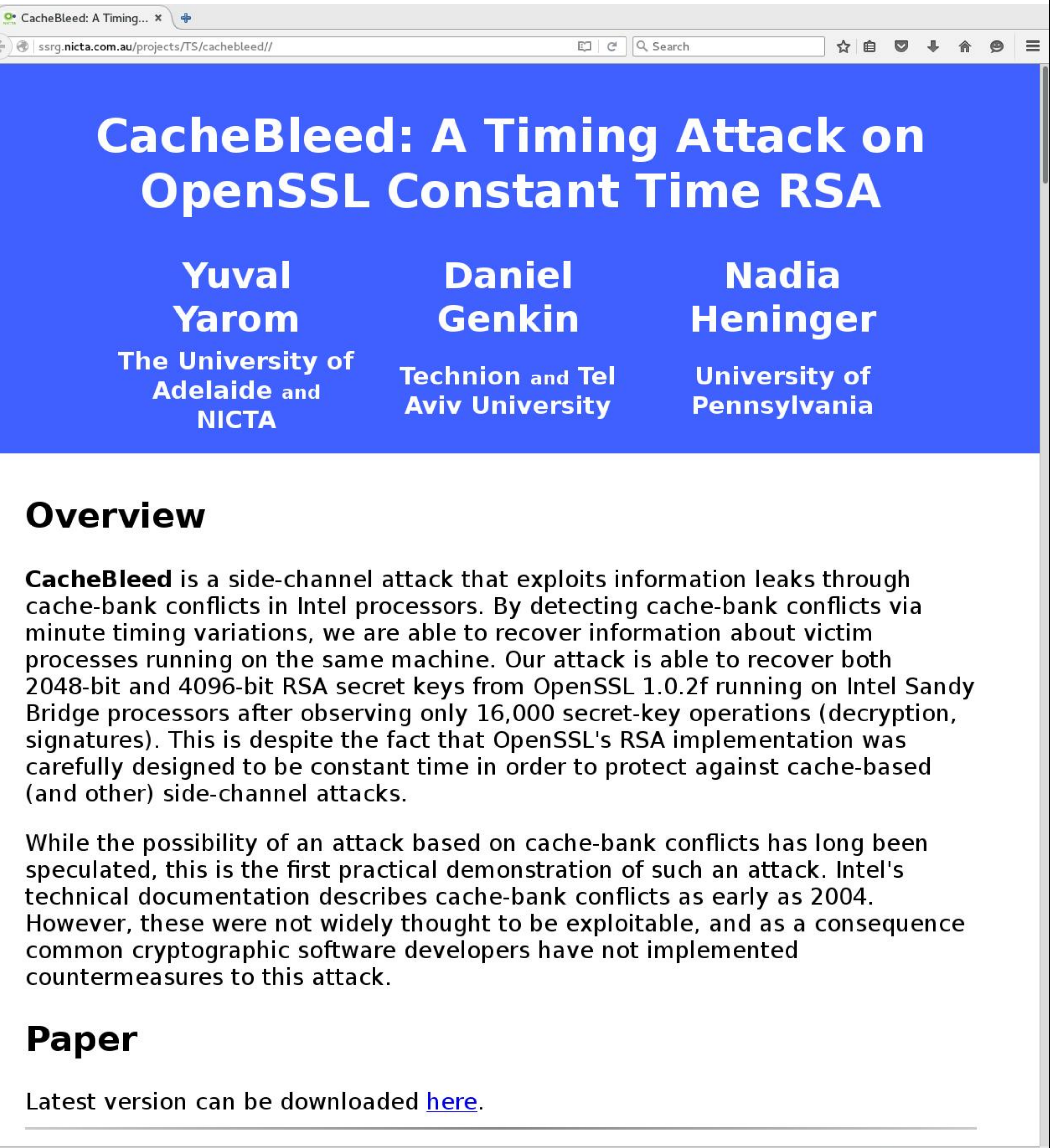





# The Curve25519 paper

Avoid “all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings”.

# 2016: OpenSSL didn't listen.



CacheBleed: A Timing...
ssrg.nicta.com.au/projects/TS/cachebleed//

## CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom — The University of Adelaide and NICTA
Daniel Genkin — Technion and Tel Aviv University
Nadia Heninger — University of Pennsylvania

### Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

### Paper

Latest version can be downloaded here.



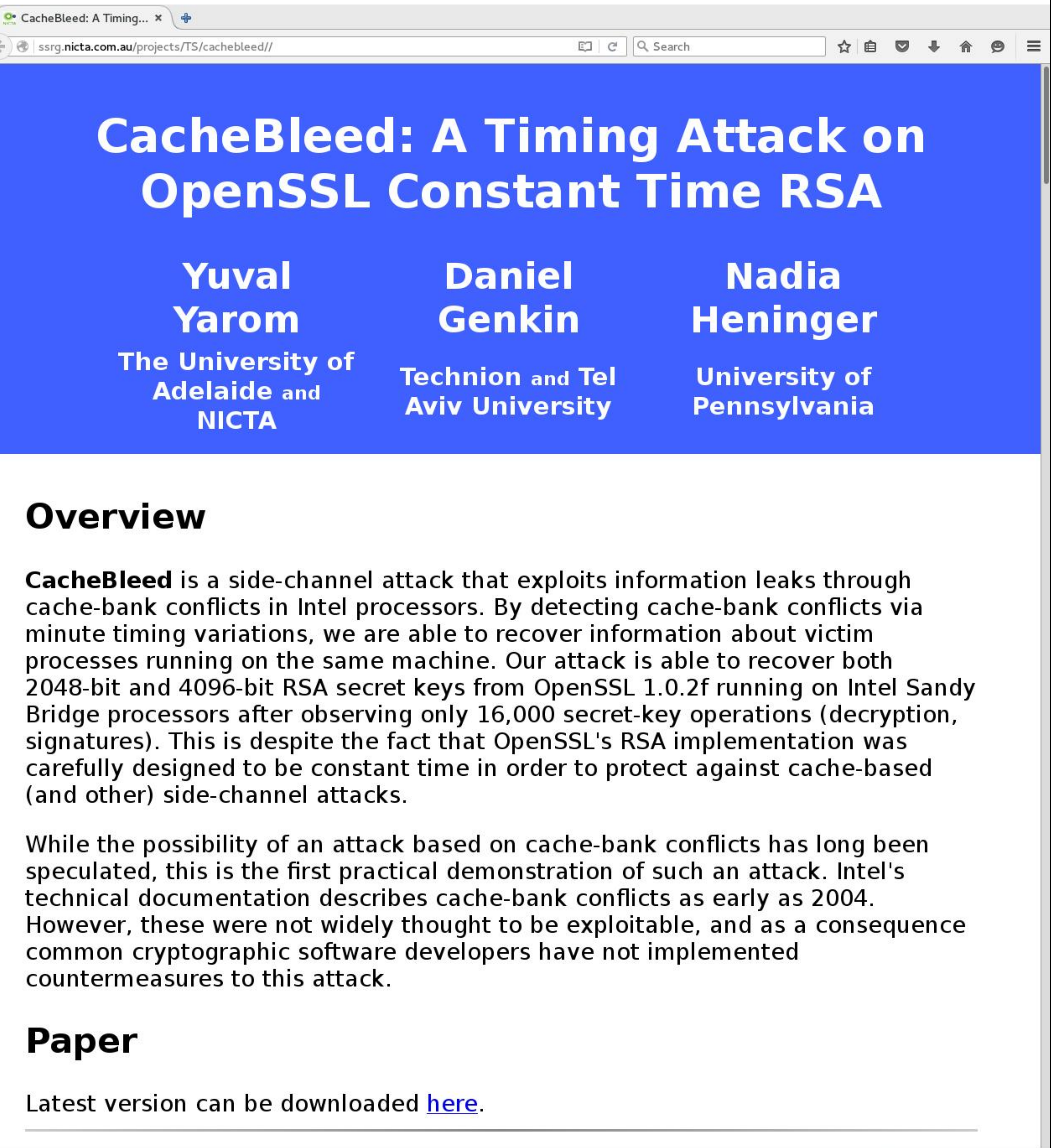
# The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square.

$\approx$25% of all elliptic curves.

# 2016: OpenSSL didn’t listen.



CacheBleed: A Timing... ×

ssrg.nicta.com.au/projects/TS/cachebleed//

## CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom — The University of Adelaide and NICTA

Daniel Genkin — Technion and Tel Aviv University

Nadia Heninger — University of Pennsylvania

### Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

### Paper

Latest version can be downloaded here.



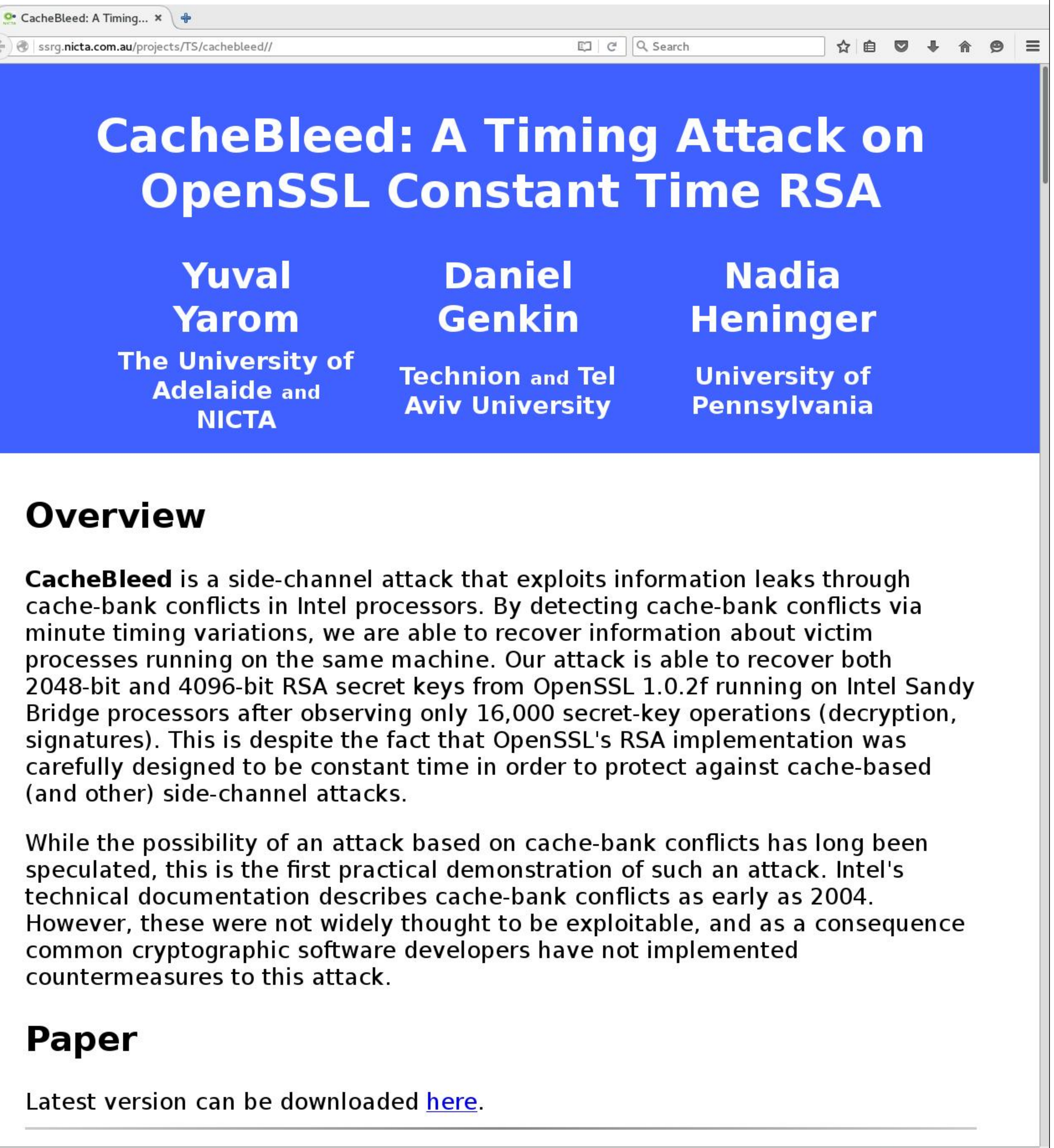
# The Curve25519 paper

Avoid “all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings”.

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

# 2016: OpenSSL didn't listen.




## CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom
The University of Adelaide and NICTA

Daniel Genkin
Technion and Tel Aviv University

Nadia Heninger
University of Pennsylvania

### Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

### Paper

Latest version can be downloaded here.



# The Curve25519 paper

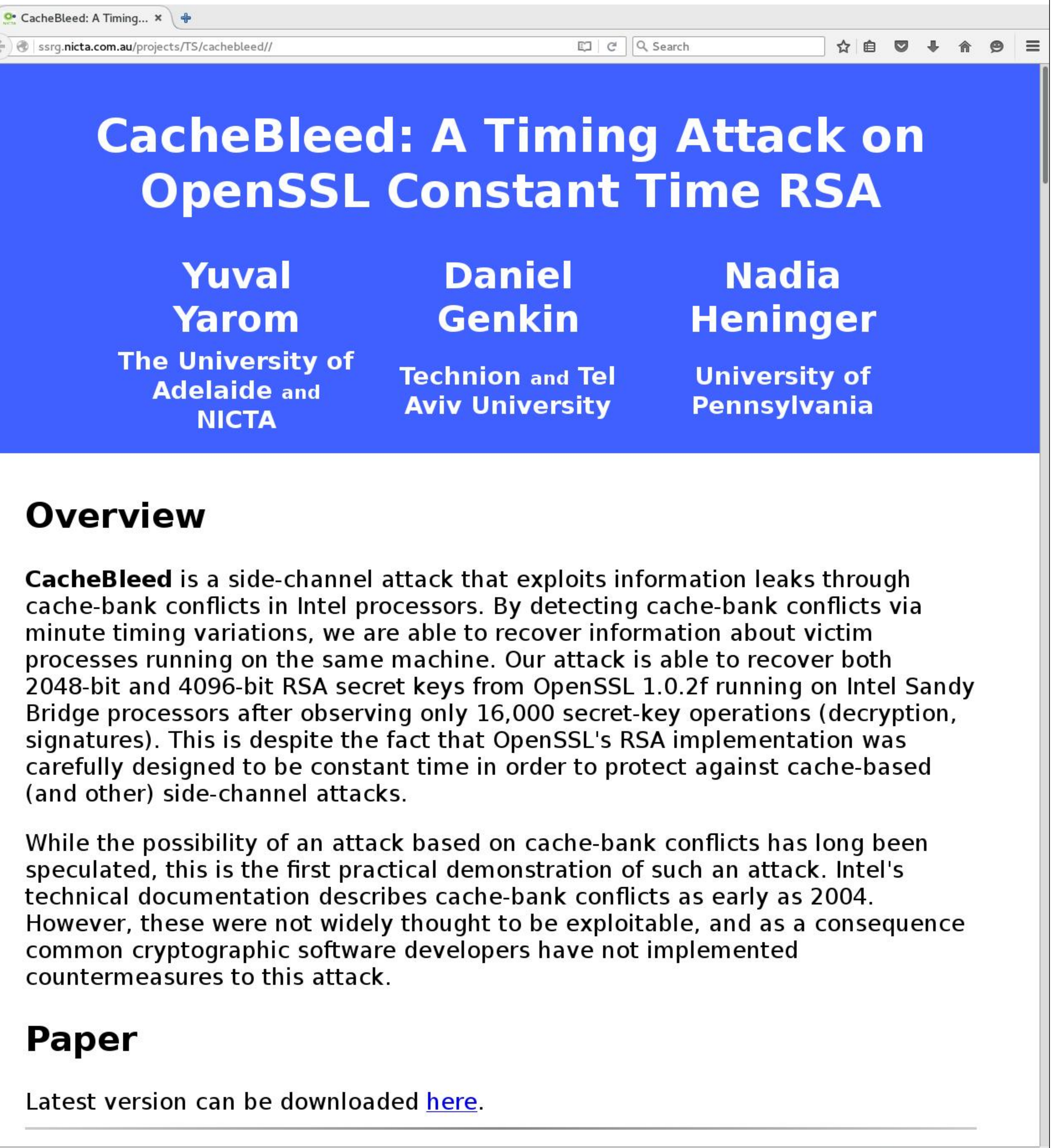
Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. ≈25% of all elliptic curves.

Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

# 2016: OpenSSL didn't listen.




## CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

Yuval Yarom
The University of Adelaide and NICTA

Daniel Genkin
Technion and Tel Aviv University

Nadia Heninger
University of Pennsylvania

### Overview

**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover both 2048-bit and 4096-bit RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors after observing only 16,000 secret-key operations (decryption, signatures). This is despite the fact that OpenSSL's RSA implementation was carefully designed to be constant time in order to protect against cache-based (and other) side-channel attacks.

While the possibility of an attack based on cache-bank conflicts has long been speculated, this is the first practical demonstration of such an attack. Intel's technical documentation describes cache-bank conflicts as early as 2004. However, these were not widely thought to be exploitable, and as a consequence common cryptographic software developers have not implemented countermeasures to this attack.

### Paper

Latest version can be downloaded here.



# The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

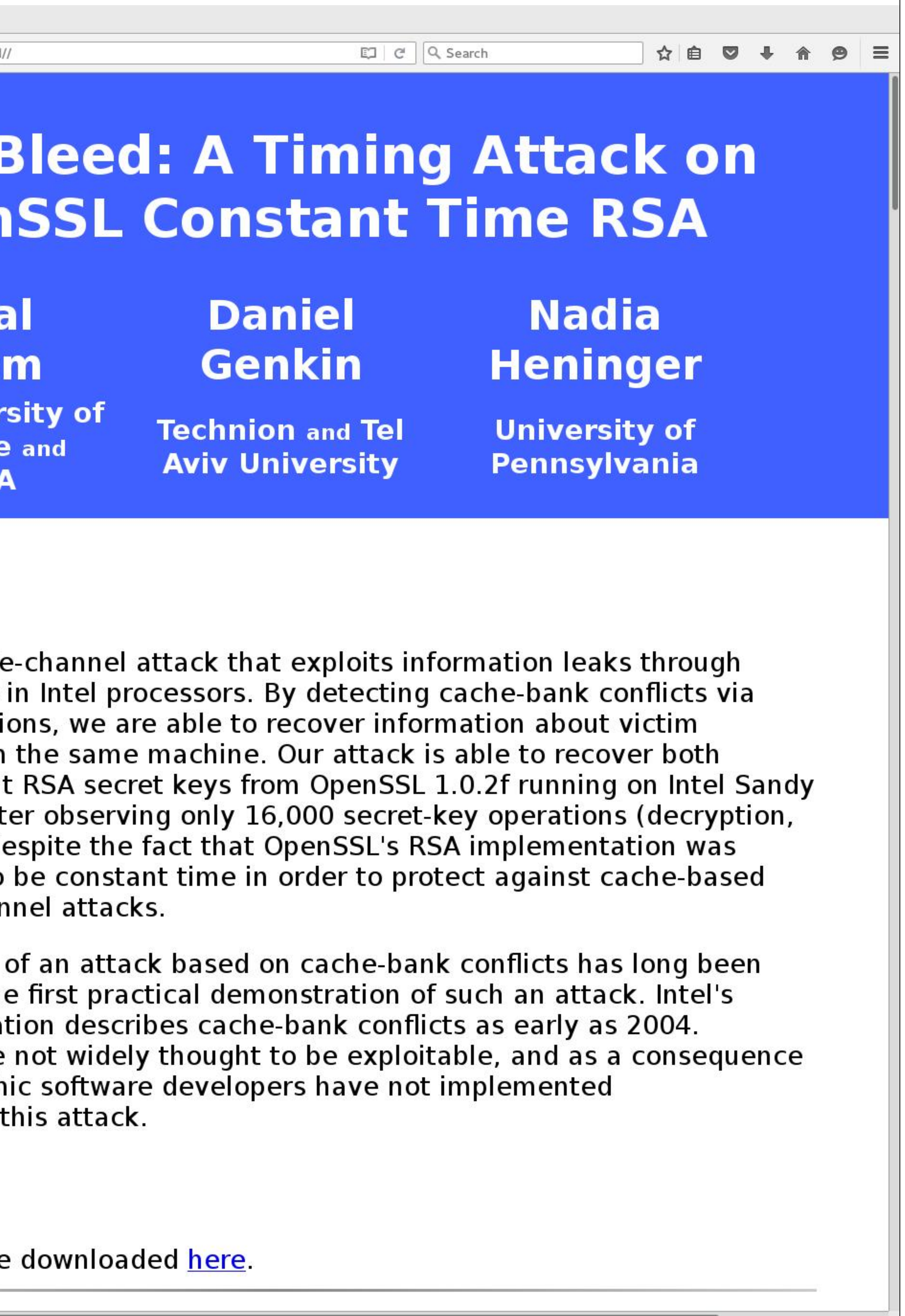
Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

Theorem: Output is $X_0(nP)$.

penSSL didn't listen.






The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

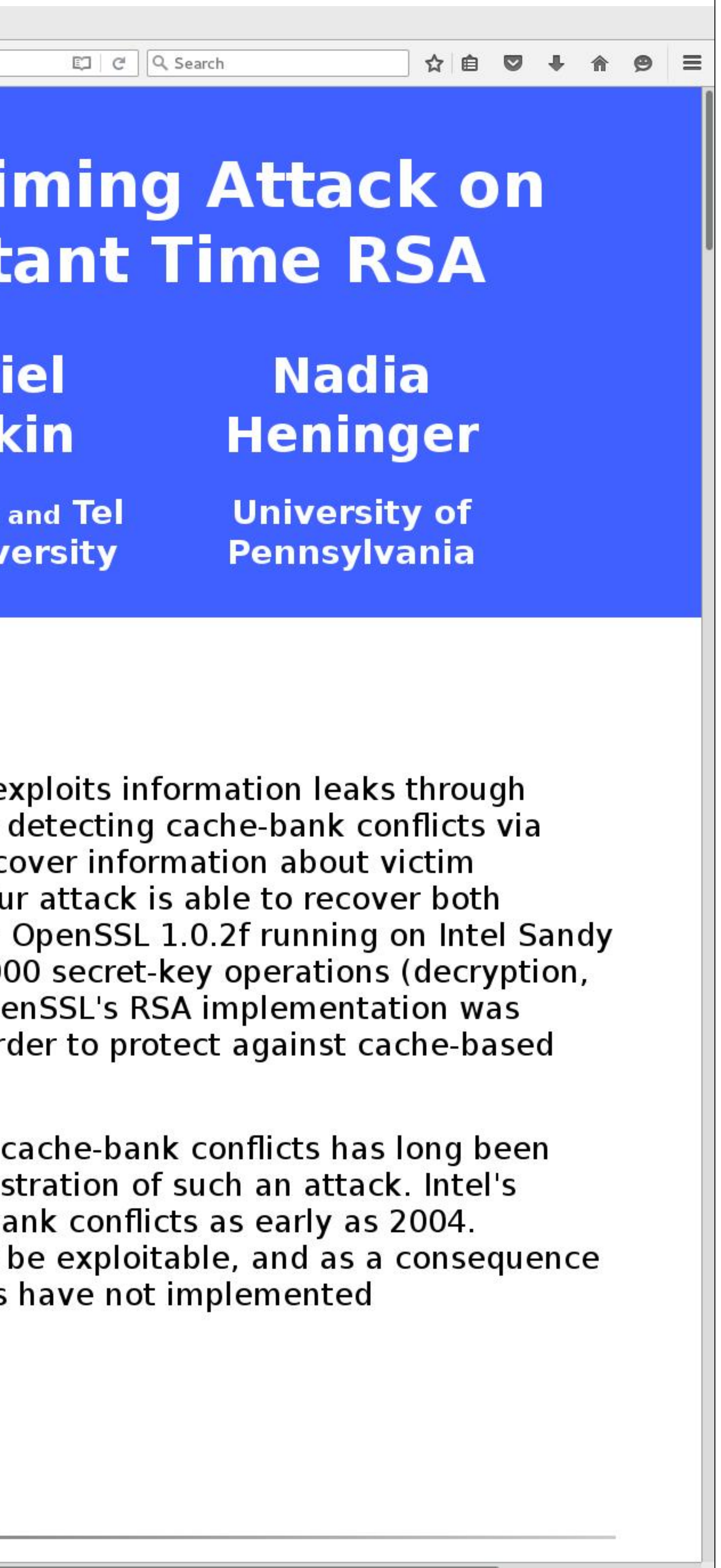
Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

Theorem: Output is $X_0(nP)$.

```
x2,z2,x3
for i in
  bit =
  x2,x3
  z2,z3
  x3,z3


  x2,z2
    4*x2
  x2,x3
  z2,z3
return x
```

idn't listen.






## The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

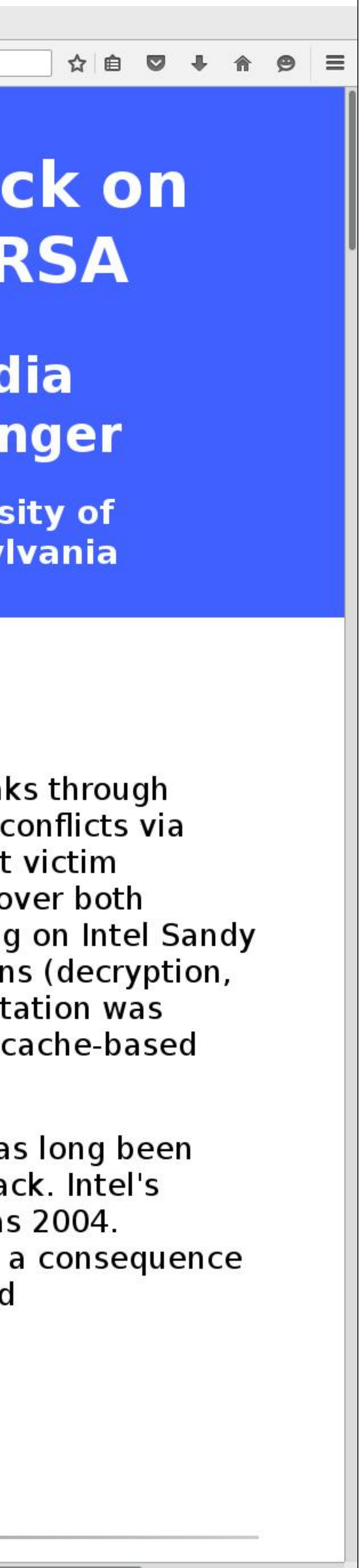
Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

Theorem: Output is $X_0(nP)$.

```
x2,z2,x3,z3 = 1,
for i in reverse
  bit = 1 & (n >
  x2,x3 = cswap(
  z2,z3 = cswap(
  x3,z3 = ((x2*x
        x1*(x2*z
  x2,z2 = ((x2^2
    4*x2*z2*(x2^
  x2,x3 = cswap(
  z2,z3 = cswap(
return x2*z2^(p-
```

n.





## The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

Theorem: Output is $X_0(nP)$.

```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(2
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit
  z2,z3 = cswap(z2,z3,bit
  x3,z3 = ((x2*x3-z2*z3)^
        x1*(x2*z3-z2*x3)^
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2
  x2,x3 = cswap(x2,x3,bit
  z2,z3 = cswap(z2,z3,bit
return x2*z2^(p-2)
```

# The Curve25519 paper

Avoid "all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings".

Choose a curve $y^2 = x^3 + Ax^2 + x$ where $A^2 - 4$ is not a square. $\approx$25% of all elliptic curves.

Define $X_0(x, y) = x$; $X_0(\infty) = 0$. Transmit each point $P$ as $X_0(P)$.

Use the Montgomery ladder **without any extra tests**.

Theorem: Output is $X_0(nP)$.



```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
           x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```

ve25519 paper

all input-dependent
s, all input-dependent array
and other instructions
ut-dependent timings”.

a curve $y^2 = x^3 + Ax^2 + x$
$^2 - 4$ is not a square.
f all elliptic curves.

$X_0(x, y) = x$; $X_0(\infty) = 0$.
t each point $P$ as $X_0(P)$.

Montgomery ladder
**any extra tests**.

n: Output is $X_0(nP)$.



```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
              x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```

Montgo
dependi

paper

ependent
-dependent array
instructions
ent timings”.

$^2 = x^3 + Ax^2 + x$
ot a square.
c curves.

$x$; $X_0(\infty) = 0$.
nt $P$ as $X_0(P)$.

ery ladder
**a tests**.

is $X_0(nP)$.



```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
              x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```

Montgomery has v
depending on top

nt array
ns
gs".
$x^2 + x$
e.

$) = 0.$
$_0(P).$

$).$



```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
        x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```

Montgomery has variable #
depending on top bit of $n$.

```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
        x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```



Montgomery has variable #loops, depending on top bit of $n$.

```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
        x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```



Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
        x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```



Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits. Use constant #loops.

Also define scalars $n$ to never have leading 0 bits, so original Montgomery ladder still takes constant time.

```
x2,z2,x3,z3 = 1,0,x1,1
for i in reversed(range(255)):
  bit = 1 & (n >> i)
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
  x3,z3 = ((x2*x3-z2*z3)^2,
        x1*(x2*z3-z2*x3)^2)
  x2,z2 = ((x2^2-z2^2)^2,
    4*x2*z2*(x2^2+A*x2*z2+z2^2))
  x2,x3 = cswap(x2,x3,bit)
  z2,z3 = cswap(z2,z3,bit)
return x2*z2^(p-2)
```



Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits. Use constant #loops.

Also define scalars $n$ to never have leading 0 bits, so original Montgomery ladder still takes constant time.

Use arithmetic to compute `cswap` in constant time.

```
3,z3 = 1,0,x1,1
n reversed(range(255)):
 1 & (n >> i)
 = cswap(x2,x3,bit)
 = cswap(z2,z3,bit)
 = ((x2*x3-z2*z3)^2,
 x1*(x2*z3-z2*x3)^2)
 = ((x2^2-z2^2)^2,
2*z2*(x2^2+A*x2*z2+z2^2))
 = cswap(x2,x3,bit)
 = cswap(z2,z3,bit)
x2*z2^(p-2)
```



Montgomery has variable #loops, depending on top bit of *n*.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

Also define scalars *n* to never have leading 0 bits, so original Montgomery ladder still takes constant time.

Use arithmetic to compute `cswap` in constant time.

"Hey, yo
the inpu

```
0,x1,1
d(range(255)):
> i)
x2,x3,bit)
z2,z3,bit)
3-z2*z3)^2,
3-z2*x3)^2)
-z2^2)^2,
2+A*x2*z2+z2^2))
x2,x3,bit)
z2,z3,bit)
2)
```



Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

Also define scalars $n$
to never have leading 0 bits,
so original Montgomery ladder
still takes constant time.

Use arithmetic to compute
`cswap` in constant time.

“Hey, you forgot t
the input is on the

```
255)):

)

)

2,

2)

2+z2^2))

)

)
```



Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

Also define scalars $n$ to never have leading 0 bits, so original Montgomery ladder still takes constant time.

Use arithmetic to compute `cswap` in constant time.

"Hey, you forgot to check th the input is on the curve!"

Montgomery has variable #loops,
depending on top bit of $n$.

Curve25519: Change initialization
to allow leading 0 bits.
Use constant #loops.

Also define scalars $n$
to never have leading 0 bits,
so original Montgomery ladder
still takes constant time.

Use arithmetic to compute
`cswap` in constant time.



"Hey, you forgot to check that
the input is on the curve!"

Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

Also define scalars $n$ to never have leading 0 bits, so original Montgomery ladder still takes constant time.

Use arithmetic to compute `cswap` in constant time.



"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

Montgomery has variable #loops, depending on top bit of $n$.

Curve25519: Change initialization to allow leading 0 bits.
Use constant #loops.

Also define scalars $n$ to never have leading 0 bits, so original Montgomery ladder still takes constant time.

Use arithmetic to compute `cswap` in constant time.



"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.

mery has variable #loops,
ng on top bit of $n$.

519: Change initialization
leading 0 bits.
stant #loops.

ine scalars $n$
have leading 0 bits,
al Montgomery ladder
es constant time.

hmetic to compute
n constant time.



"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.


Curve25
"free key
eliminate
No cost
no code

variable #loops,
bit of $n$.

nge initialization
bits.
ops.

s $n$
ing 0 bits,
omery ladder
t time.

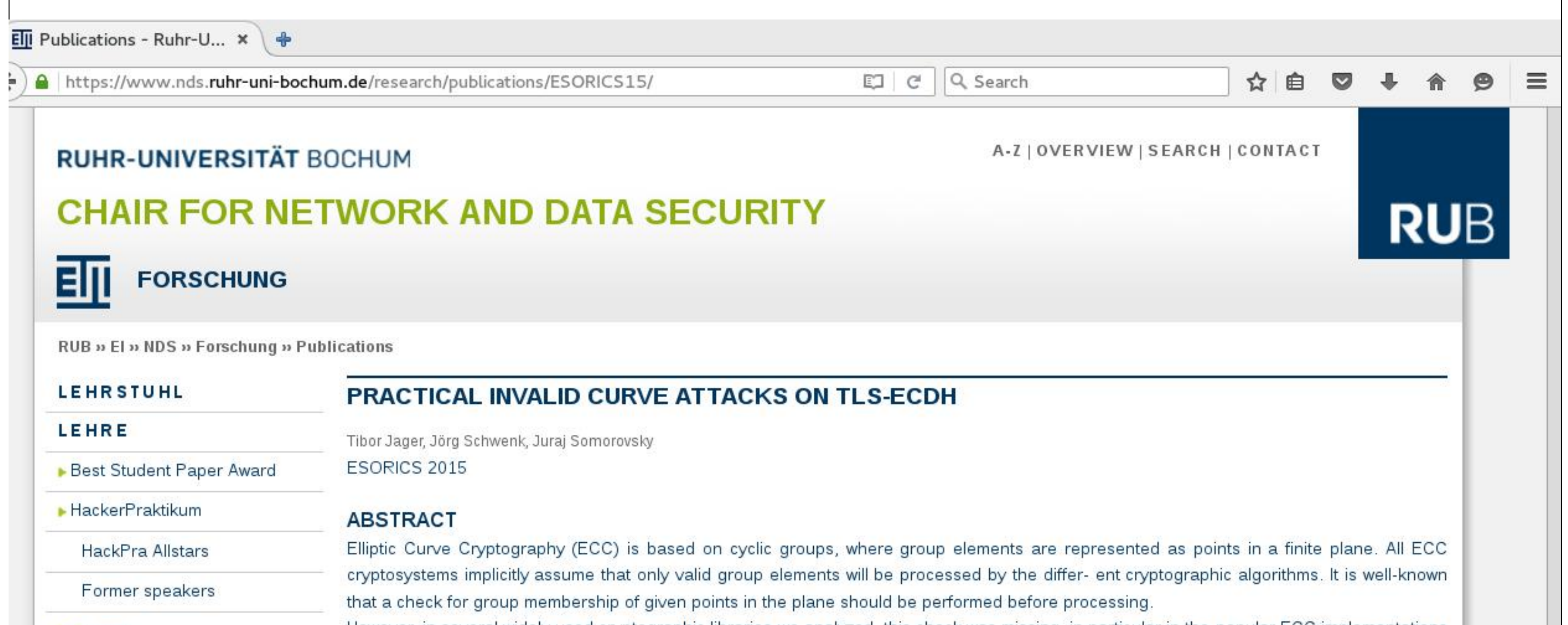
compute
time.



"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.




Curve25519 paper
"free key validatio
eliminates these at
No cost for checki
no code to forget.

loops,

zation

,

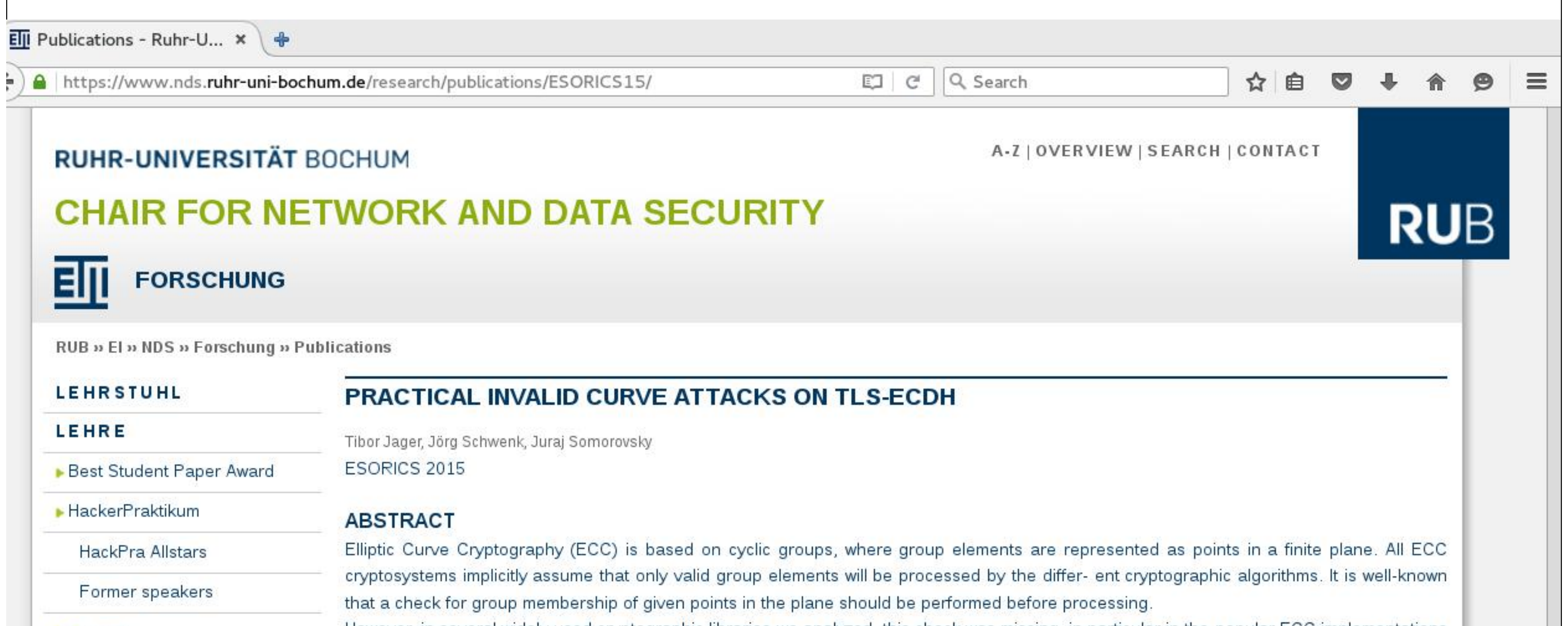
der



"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.




Curve25519 paper: "free key validation" eliminates these attacks. No cost for checking input; no code to forget.

"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.




Curve25519 paper: "free key validation" eliminates these attacks. No cost for checking input; no code to forget.

"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.



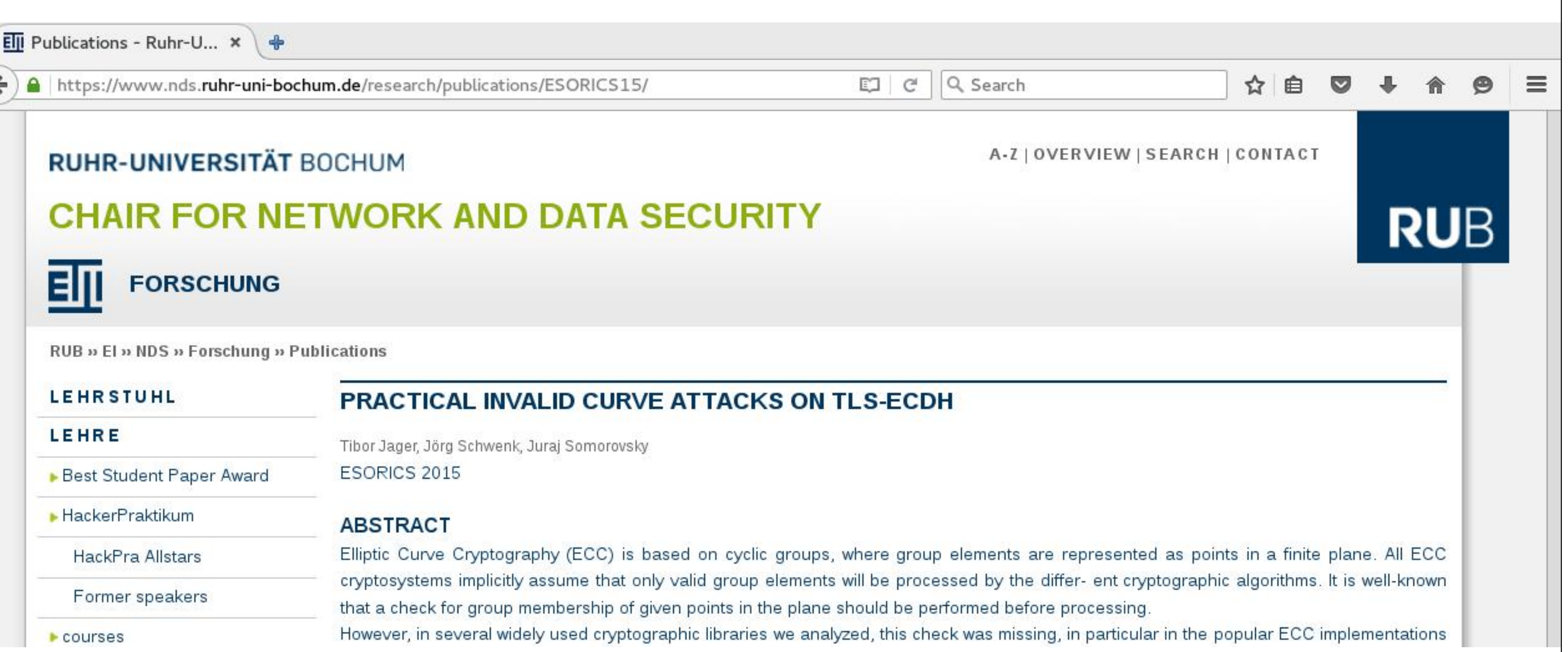

Curve25519 paper: "free key validation" eliminates these attacks. No cost for checking input; no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.


Publications - Ruhr-U...
https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/
Search
RUHR-UNIVERSITÄT BOCHUM
A-Z | OVERVIEW | SEARCH | CONTACT
CHAIR FOR NETWORK AND DATA SECURITY
RUB
FORSCHUNG
RUB » EI » NDS » Forschung » Publications
LEHRSTUHL
LEHRE
Best Student Paper Award
HackerPraktikum
HackPra Allstars
Former speakers
courses

PRACTICAL INVALID CURVE ATTACKS ON TLS-ECDH

Tibor Jager, Jörg Schwenk, Juraj Somorovsky

ESORICS 2015

ABSTRACT

Elliptic Curve Cryptography (ECC) is based on cyclic groups, where group elements are represented as points in a finite plane. All ECC cryptosystems implicitly assume that only valid group elements will be processed by the differ- ent cryptographic algorithms. It is well-known that a check for group membership of given points in the plane should be performed before processing.

However, in several widely used cryptographic libraries we analyzed, this check was missing, in particular in the popular ECC implementations



Curve25519 paper: "free key validation" eliminates these attacks. No cost for checking input; no code to forget.

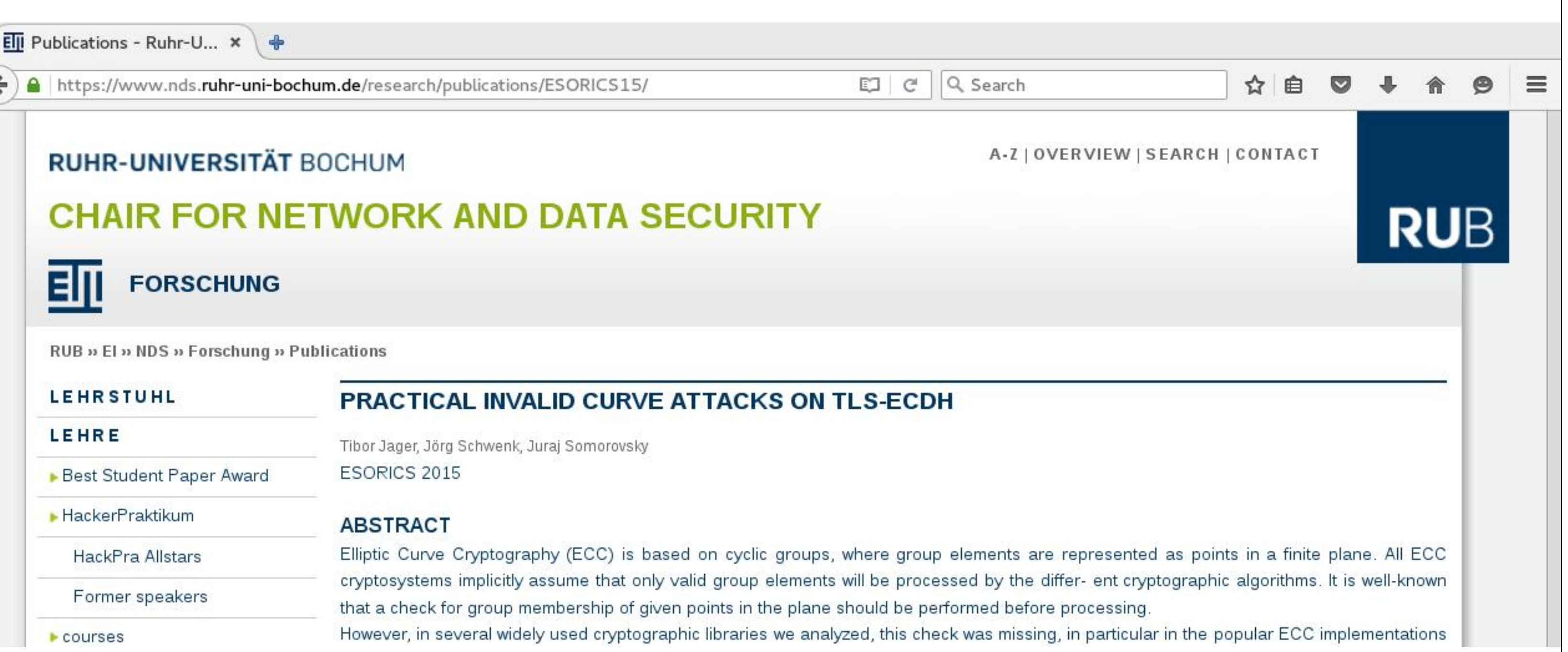
1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

"Hey, you forgot to check that the input is on the curve!"

Conventional wisdom: Important to check; otherwise broken by Crypto 2000 Biehl–Meyer–Müller.

ESORICS 2015 Jager–Schwenk–Somorovsky: Successful attacks! Checking is easy to forget.






Curve25519 paper: "free key validation" eliminates these attacks. No cost for checking input; no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

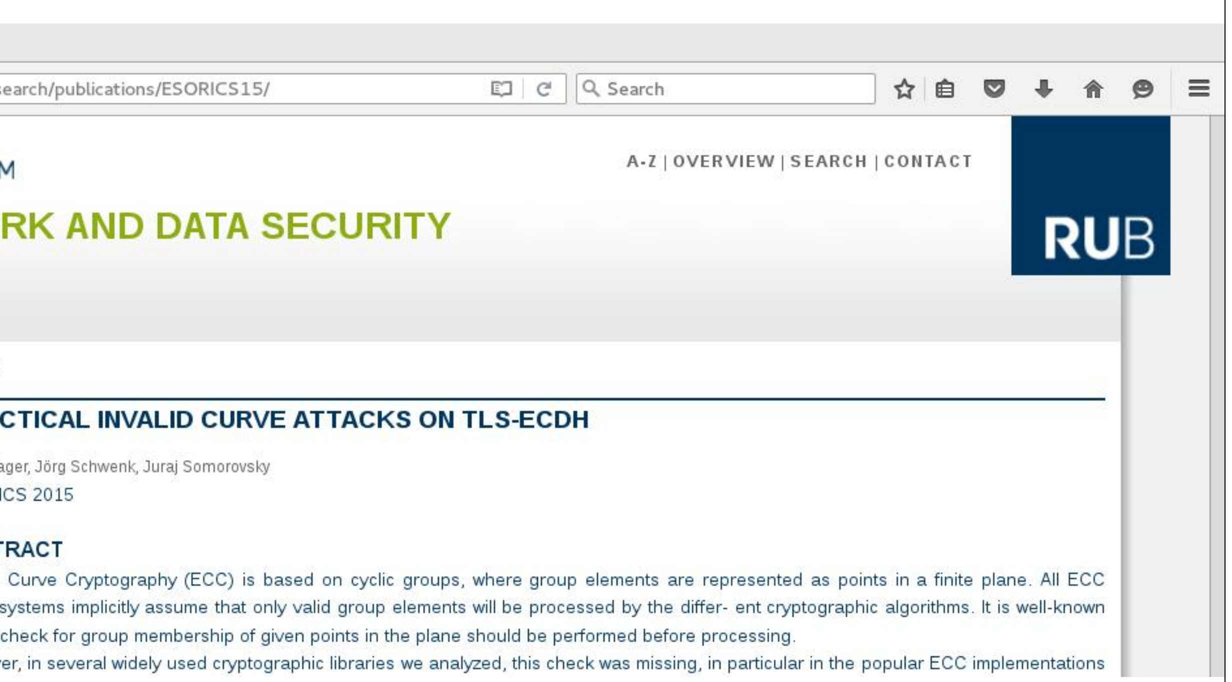
2. Montgomery ladder works correctly for inputs on twist.

3. Choose twist-secure curve.

ou forgot to check that
t is on the curve!"

ional wisdom: Important
; otherwise broken by
2000 Biehl–Meyer–Müller.

S 2015 Jager–Schwenk–
vsky: Successful attacks!
g is easy to forget.






Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally
follows 1986 Miller compression:
send only $x$-coordinate, not $(x, y)$.
Forces input onto "curve" or
"twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works
correctly for inputs on twist.

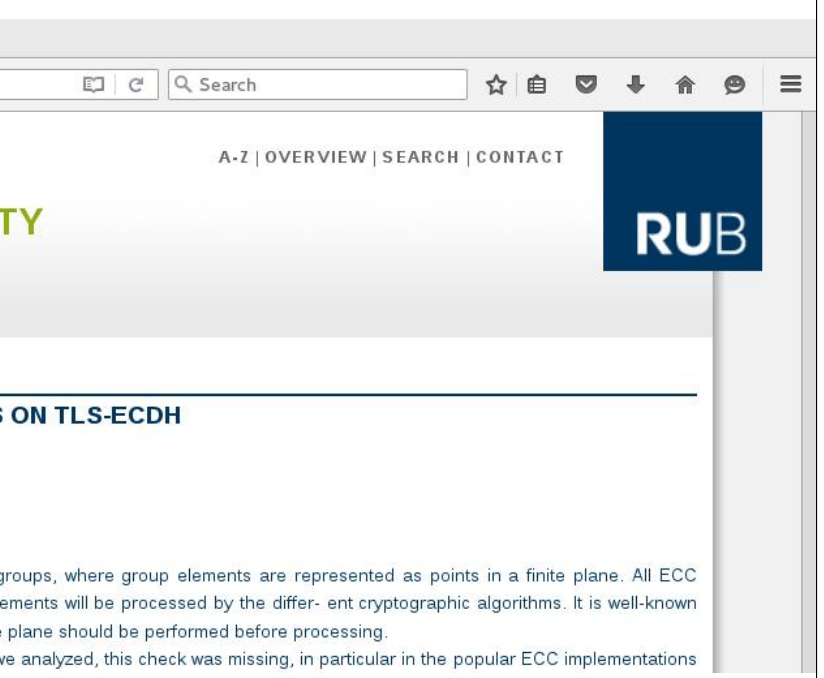
3. Choose twist-secure curve.

Longest
paper: f
improvin
from 199

o check that
e curve!"

om: Important
e broken by
–Meyer–Müller.

ger–Schwenk–
cessful attacks!
o forget.






Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

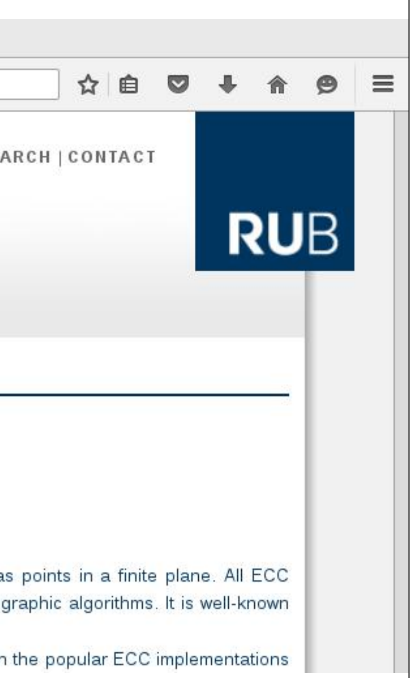
3. Choose twist-secure curve.

Longest section in
paper: fast finite-f
improving on algor
from 1999–2004 B

nat

rtant
by
Müller.

enk–
acks!






Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

3. Choose twist-secure curve.

Longest section in Curve255
paper: fast finite-field arithr
improving on algorithm desi
from 1999–2004 Bernstein.

Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

3. Choose twist-secure curve.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

3. Choose twist-secure curve.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

Curve25519 paper:
"free key validation"
eliminates these attacks.
No cost for checking input;
no code to forget.

1. Montgomery naturally follows 1986 Miller compression: send only $x$-coordinate, not $(x, y)$. Forces input onto "curve" or "twist". (Bonus: 32-byte keys!)

2. Montgomery ladder works correctly for inputs on twist.

3. Choose twist-secure curve.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper:
new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime
$2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

"Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography."

519 paper:
y validation"
es these attacks.
for checking input;
to forget.

tgomery naturally
1986 Miller compression:
y $x$-coordinate, not $(x, y)$.
nput onto "curve" or
(Bonus: 32-byte keys!)

tgomery ladder works
y for inputs on twist.

se twist-secure curve.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

"Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography."

Curve25
**multi-us**
1976 Dif
1999 Re
mode";

:
n”
ttacks.
ng input;

aturally
r compression:
inate, not $(x, y)$.
“curve” or
32-byte keys!)

dder works
s on twist.

ecure curve.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

“Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography.”

Curve25519 paper
**multi-user** DH sy
1976 Diffie–Hellm
1999 Rescorla “sta
mode”; 2006 NIST

sion:
$(x, y)$.
r
eys!)
s
e.



Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

“Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography.”



Curve25519 paper specified
**multi-user** DH system. See
1976 Diffie–Hellman; also, e
1999 Rescorla “static-static
mode”; 2006 NIST “C(0,2)”

Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

“Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography.”



Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla “static-static mode”; 2006 NIST “C(0,2)”.

Longest section in Curve25519 paper: fast finite-field arithmetic, improving on algorithm designs from 1999–2004 Bernstein.

Barely mentioned in paper: new programming language.

New prime $2^{255} - 19$.
Faster than NIST P-256 prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.

"Prime fields also have the virtue of minimizing the number of security concerns for elliptic-curve cryptography."



Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla "static-static mode"; 2006 NIST "C(0,2)".

Included security survey:

- Reductions: intolerably loose.
- Known attack ideas: rho etc.
- Multi-user batch attacks.
- Special-purpose hardware: 160-bit ECC is breakable.
- Small-subgroup attacks, invalid-curve attacks, etc.

section in Curve25519
ast finite-field arithmetic,
g on algorithm designs
99–2004 Bernstein.

entioned in paper:
gramming language.

me $2^{255} - 19$.
han NIST P-256 prime
$^{224} + 2^{192} + 2^{96} - 1$.

fields also have
e of minimizing the
of security concerns for
urve cryptography."

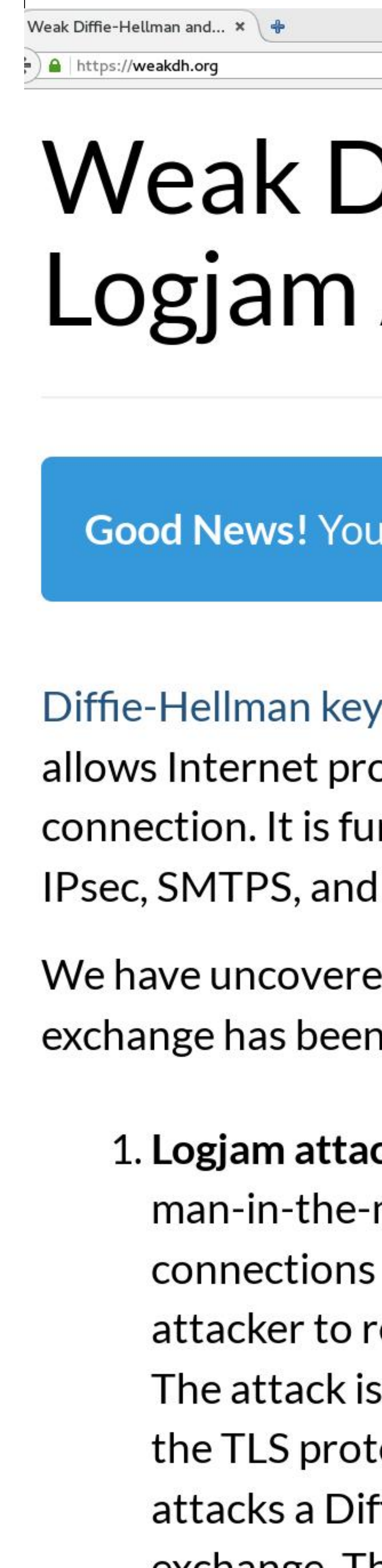


Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla "static-static mode"; 2006 NIST "C(0,2)".

Included security survey:

- Reductions: intolerably loose.
- Known attack ideas: rho etc.
- Multi-user batch attacks.
- Special-purpose hardware: 160-bit ECC is breakable.
- Small-subgroup attacks, invalid-curve attacks, etc.

2015: B



Curve25519
field arithmetic,
rithm designs
Bernstein.

in paper:
language.

19.
P-256 prime
$+ 2^{96} - 1$.

have
mizing the
concerns for
tography."

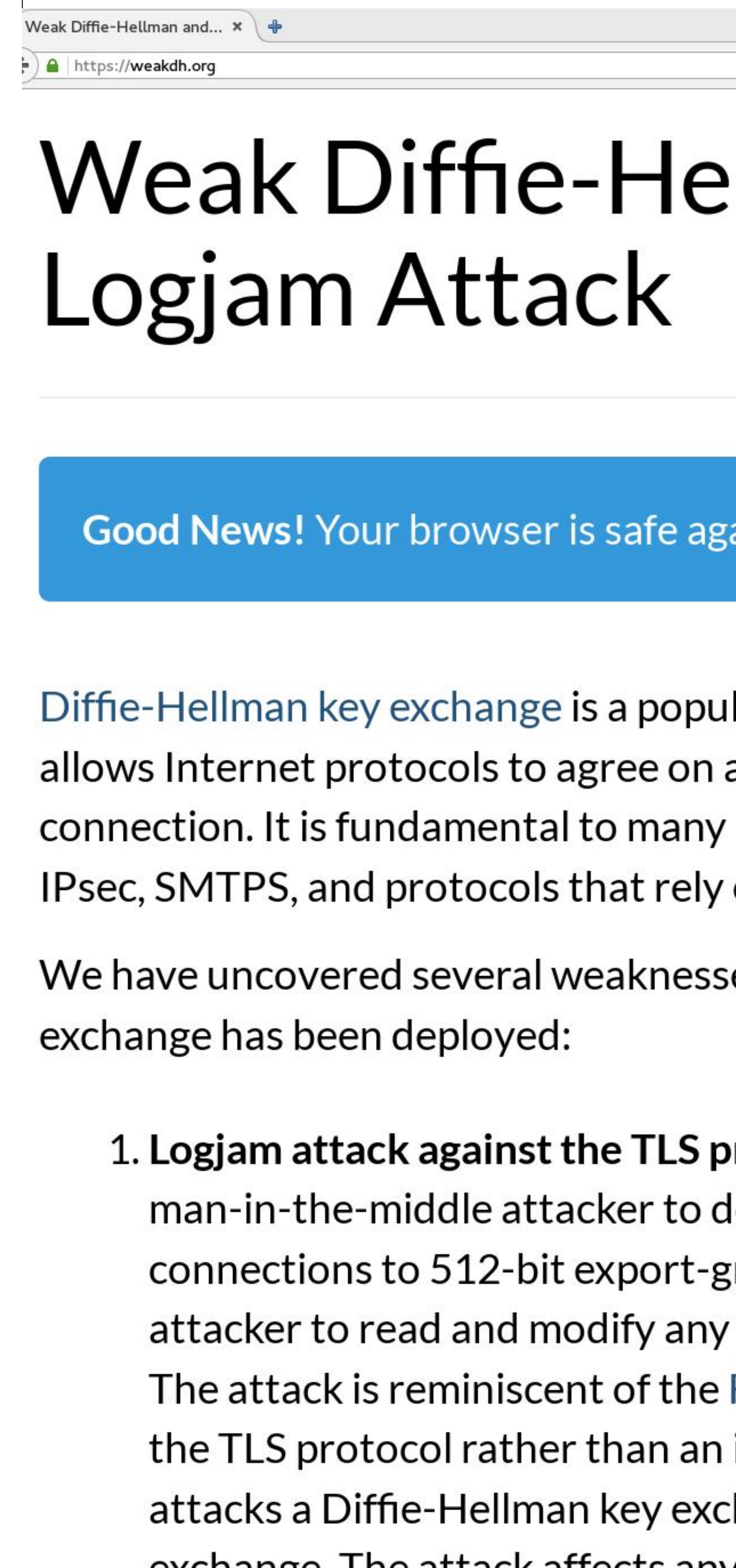


Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla "static-static mode"; 2006 NIST "C(0,2)".

Included security survey:

- Reductions: intolerably loose.
- Known attack ideas: rho etc.
- Multi-user batch attacks.
- Special-purpose hardware: 160-bit ECC is breakable.
- Small-subgroup attacks, invalid-curve attacks, etc.

2015: Beware bat



19
netic,
gns
me
1.
for



Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla "static-static mode"; 2006 NIST "C(0,2)".

Included security survey:

- Reductions: intolerably loose.
- Known attack ideas: rho etc.
- Multi-user batch attacks.
- Special-purpose hardware: 160-bit ECC is breakable.
- Small-subgroup attacks, invalid-curve attacks, etc.



2015: Beware batch attacks

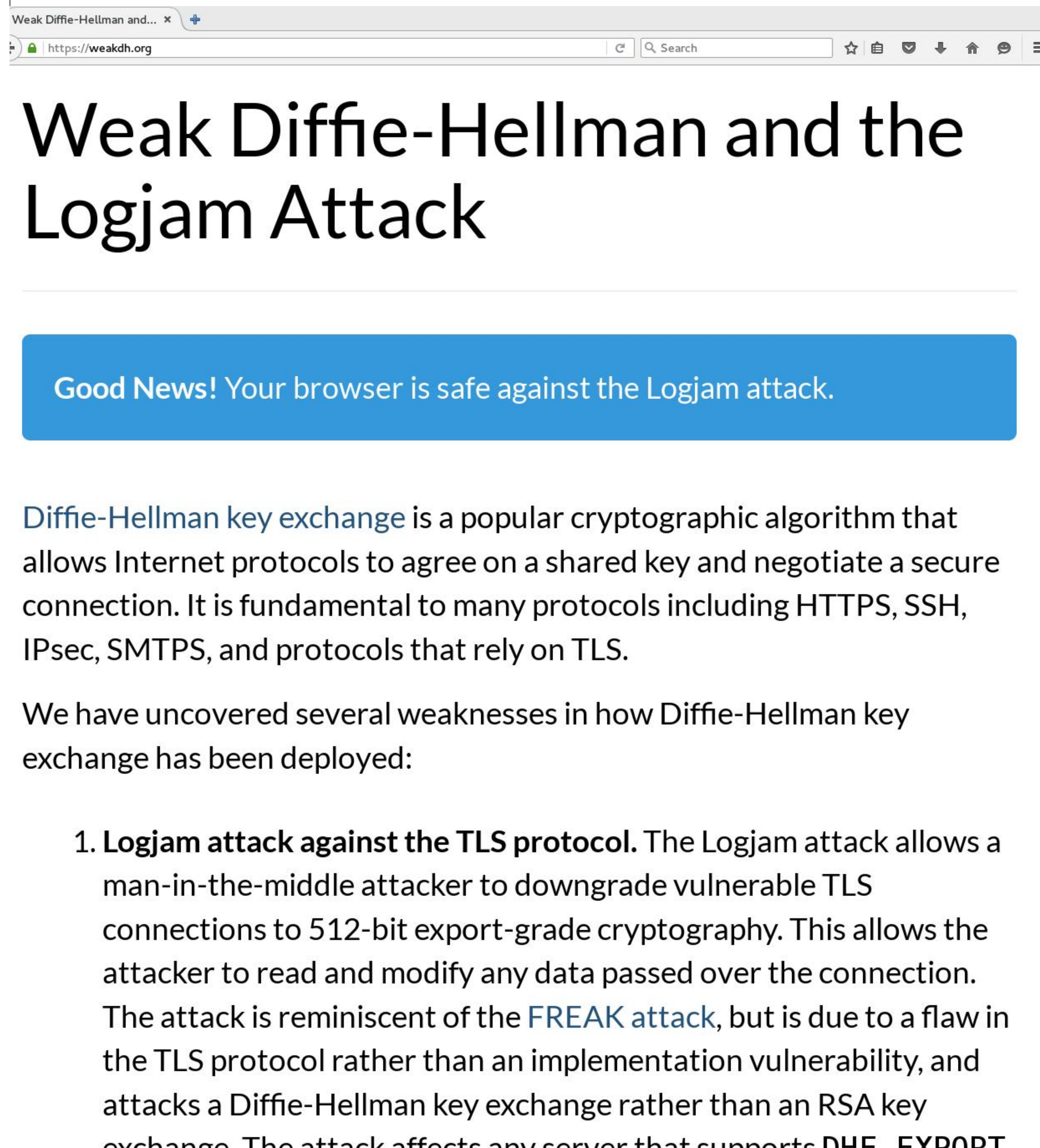
Curve25519 paper specified a **multi-user** DH system. See 1976 Diffie–Hellman; also, e.g., 1999 Rescorla "static-static mode"; 2006 NIST "C(0,2)".

Included security survey:

- Reductions: intolerably loose.
- Known attack ideas: rho etc.
- Multi-user batch attacks.
- Special-purpose hardware: 160-bit ECC is breakable.
- Small-subgroup attacks, invalid-curve attacks, etc.



2015: Beware batch attacks.


Weak Diffie-Hellman and... ×
https://weakdh.org
Search

# Weak Diffie-Hellman and the Logjam Attack

**Good News!** Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

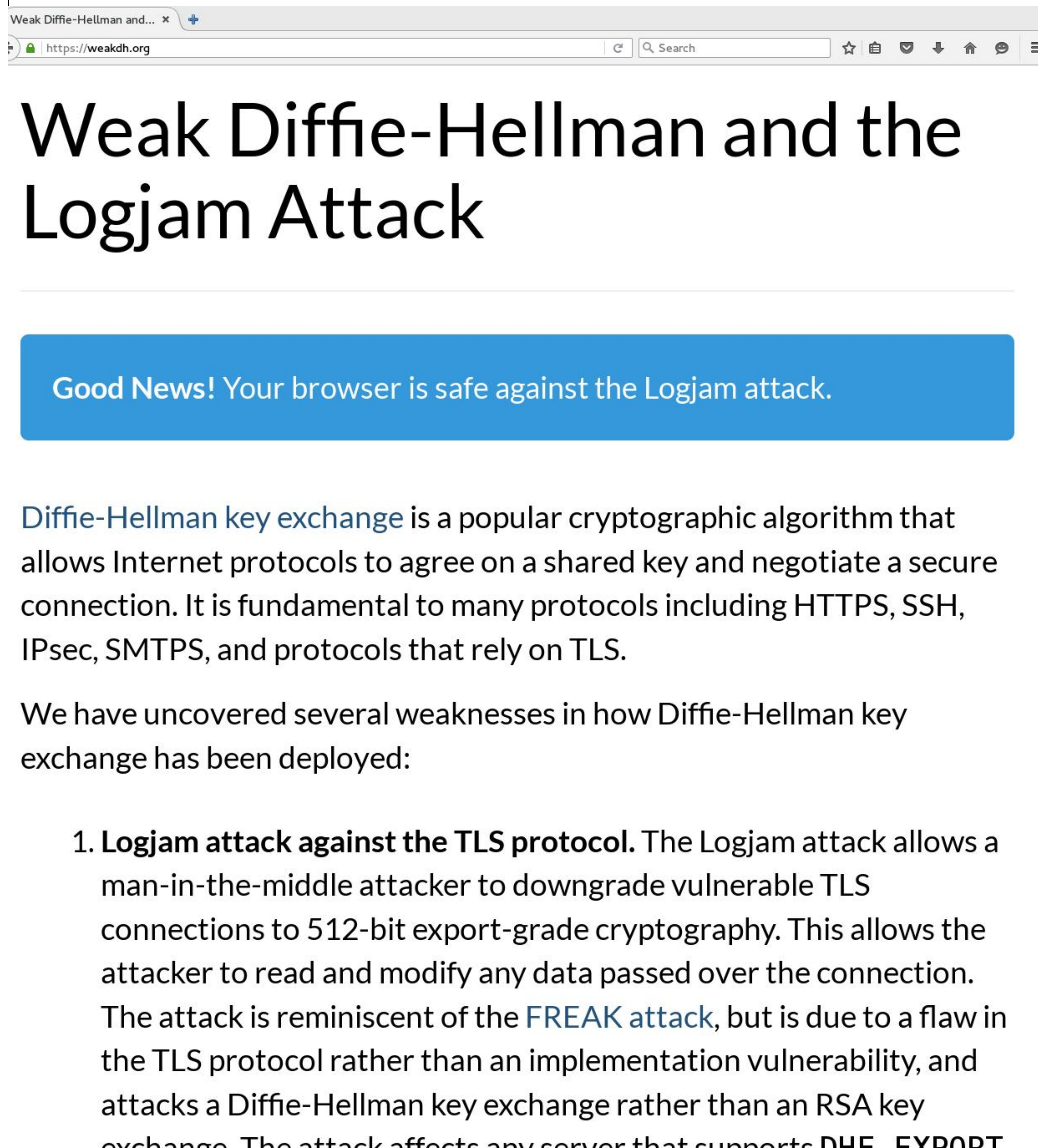
1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DHE_EXPORT

519 paper specified a
ser DH system. See
ffie–Hellman; also, e.g.,
scorla “static-static
2006 NIST “C(0,2)”.

security survey:
tions: intolerably loose.
n attack ideas: rho etc.
user batch attacks.
al-purpose hardware:
t ECC is breakable.
subgroup attacks,
l-curve attacks, etc.



2015: Beware batch attacks.


Weak Diffie-Hellman and... × https://weakdh.org Search

# Weak Diffie-Hellman and the Logjam Attack

**Good News!** Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DHE_EXPORT

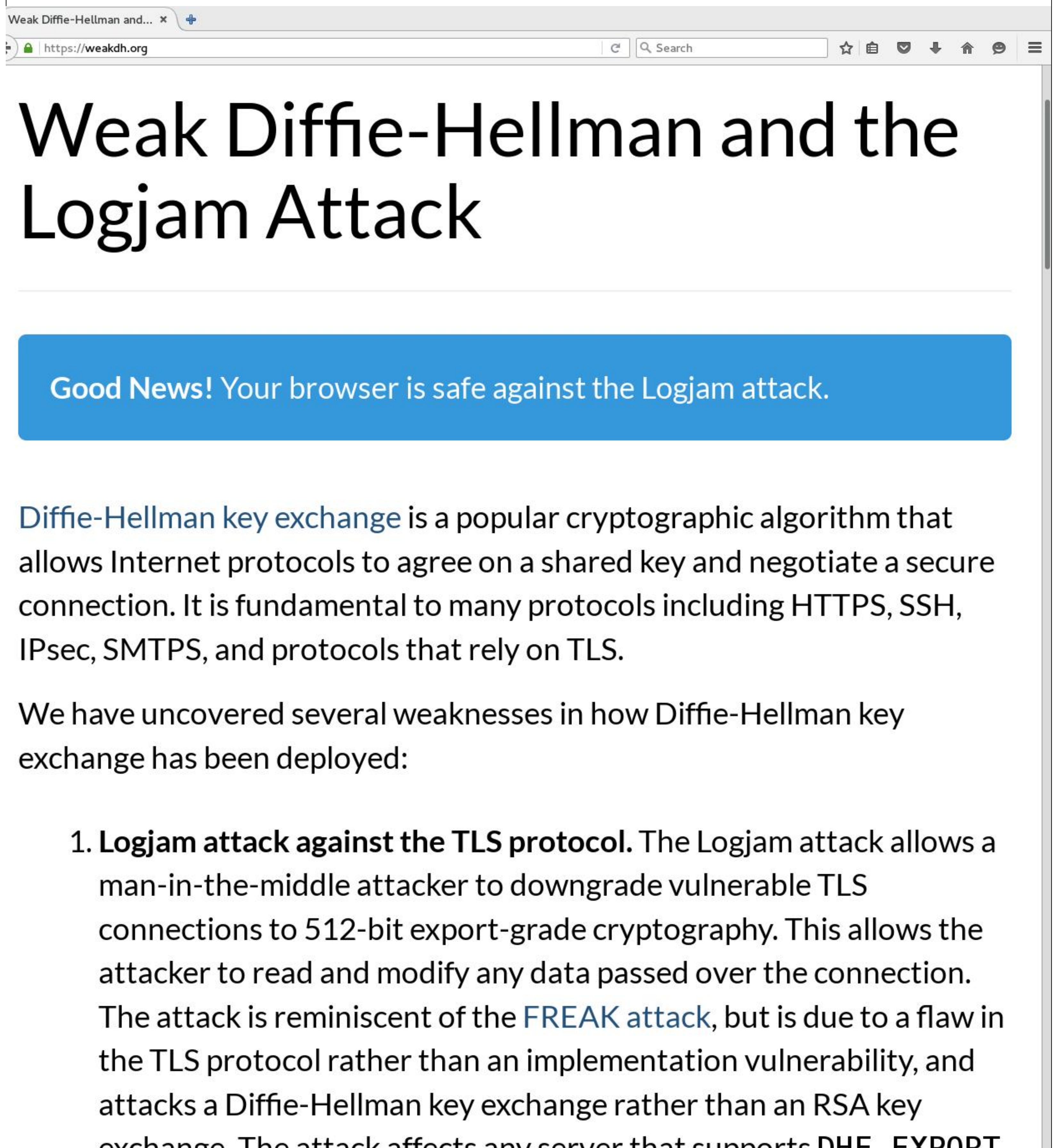
Paper sk
attack m
composi
multi-us
(as in, e
“public-k
attacks o
(the mo
“Reveal”
Freire–H
dishones
(as in, e
Cash–Ki
keys as s
e.g., 200

specified a
stem. See
an; also, e.g.,
atic-static
T "C(0,2)".

survey:
olerably loose.
deas: rho etc.
attacks.
hardware:
oreakable.
attacks,
acks, etc.



2015: Beware batch attacks.




# Weak Diffie-Hellman and the Logjam Attack

**Good News!** Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DHE_EXPORT

Paper sketched co
attack model, incl
composition with s
multi-user secret-k
(as in, e.g., 2001 I
"public-key auther
attacks on secret-l
(the motivation gi
"Reveal" queries i
Freire–Hofheinz–K
dishonest key regis
(as in, e.g., Euroc
Cash–Kiltz–Shoup
keys as strings (all
e.g., 2000 Biehl–M

a

.g.,

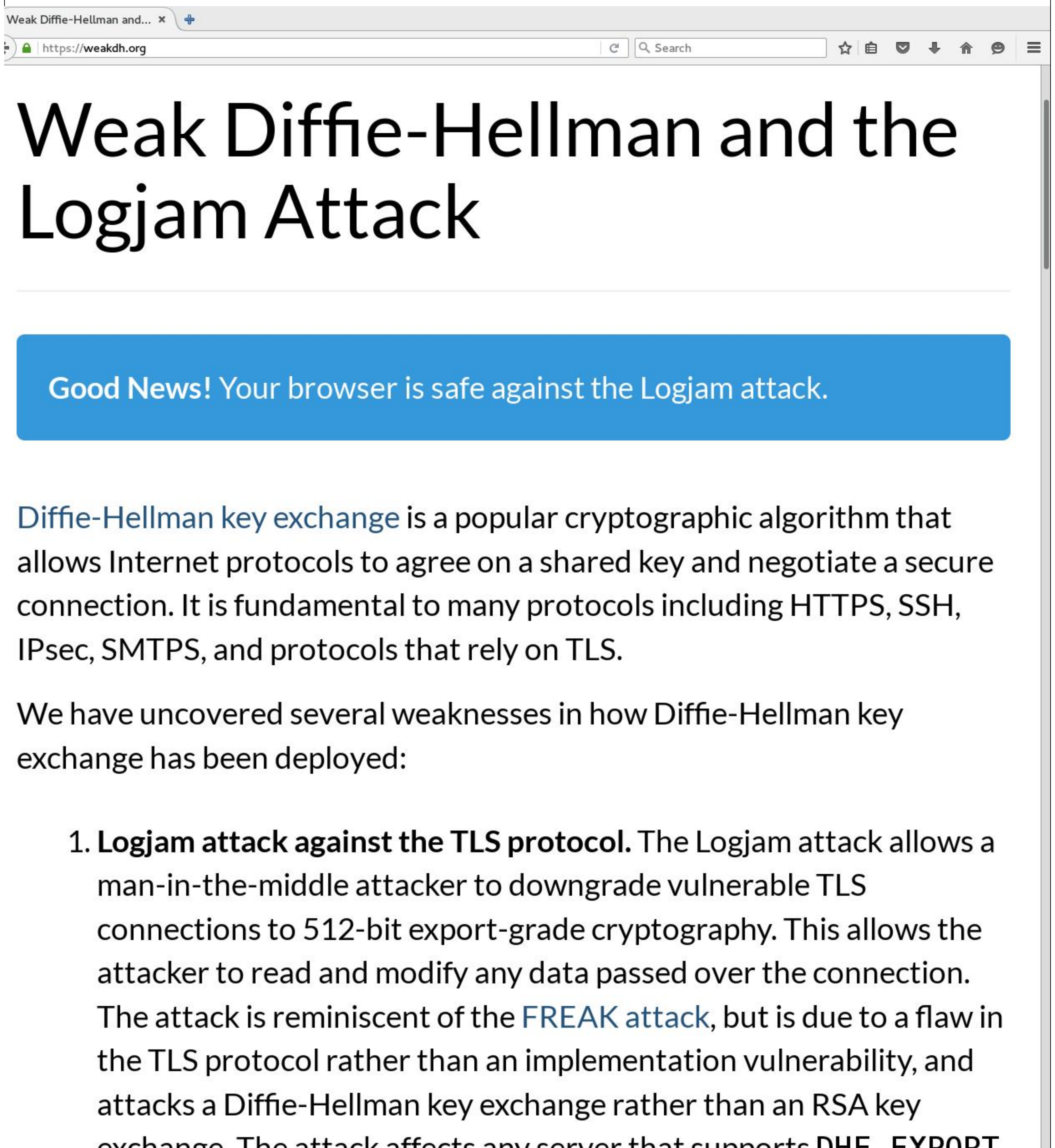
'.

ose.

etc.



2015: Beware batch attacks.


Weak Diffie-Hellman and... ×
https://weakdh.org

# Weak Diffie-Hellman and the Logjam Attack

**Good News!** Your browser is safe against the Logjam attack.

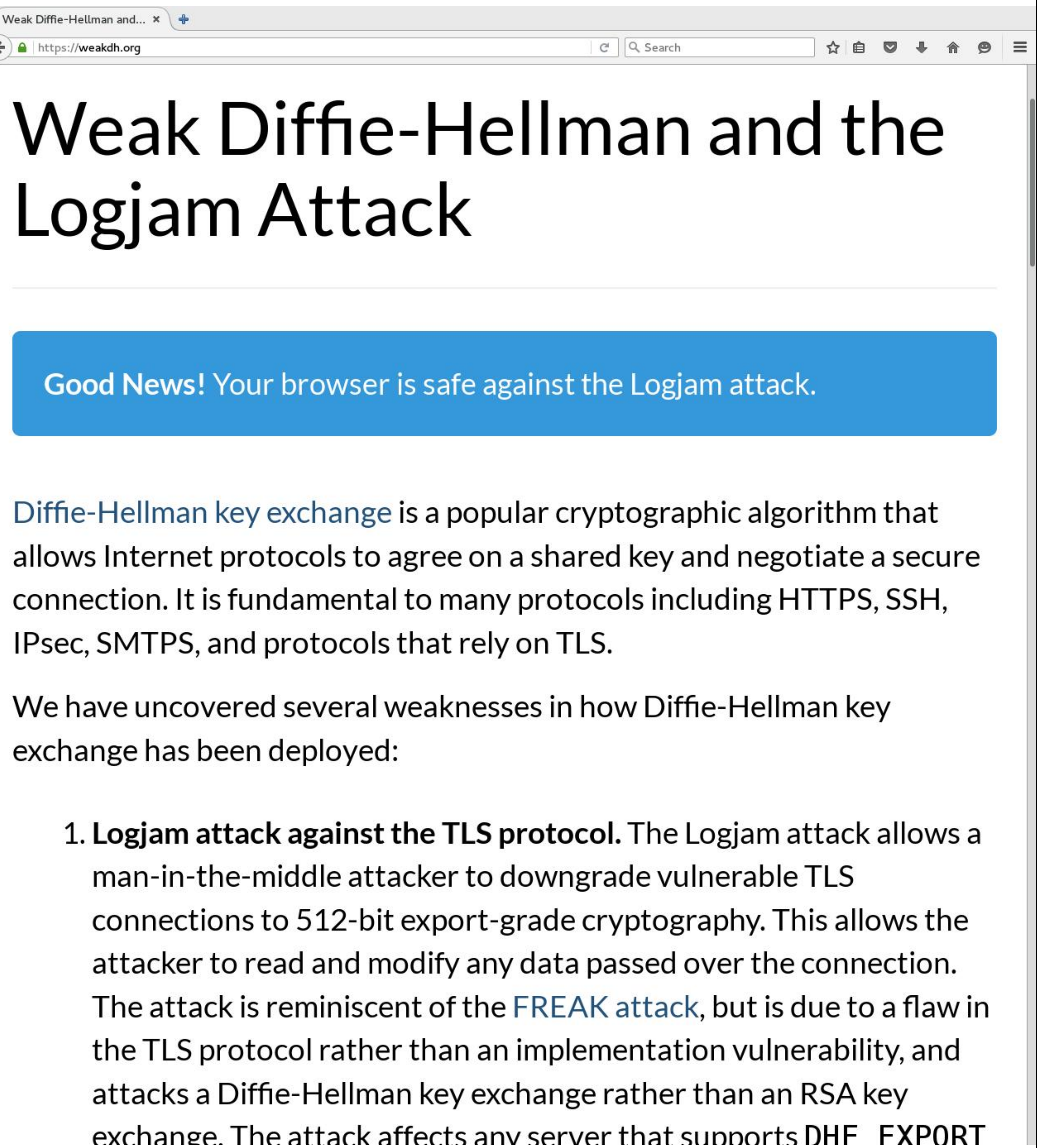
Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DHE_EXPORT

Paper sketched common-sen
attack model, including
composition with subsequen
multi-user secret-key system
(as in, e.g., 2001 Bernstein
"public-key authenticators")
attacks on secret-key system
(the motivation given for
"Reveal" queries in PKC 20
Freire–Hofheinz–Kiltz–Pater
dishonest key registrations
(as in, e.g., Eurocrypt 2008
Cash–Kiltz–Shoup);
keys as strings (allows mode
e.g., 2000 Biehl–Meyer–Mül

## 2015: Beware batch attacks.


Weak Diffie-Hellman and... ×
https://weakdh.org

# Weak Diffie-Hellman and the Logjam Attack

**Good News!** Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DHE_EXPORT



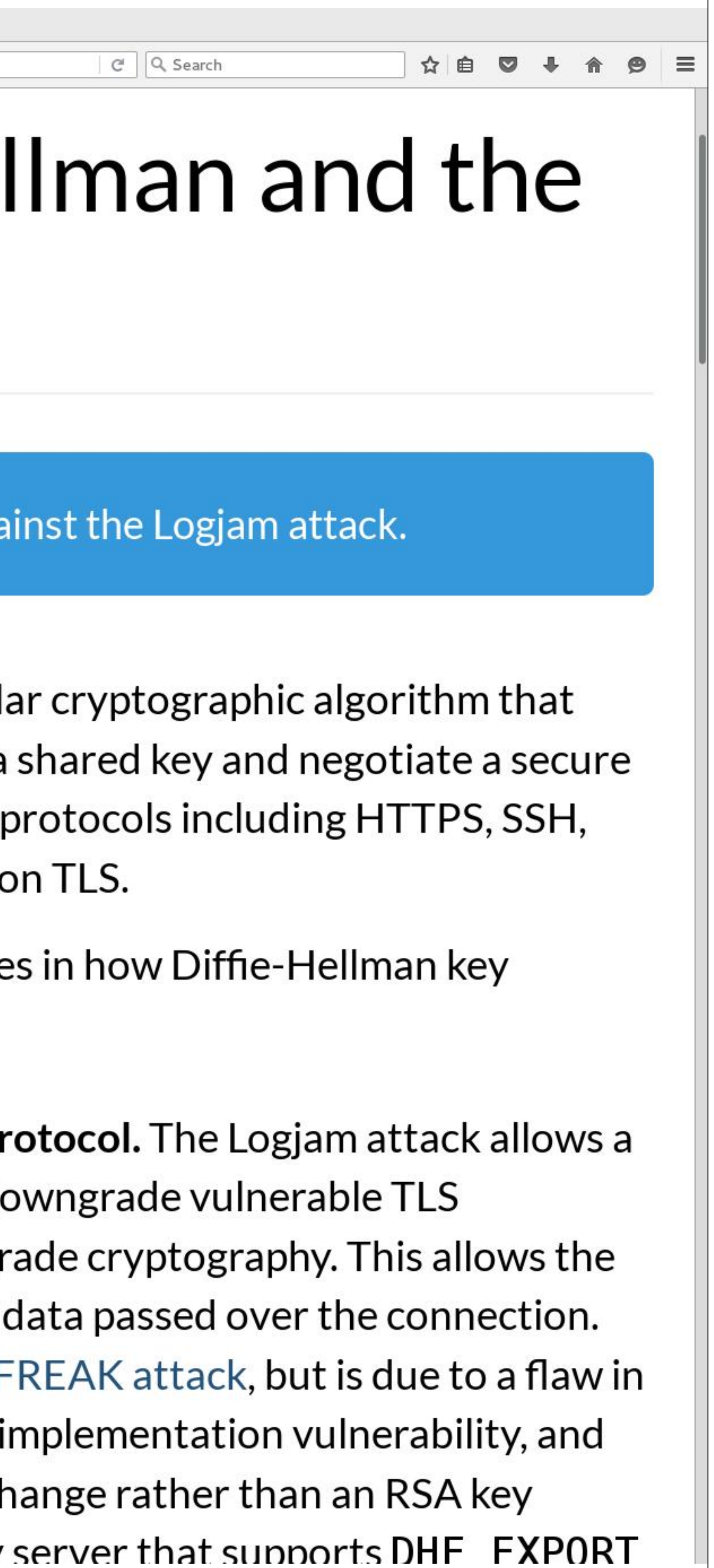
Paper sketched common-sense attack model, including composition with subsequent multi-user secret-key system (as in, e.g., 2001 Bernstein "public-key authenticators"); attacks on secret-key system (the motivation given for "Reveal" queries in PKC 2013 Freire–Hofheinz–Kiltz–Paterson); dishonest key registrations (as in, e.g., Eurocrypt 2008 Cash–Kiltz–Shoup); keys as strings (allows modeling, e.g., 2000 Biehl–Meyer–Müller).

eware batch attacks.



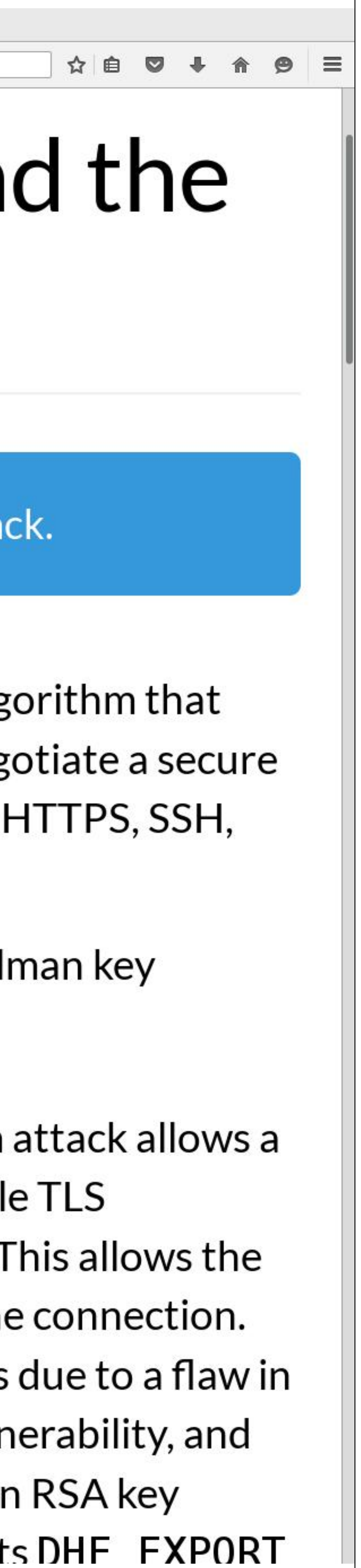

Paper sketched common-sense attack model, including composition with subsequent multi-user secret-key system (as in, e.g., 2001 Bernstein "public-key authenticators");
attacks on secret-key system (the motivation given for "Reveal" queries in PKC 2013 Freire–Hofheinz–Kiltz–Paterson);
dishonest key registrations (as in, e.g., Eurocrypt 2008 Cash–Kiltz–Shoup);
keys as strings (allows modeling, e.g., 2000 Biehl–Meyer–Müller).

ch attacks.



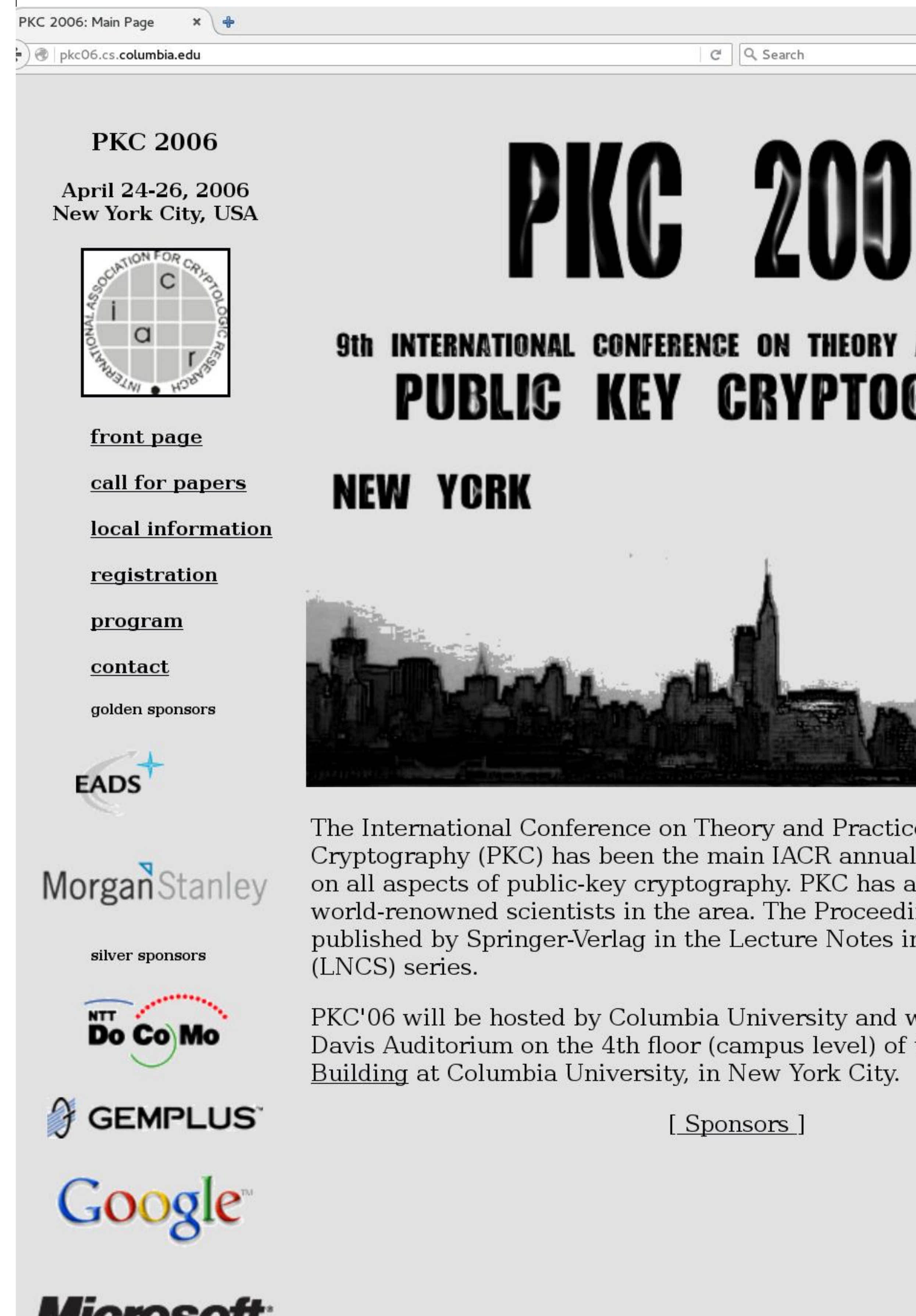



Paper sketched common-sense attack model, including composition with subsequent multi-user secret-key system (as in, e.g., 2001 Bernstein "public-key authenticators"); attacks on secret-key system (the motivation given for "Reveal" queries in PKC 2013 Freire–Hofheinz–Kiltz–Paterson); dishonest key registrations (as in, e.g., Eurocrypt 2008 Cash–Kiltz–Shoup); keys as strings (allows modeling, e.g., 2000 Biehl–Meyer–Müller).



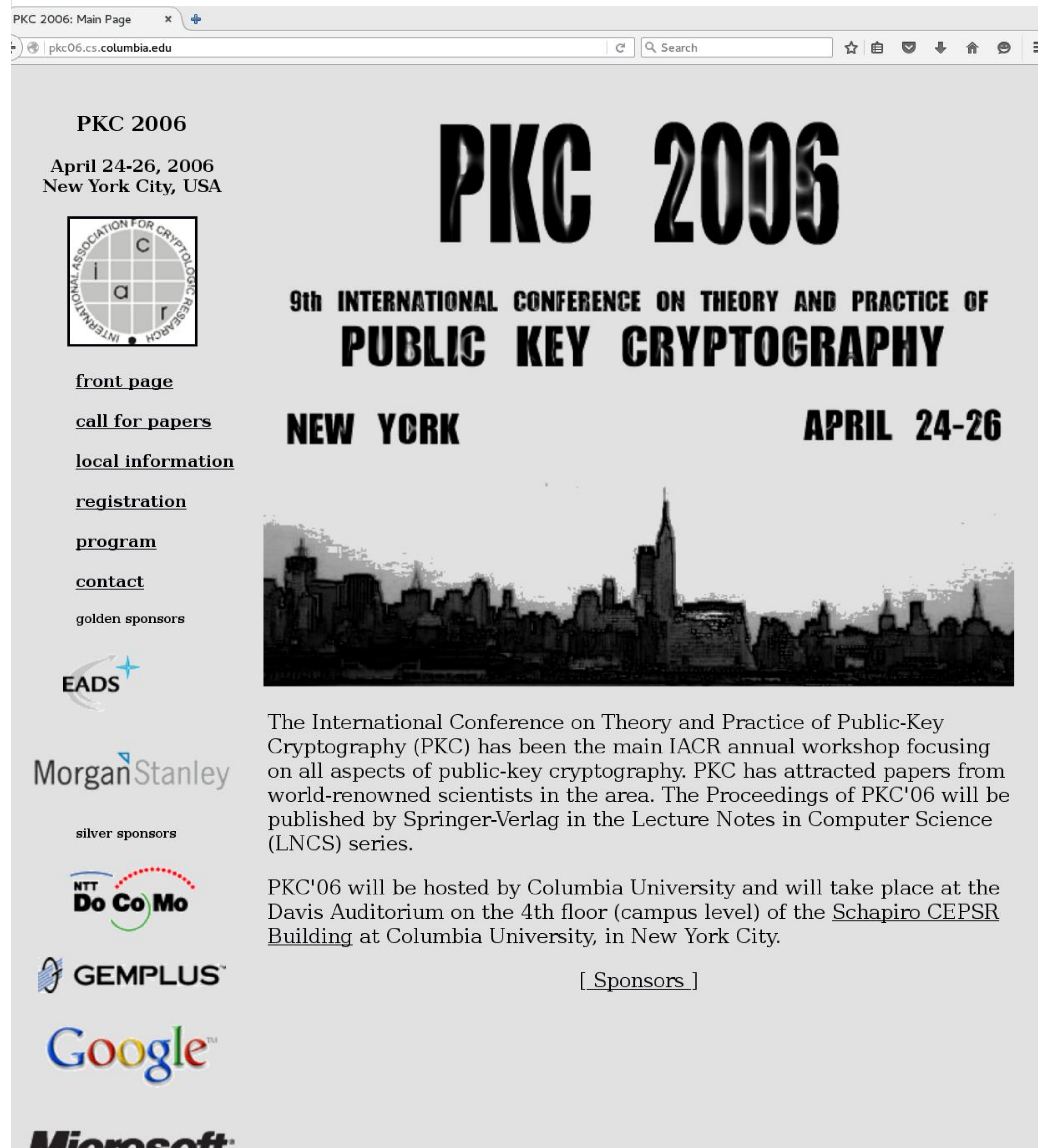



Paper sketched common-sense attack model, including composition with subsequent multi-user secret-key system (as in, e.g., 2001 Bernstein “public-key authenticators”); attacks on secret-key system (the motivation given for “Reveal” queries in PKC 2013 Freire–Hofheinz–Kiltz–Paterson); dishonest key registrations (as in, e.g., Eurocrypt 2008 Cash–Kiltz–Shoup); keys as strings (allows modeling, e.g., 2000 Biehl–Meyer–Müller).



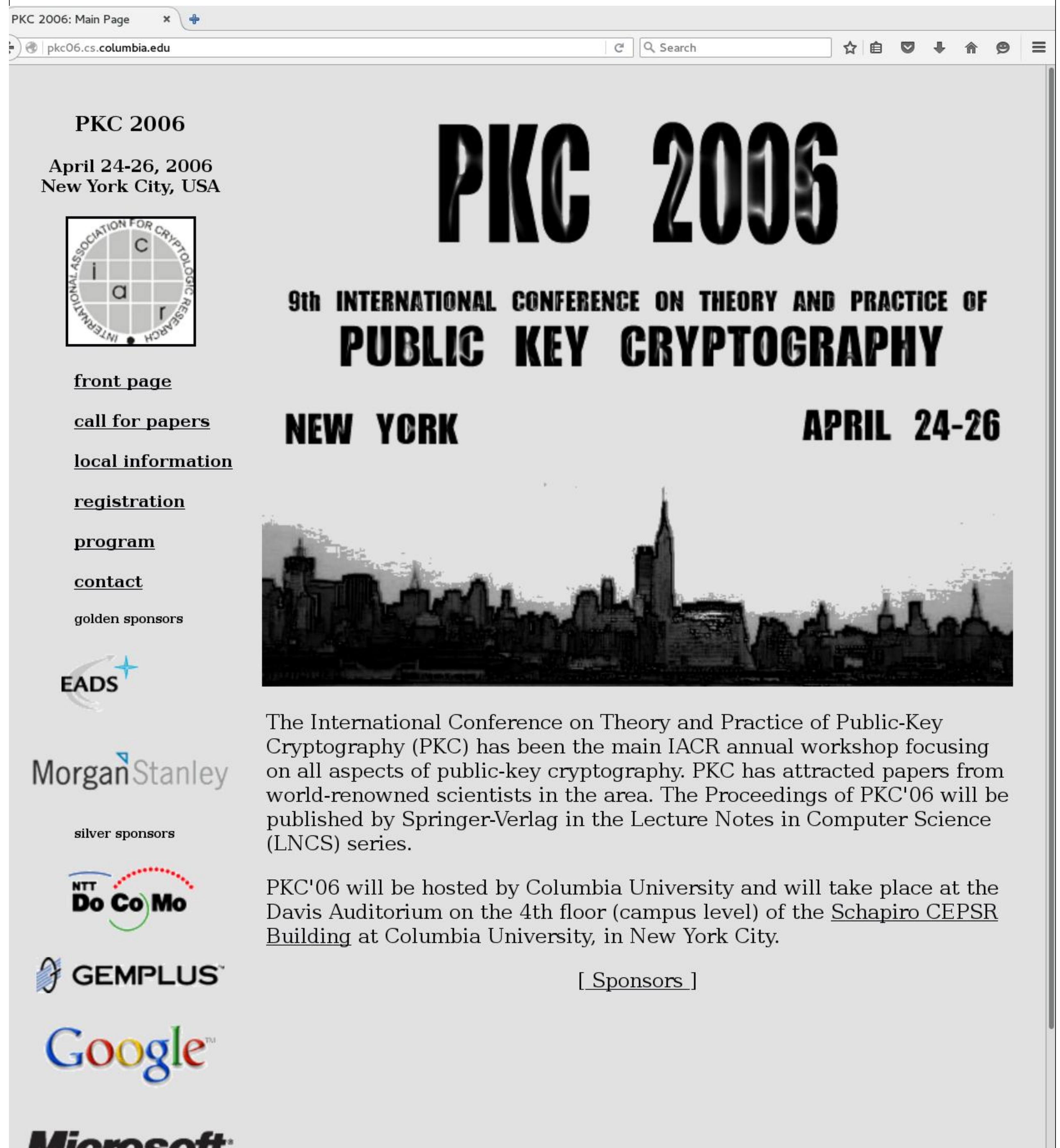
Paper sketched common-sense attack model, including composition with subsequent multi-user secret-key system (as in, e.g., 2001 Bernstein "public-key authenticators"); attacks on secret-key system (the motivation given for "Reveal" queries in PKC 2013 Freire–Hofheinz–Kiltz–Paterson); dishonest key registrations (as in, e.g., Eurocrypt 2008 Cash–Kiltz–Shoup); keys as strings (allows modeling, e.g., 2000 Biehl–Meyer–Müller).





ketched common-sense
nodel, including
tion with subsequent
er secret-key system
.g., 2001 Bernstein
key authenticators" );
on secret-key system
tivation given for
' queries in PKC 2013
lofheinz–Kiltz–Paterson);
t key registrations
.g., Eurocrypt 2008
ltz–Shoup);
strings (allows modeling,
00 Biehl–Meyer–Müller).



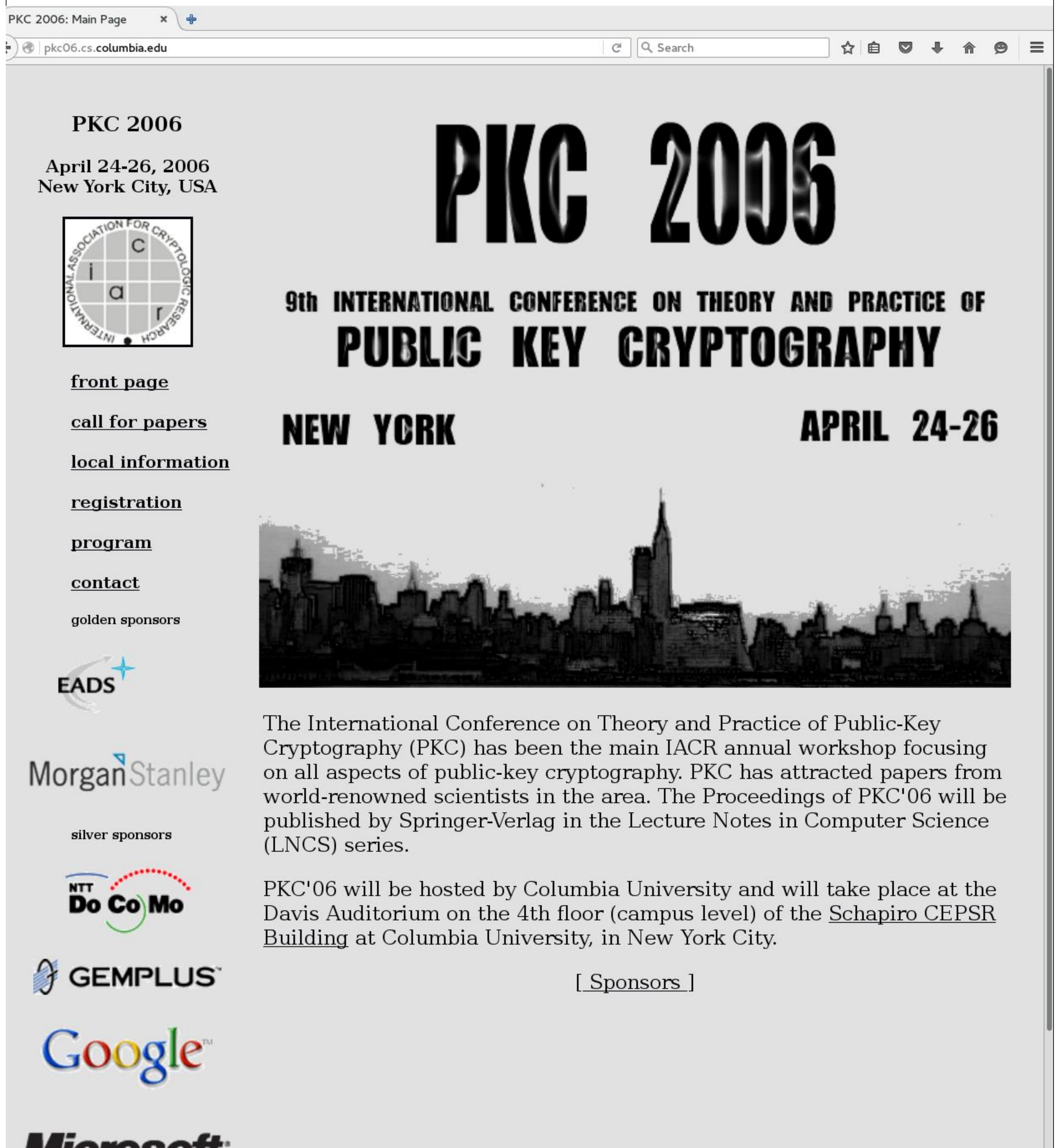



Email fr

It is my
that you
new Dif
records
PKC'06.

mmon-sense
uding
subsequent
key system
Bernstein
nticators” );
key system
ven for
n PKC 2013
Kiltz–Paterson);
strations
rypt 2008
);
lows modeling,
Meyer–Müller).






Email from progra

---

```
It is my pleasur
that your paper
new Diffie-Hellm
records" was acc
PKC'06. Congratu
```

---

se

t

;

n

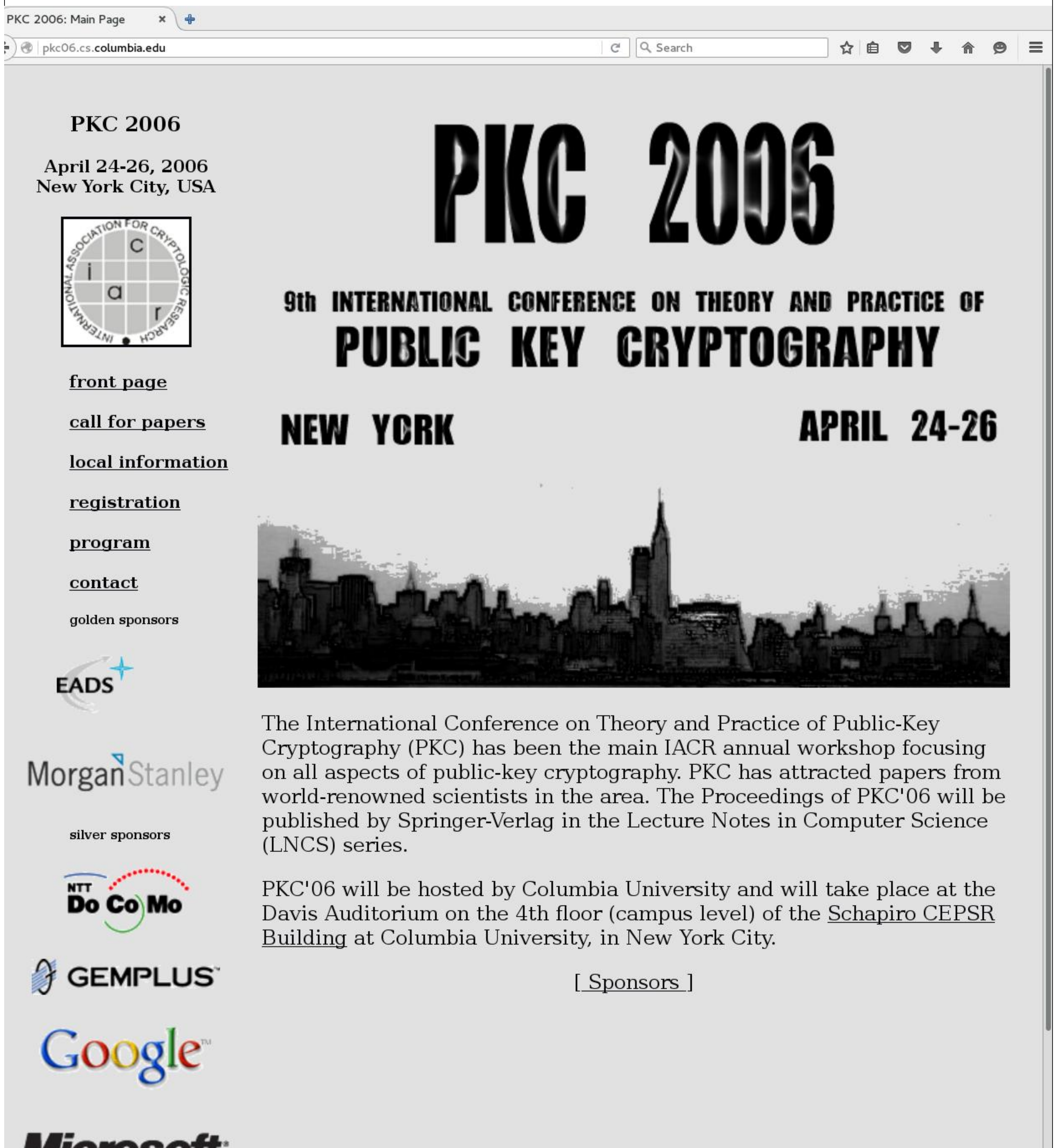
13

son);

ling,

ler).



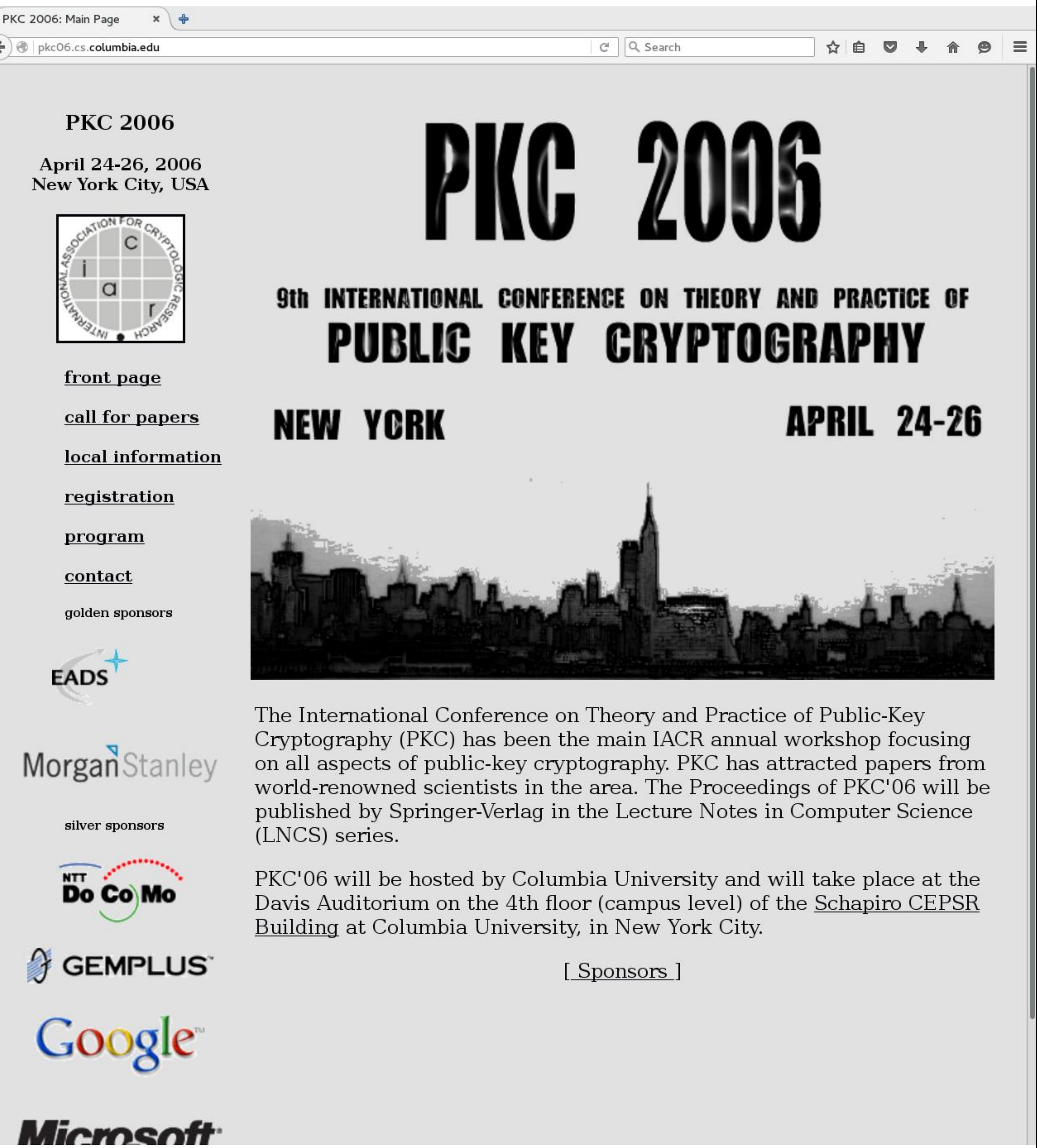



Email from program chairs:

---

```
It is my pleasure to info
that your paper "Curve255
new Diffie-Hellman speed
records" was accepted to
PKC'06. Congratulations!
```

---



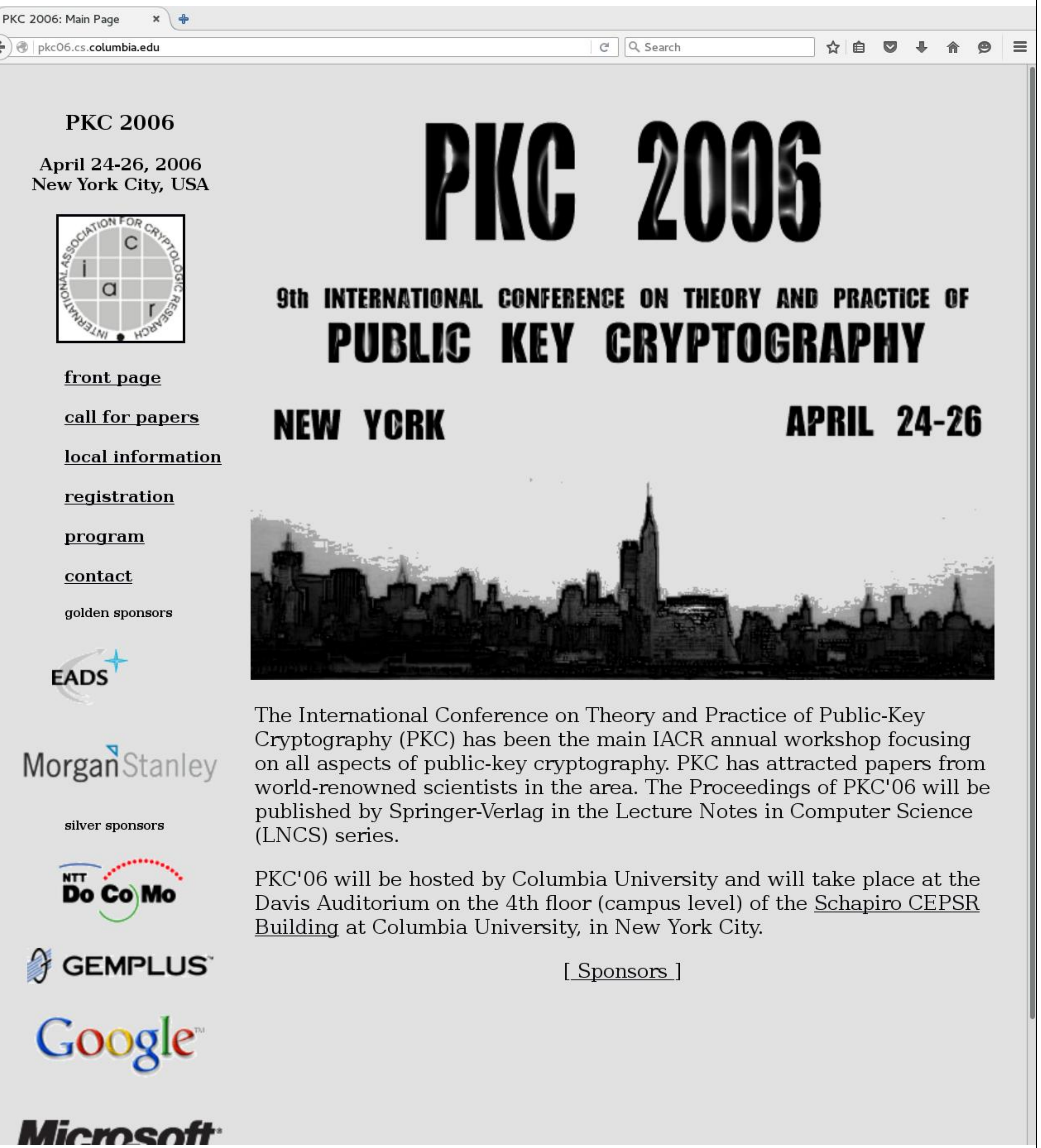



# Email from program chairs:

---

It is my pleasure to inform you that your paper "Curve25519: new Diffie-Hellman speed records" was accepted to PKC'06. Congratulations!

---






Email from program chairs:

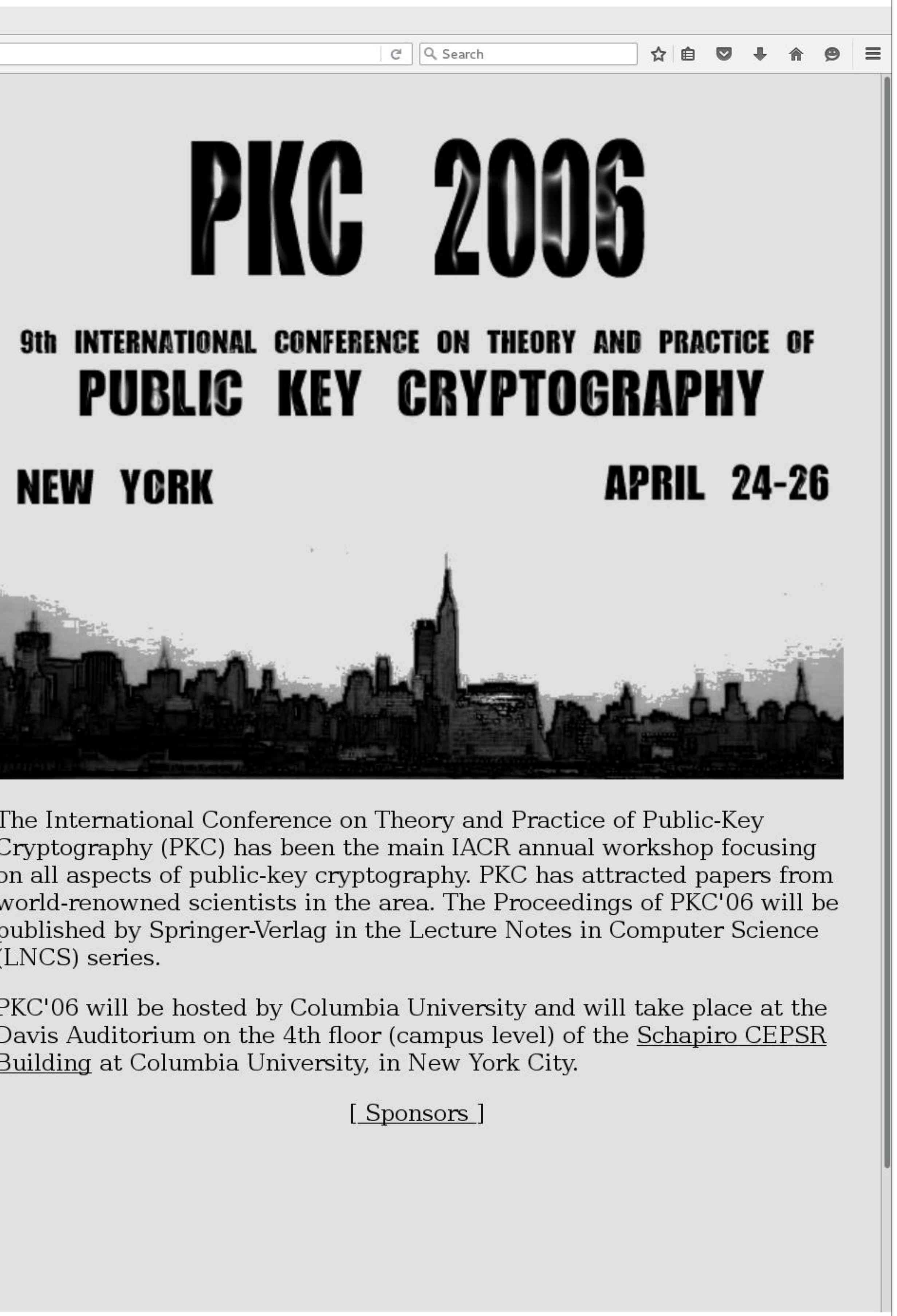
---

It is my pleasure to inform you that your paper "Curve25519: new Diffie-Hellman speed records" was accepted to PKC'06. Congratulations!

---

Below please find the reviewers' comments on your paper "Curve25519: new Diffie-Hellman speed records" that was submitted to PKC 2006.

---






Email from program chairs:

---

```
It is my pleasure to inform you
that your paper "Curve25519:
new Diffie-Hellman speed
records" was accepted to
PKC'06. Congratulations!
```

---

```
Below please find the reviewers'
comments on your paper
"Curve25519: new Diffie-
Hellman speed records"
that was submitted to PKC 2006.
```

---

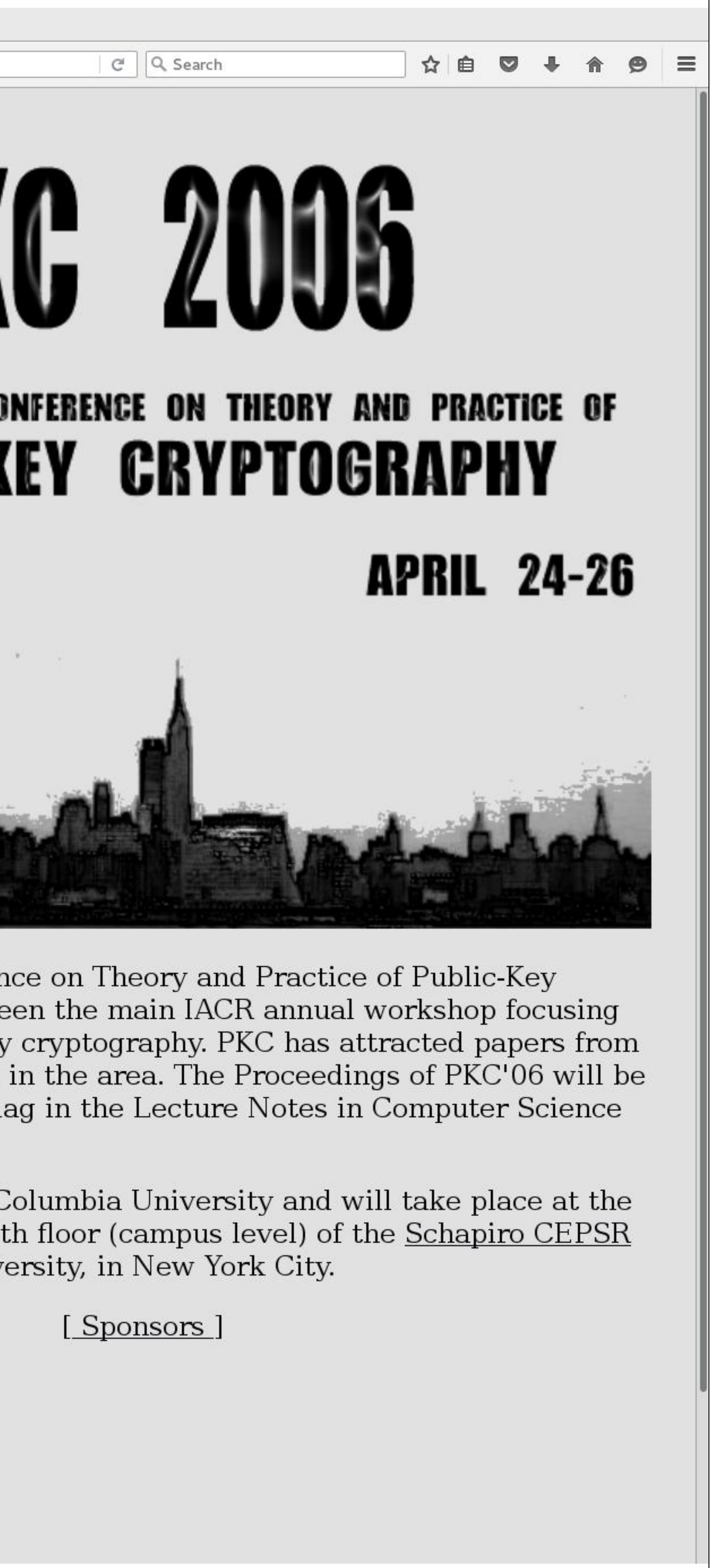
Reviewe

---

```
While I
this is
I think
"real" i
I don't
correctn
the appr
paper to
```

---

So engin






# Email from program chairs:

---

It is my pleasure to inform you that your paper "Curve25519: new Diffie-Hellman speed records" was accepted to PKC'06. Congratulations!

---

Below please find the reviewers' comments on your paper "Curve25519: new Diffie-Hellman speed records" that was submitted to PKC 2006.

---

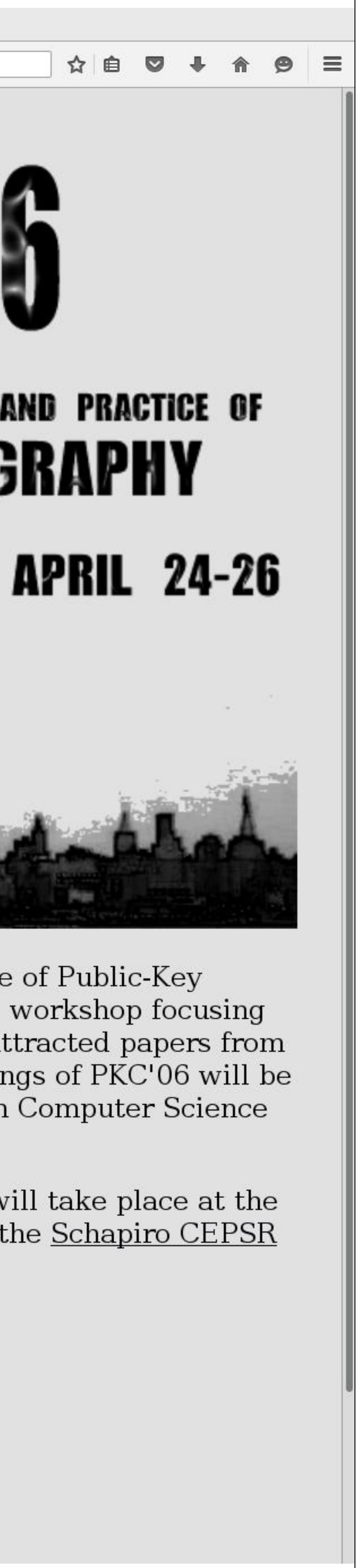
# Reviewer #1:

---

While I think (f
this is a nice e
I think that thi
"real" research
I don't question
correctness but
the appropriaten
paper to the con

---

So engineering isn






Email from program chairs:

---

```
It is my pleasure to inform you
that your paper "Curve25519:
new Diffie-Hellman speed
records" was accepted to
PKC'06. Congratulations!
```

---

```
Below please find the reviewers'
comments on your paper
"Curve25519: new Diffie-
Hellman speed records"
that was submitted to PKC 2006.
```

---

Reviewer #1:

---

```
While I think (frankly) t
this is a nice engineerin
I think that this is not
"real" research paper.
I don't question the
correctness but I questio
the appropriateness of th
paper to the conference.
```

---

So engineering isn't research

Email from program chairs:

---

It is my pleasure to inform you that your paper "Curve25519: new Diffie-Hellman speed records" was accepted to PKC'06. Congratulations!

---

Below please find the reviewers' comments on your paper "Curve25519: new Diffie-Hellman speed records" that was submitted to PKC 2006.

---



Reviewer #1:

---

While I think (frankly) that this is a nice engineering work, I think that this is not a "real" research paper.
I don't question the correctness but I question the appropriateness of the paper to the conference.

---

So engineering isn't research?

om program chairs:

---

y pleasure to inform you
ur paper "Curve25519:
fie-Hellman speed
" was accepted to
 Congratulations!

---

lease find the reviewers'
s on your paper
5519: new Diffie-
 speed records"
s submitted to PKC 2006.

---



Reviewer #1:

---

While I think (frankly) that
this is a nice engineering work,
I think that this is not a
"real" research paper.
I don't question the
correctness but I question
the appropriateness of the
paper to the conference.

---

So engineering isn't research?

Reviewe

---

... bene
against
apparre
and very

On the
does no
nor doe
things
"conjec
through
a consi
achieve

---

m chairs:

---

e to inform you
"Curve25519:
an speed
epted to
lations!

---

d the reviewers'
paper
Diffie-
cords"
ed to PKC 2006.

---



Reviewer #1:

---

```
While I think (frankly) that
this is a nice engineering work,
I think that this is not a
"real" research paper.
I don't question the
correctness but I question
the appropriateness of the
paper to the conference.
```

---

So engineering isn't research?

Reviewer #2:

---

```
... benefits inc
against timing a
apparent patent
and very good sp

On the negative
does not introdu
nor does it atte
things rigorousl
"conjecture" is
throughout). It
a considerable e
achievement.
```

---

rm you
19:

iewers'

2006.



Reviewer #1:

---

```
While I think (frankly) that
this is a nice engineering work,
I think that this is not a
"real" research paper.
I don't question the
correctness but I question
the appropriateness of the
paper to the conference.
```

---

So engineering isn't research?

Reviewer #2:

---

```
... benefits including pr
against timing attacks, n
apparrent patent infringe
and very good speed. ...

On the negative side, the
does not introduce novel
nor does it attempt to pr
things rigorously (the wo
"conjecture" is used repe
throughout). It is princi
a considerable engineerin
achievement.
```

---

## Reviewer #1:

---

```
While I think (frankly) that
this is a nice engineering work,
I think that this is not a
"real" research paper.
I don't question the
correctness but I question
the appropriateness of the
paper to the conference.
```

---

So engineering isn't research?



## Reviewer #2:

---

```
... benefits including protection
against timing attacks, no
apparrent patent infringements,
and very good speed. ...

On the negative side, the paper
does not introduce novel ideas,
nor does it attempt to prove
things rigorously (the word
"conjecture" is used repeatedly
throughout). It is principally
a considerable engineering
achievement.
```

---

r #1:

```
 think (frankly) that
 a nice engineering work,
 that this is not a
research paper.
 question the
ness but I question
ropriateness of the
o the conference.
```

neering isn't research?



Reviewer #2:

```
... benefits including protection
against timing attacks, no
apparrent patent infringements,
and very good speed. ...

On the negative side, the paper
does not introduce novel ideas,
nor does it attempt to prove
things rigorously (the word
"conjecture" is used repeatedly
throughout). It is principally
a considerable engineering
achievement.
```

e.g. "Br
function
the shar
public k
extreme
attack is
performi
on a typ
cipher. .
have ord
a margi
same co
but this
extra sp

rankly) that
ngineering work,
s is not a
paper.
the
I question
ess of the
ference.

't research?



Reviewer #2:

```
... benefits including protection
against timing attacks, no
apparrent patent infringements,
and very good speed. ...

On the negative side, the paper
does not introduce novel ideas,
nor does it attempt to prove
things rigorously (the word
"conjecture" is used repeatedly
throughout). It is principally
a considerable engineering
achievement.
```

e.g. "Breaking the
function—for exar
the shared secret f
public keys—is co
extremely difficult.
attack is more exp
performing a brute
on a typical 128-b
cipher. . . . Curves
have order divisibl
a marginally larger
same conjectured
but this is outweig
extra speed of cur

hat
g work,
a

n
e

n?



Reviewer #2:

---

```
... benefits including protection
against timing attacks, no
apparrent patent infringements,
and very good speed. ...

On the negative side, the paper
does not introduce novel ideas,
nor does it attempt to prove
things rigorously (the word
"conjecture" is used repeatedly
throughout). It is principally
a considerable engineering
achievement.
```

---



e.g. "Breaking the Curve255
function—for example, comp
the shared secret from the t
public keys—is conjectured
extremely difficult. Every kn
attack is more expensive tha
performing a brute-force sea
on a typical 128-bit secret-k
cipher. . . . Curves of this sh
have order divisible by 4, rec
a marginally larger prime fo
same conjectured security le
but this is outweighed by th
extra speed of curve operati

Reviewer #2:

---

```
... benefits including protection
against timing attacks, no
apparrent patent infringements,
and very good speed. ...

On the negative side, the paper
does not introduce novel ideas,
nor does it attempt to prove
things rigorously (the word
"conjecture" is used repeatedly
throughout). It is principally
a considerable engineering
achievement.
```

---



e.g. "Breaking the Curve25519 function—for example, computing the shared secret from the two public keys—is conjectured to be extremely difficult. Every known attack is more expensive than performing a brute-force search on a typical 128-bit secret-key cipher. . . . Curves of this shape have order divisible by 4, requiring a marginally larger prime for the same conjectured security level, but this is outweighed by the extra speed of curve operations."

r #2:

---

efits including protection
 timing attacks, no
nt patent infringements,
y good speed. ...

negative side, the paper
t introduce novel ideas,
s it attempt to prove
rigorously (the word
ture" is used repeatedly
out). It is principally
derable engineering
ment.

---



e.g. “Breaking the Curve25519 function—for example, computing the shared secret from the two public keys—is conjectured to be extremely difficult. Every known attack is more expensive than performing a brute-force search on a typical 128-bit secret-key cipher. … Curves of this shape have order divisible by 4, requiring a marginally larger prime for the same conjectured security level, but this is outweighed by the extra speed of curve operations.”

Reviewe

---

... The
hardwir
which l
if chan
... My
paper a
as low
mostly
very st
therefo
The pap

luding protection
ttacks, no
 infringements,
eed. ...

side, the paper
ce novel ideas,
mpt to prove
y (the word
used repeatedly
is principally
ngineering



e.g. "Breaking the Curve25519 function—for example, computing the shared secret from the two public keys—is conjectured to be extremely difficult. Every known attack is more expensive than performing a brute-force search on a typical 128-bit secret-key cipher. ... Curves of this shape have order divisible by 4, requiring a marginally larger prime for the same conjectured security level, but this is outweighed by the extra speed of curve operations."

Reviewer #3:

... The curve an
hardwired into t
which leaves lit
if changes are s
... My main conc
paper are that i
as low on useful
mostly about one
very strangely w
therefore unplea
The paper is wri

rotection
o
ements,

paper
ideas,
rove
rd
eatedly
pally
g



e.g. "Breaking the Curve25519 function—for example, computing the shared secret from the two public keys—is conjectured to be extremely difficult. Every known attack is more expensive than performing a brute-force search on a typical 128-bit secret-key cipher. . . . Curves of this shape have order divisible by 4, requiring a marginally larger prime for the same conjectured security level, but this is outweighed by the extra speed of curve operations."

Reviewer #3:

```
... The curve and the fie
hardwired into the progra
which leaves little flexi
if changes are someday ne
... My main concerns abou
paper are that it comes a
as low on useful content
mostly about one curve),
very strangely written, a
therefore unpleasant to r
The paper is written in w
```

e.g. “Breaking the Curve25519 function—for example, computing the shared secret from the two public keys—is conjectured to be extremely difficult. Every known attack is more expensive than performing a brute-force search on a typical 128-bit secret-key cipher. . . . Curves of this shape have order divisible by 4, requiring a marginally larger prime for the same conjectured security level, but this is outweighed by the extra speed of curve operations.”



Reviewer #3:

---

... The curve and the field are hardwired into the program, which leaves little flexibility if changes are someday needed. ... My main concerns about the paper are that it comes across as low on useful content (it’s mostly about one curve), and is very strangely written, and therefore unpleasant to read ... The paper is written in what

eaking the Curve25519
—for example, computing
ed secret from the two
eys—is conjectured to be
ly difficult. Every known
s more expensive than
ng a brute-force search
ical 128-bit secret-key
. . Curves of this shape
ler divisible by 4, requiring
nally larger prime for the
njectured security level,
is outweighed by the
eed of curve operations."



Reviewer #3:

---

```
... The curve and the field are
hardwired into the program,
which leaves little flexibility
if changes are someday needed.
... My main concerns about the
paper are that it comes across
as low on useful content (it's
mostly about one curve), and is
very strangely written, and
therefore unpleasant to read ...
The paper is written in what
```

```
comes ac
incohere
rewritin
to make
signific
someone
I'm not
be done
the cont
"results
stated i
trivial
signific
```

e Curve25519
mple, computing
from the two
njectured to be
. Every known
pensive than
e-force search
it secret-key
of this shape
e by 4, requiring
r prime for the
security level,
ghed by the
ve operations."



Reviewer #3:

---

`... The curve and the field are hardwired into the program, which leaves little flexibility if changes are someday needed. ... My main concerns about the paper are that it comes across as low on useful content (it's mostly about one curve), and is very strangely written, and therefore unpleasant to read ... The paper is written in what`

comes across as
incoherent style
rewriting that w
to make this pap
significant (tho
someone willing
I'm not optimist
be done by the d
the content (I c
"results" since
stated results,
trivial mathemat
significant enou

519
puting
wo
to be
nown
an
rch
ey
ape
quiring
the
vel,
e
ons.”



Reviewer #3:

---

... The curve and the field are hardwired into the program, which leaves little flexibility if changes are someday needed. ... My main concerns about the paper are that it comes across as low on useful content (it’s mostly about one curve), and is very strangely written, and therefore unpleasant to read ... The paper is written in what

comes across as a ramblin
incoherent style. ... The
rewriting that would be r
to make this paper readab
significant (though easy
someone willing to do it)
I’m not optimistic that i
be done by the deadline,
the content (I can’t say
"results" since there are
stated results, other tha
trivial mathematical resu
significant enough to jus

# Reviewer #3:

---

... The curve and the field are hardwired into the program, which leaves little flexibility if changes are someday needed. ... My main concerns about the paper are that it comes across as low on useful content (it's mostly about one curve), and is very strangely written, and therefore unpleasant to read ... The paper is written in what



comes across as a rambling incoherent style. ... The rewriting that would be required to make this paper readable is significant (though easy for someone willing to do it), and I'm not optimistic that it would be done by the deadline, or that the content (I can't say "results" since there aren't any stated results, other than a trivial mathematical result) is significant enough to justify

r #3:

curve and the field are
ed into the program,
eaves little flexibility
ges are someday needed.
main concerns about the
re that it comes across
on useful content (it's
about one curve), and is
rangely written, and
re unpleasant to read ...
er is written in what



comes across as a rambling incoherent style. ... The rewriting that would be required to make this paper readable is significant (though easy for someone willing to do it), and I'm not optimistic that it would be done by the deadline, or that the content (I can't say "results" since there aren't any stated results, other than a trivial mathematical result) is significant enough to justify

acceptan
Curve255
section
there's
in it,
clear.
appendi
For exam
discuss
either
to be a
discuss
discuss

d the field are
he program,
tle flexibility
omeday needed.
erns about the
t comes across
content (it's
curve), and is
ritten, and
sant to read ...
tten in what



comes across as a rambling incoherent style. ... The rewriting that would be required to make this paper readable is significant (though easy for someone willing to do it), and I'm not optimistic that it would be done by the deadline, or that the content (I can't say "results" since there aren't any stated results, other than a trivial mathematical result) is significant enough to justify

acceptance. ...
Curve25519 secur
section should b
there's useful a
in it, that shou
clear. ... Most
appendices shoul
For example, the
discussion of pa
either be remove
to be a purely s
discussion and n
discussion, and

eld are
m,
bility
eeded.
t the
cross
(it's
and is
nd
ead ...
hat



comes across as a rambling incoherent style. ... The rewriting that would be required to make this paper readable is significant (though easy for someone willing to do it), and I'm not optimistic that it would be done by the deadline, or that the content (I can't say "results" since there aren't any stated results, other than a trivial mathematical result) is significant enough to justify

acceptance. ... The "Conj
Curve25519 security level
section should be omitted
there's useful and new co
in it, that should be mad
clear. ... Most of the
appendices should be remo
For example, the irreleva
discussion of patents sho
either be removed, or rep
to be a purely scientific
discussion and not a pate
discussion, and the appen

comes across as a rambling incoherent style. ... The rewriting that would be required to make this paper readable is significant (though easy for someone willing to do it), and I'm not optimistic that it would be done by the deadline, or that the content (I can't say "results" since there aren't any stated results, other than a trivial mathematical result) is significant enough to justify



acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there's useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix

cross as a rambling
ent style. ... The
ng that would be required
this paper readable is
cant (though easy for
willing to do it), and
optimistic that it would
by the deadline, or that
tent (I can't say
s" since there aren't any
results, other than a
mathematical result) is
cant enough to justify



acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there's useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix

that sh
prime s
The pap
interes
Diffie-
curves.
the exp
y-coord
being u
ECC pro

26

a rambling
. ... The
ould be required
er readable is
ugh easy for
to do it), and
ic that it would
eadline, or that
an't say
there aren't any
other than a
ical result) is
gh to justify

27

acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there's useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix

that shows that
prime should be
The paper will b
interest to thos
Diffie-Hellman w
curves. But the
the exponent (an
y-coordinate) pr
being used by El
ECC protocols. .

g
e
required
ble is
for
, and
t would
or that

n't any
n a
lt) is
tify



acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there's useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix

that shows that 3 numbers
prime should be removed.
The paper will be of grea
interest to those impleme
Diffie-Hellman with ellip
curves. But the limitatio
the exponent (and the lac
y-coordinate) prevent it
being used by El Gamal an
ECC protocols. ...

acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there's useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix



that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ...

acceptance. ... The "Conjectured Curve25519 security level" section should be omitted; or if there’s useful and new content in it, that should be made clear. ... Most of the appendices should be removed. For example, the irrelevant discussion of patents should either be removed, or rephrased to be a purely scientific discussion and not a patent discussion, and the appendix



that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ... The paper is remarkably free of grammatical errors.

---

nce. ... The "Conjectured
519 security level"
should be omitted; or if
useful and new content
that should be made
... Most of the
ces should be removed.
mple, the irrelevant
ion of patents should
be removed, or rephrased
purely scientific
ion and not a patent
ion, and the appendix



that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ... The paper is remarkably free of grammatical errors.

---

2016: C

The "Conjectured
ity level"
e omitted; or if
nd new content
ld be made
of the
d be removed.
 irrelevant
tents should
d, or rephrased
cientific
ot a patent
the appendix



that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ... The paper is remarkably free of grammatical errors.

---

2016: Counterfeit

ectured
"
; or if
ntent
e

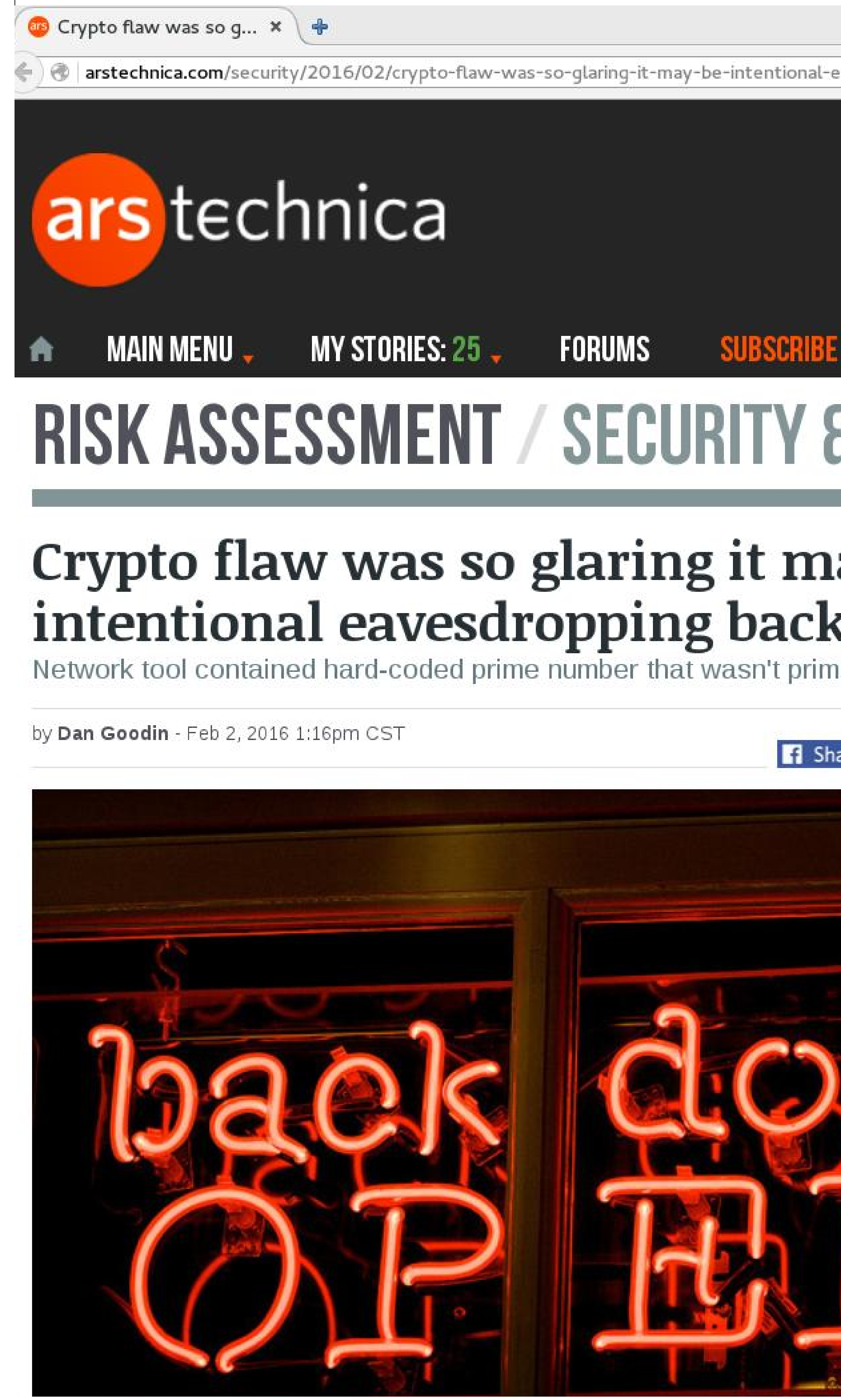
ved.
nt
uld
hrased
:
nt
dix



that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ... The paper is remarkably free of grammatical errors.

---

# 2016: Counterfeit “primes”.

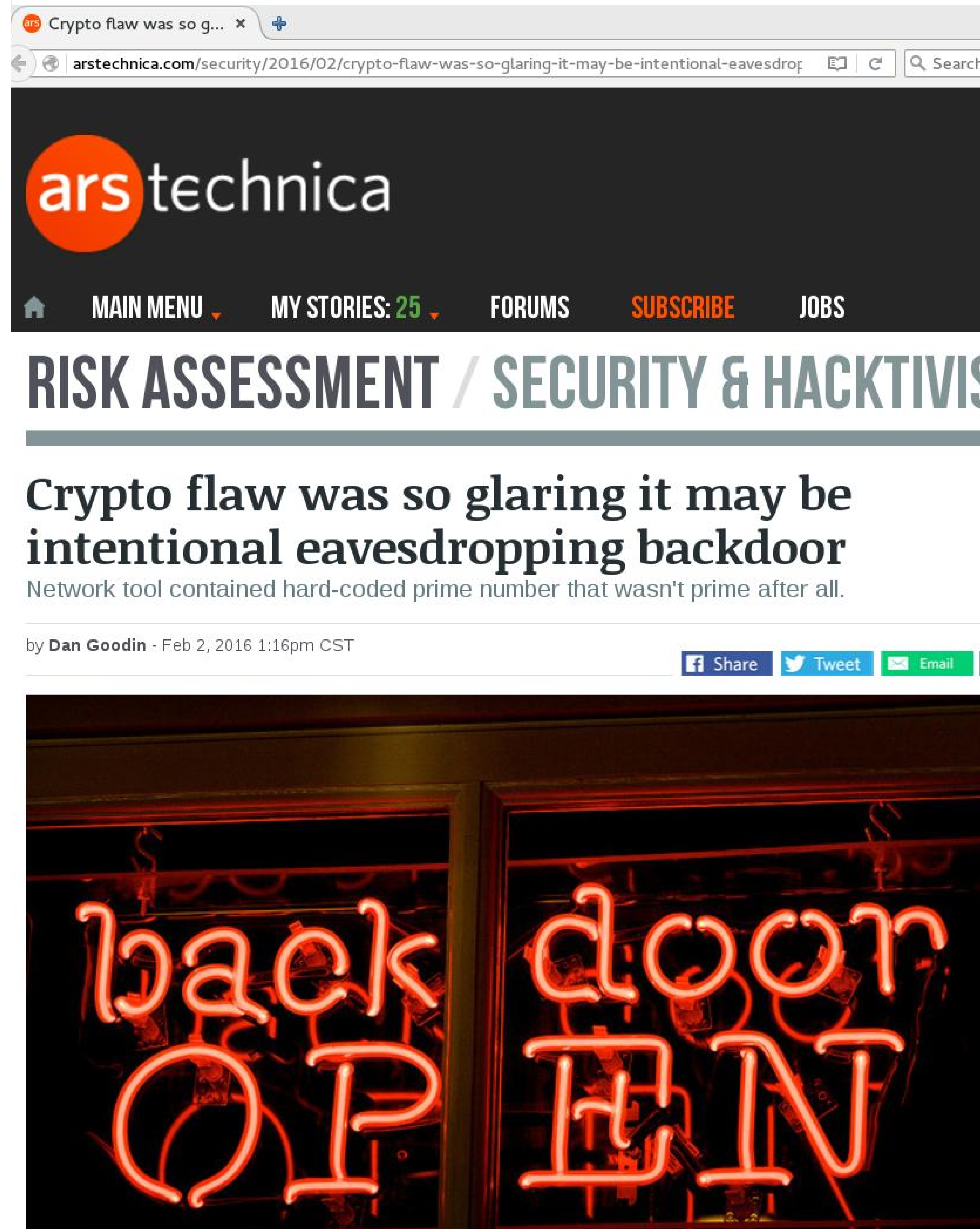


that shows that 3 numbers are prime should be removed. ... The paper will be of greatest interest to those implementing Diffie-Hellman with elliptic curves. But the limitations on the exponent (and the lack of a y-coordinate) prevent it from being used by El Gamal and other ECC protocols. ... The paper is remarkably free of grammatical errors.

---



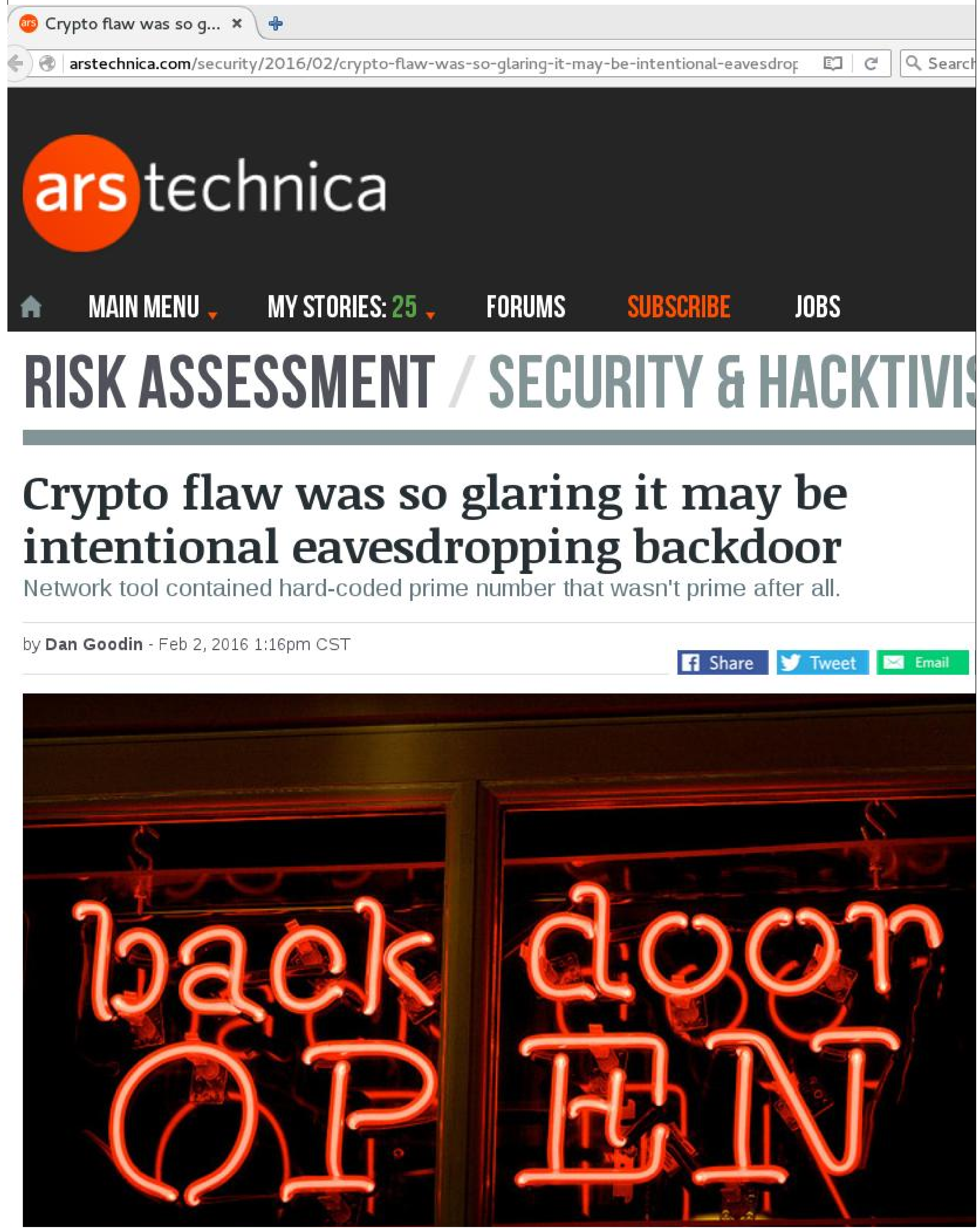
# 2016: Counterfeit “primes”.



ows that 3 numbers are
hould be removed. ...
er will be of greatest
t to those implementing
Hellman with elliptic
 But the limitations on
onent (and the lack of a
inate) prevent it from
sed by El Gamal and other
tocols. ... The paper is
bly free of grammatical



2016: Counterfeit “primes”.

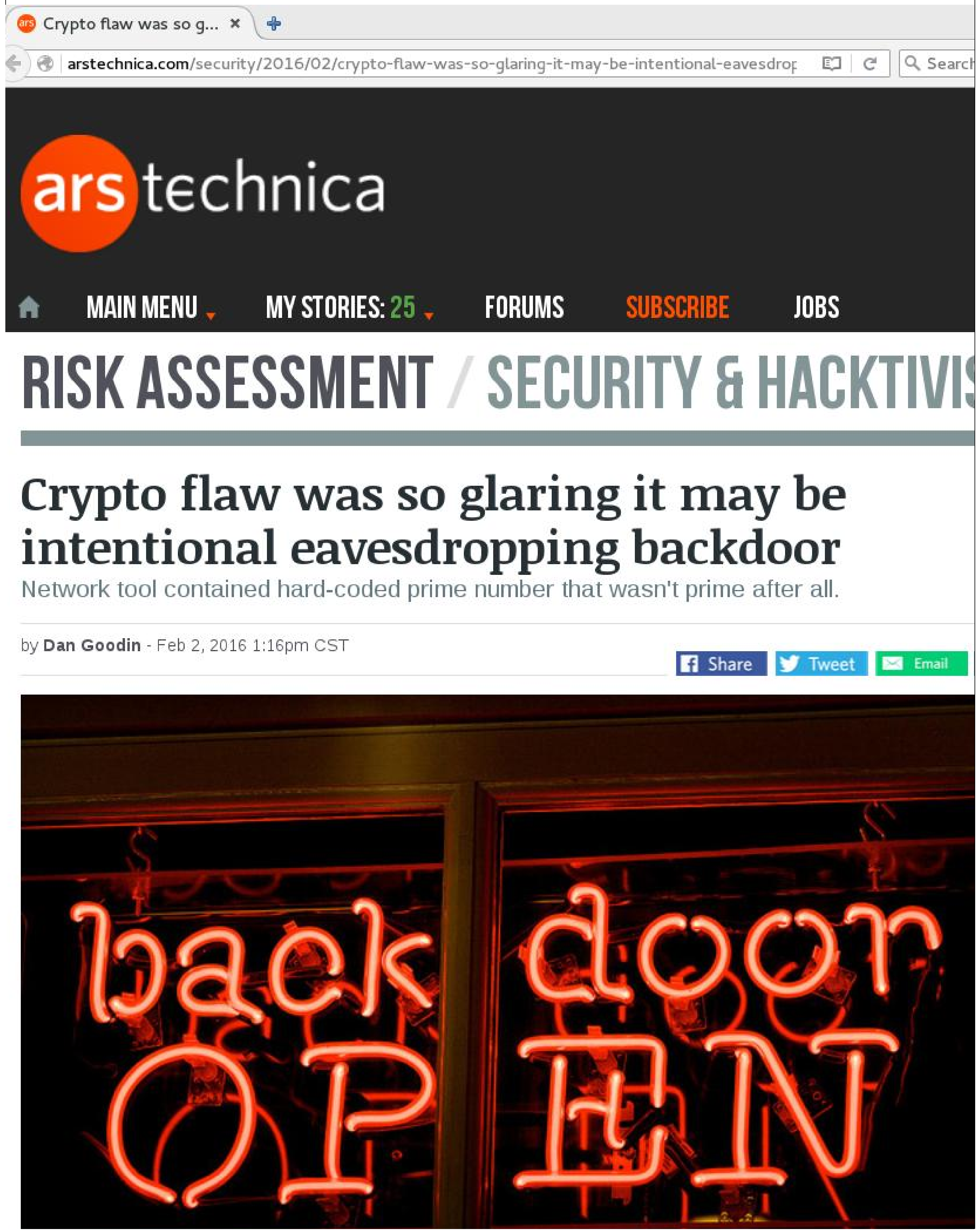



With rev
how did

3 numbers are
removed. ...
e of greatest
e implementing
ith elliptic
limitations on
d the lack of a
event it from
Gamal and other
.. The paper is
of grammatical



2016: Counterfeit “primes”.



Crypto flaw was so g... ×
arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdrop
Search

ars technica

MAIN MENU MY STORIES: 25 FORUMS SUBSCRIBE JOBS

RISK ASSESSMENT / SECURITY & HACKTIVIS

**Crypto flaw was so glaring it may be intentional eavesdropping backdoor**

Network tool contained hard-coded prime number that wasn't prime after all.

by **Dan Goodin** - Feb 2, 2016 1:16pm CST

Share Tweet Email

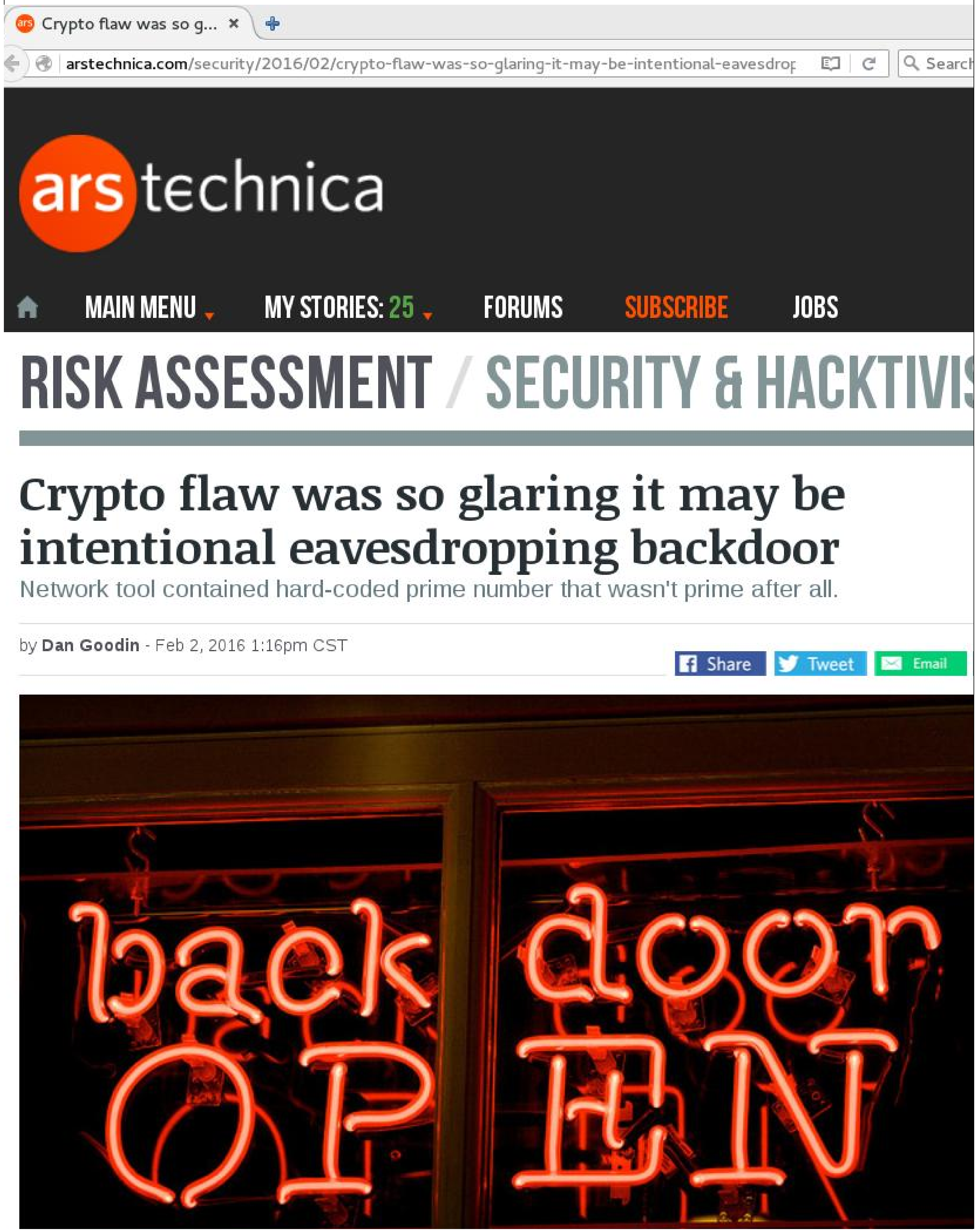


With reviews like t
how did PKC acce

are
...
test
nting
tic
ns on
k of a
from
d other
per is
tical



2016: Counterfeit “primes”.


Crypto flaw was so g...
arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdrop
ars technica
MAIN MENU MY STORIES: 25 FORUMS SUBSCRIBE JOBS
RISK ASSESSMENT / SECURITY & HACKTIVIS

Crypto flaw was so glaring it may be intentional eavesdropping backdoor

Network tool contained hard-coded prime number that wasn't prime after all.

by Dan Goodin - Feb 2, 2016 1:16pm CST

Share Tweet Email



With reviews like these,
how did PKC accept Curve2

2016: Counterfeit “primes”.




With reviews like these,

how did PKC accept Curve25519?

# 2016: Counterfeit “primes”.




With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced
other people as part of discussion.
Or program chairs liked paper.

2016: Counterfeit “primes”.

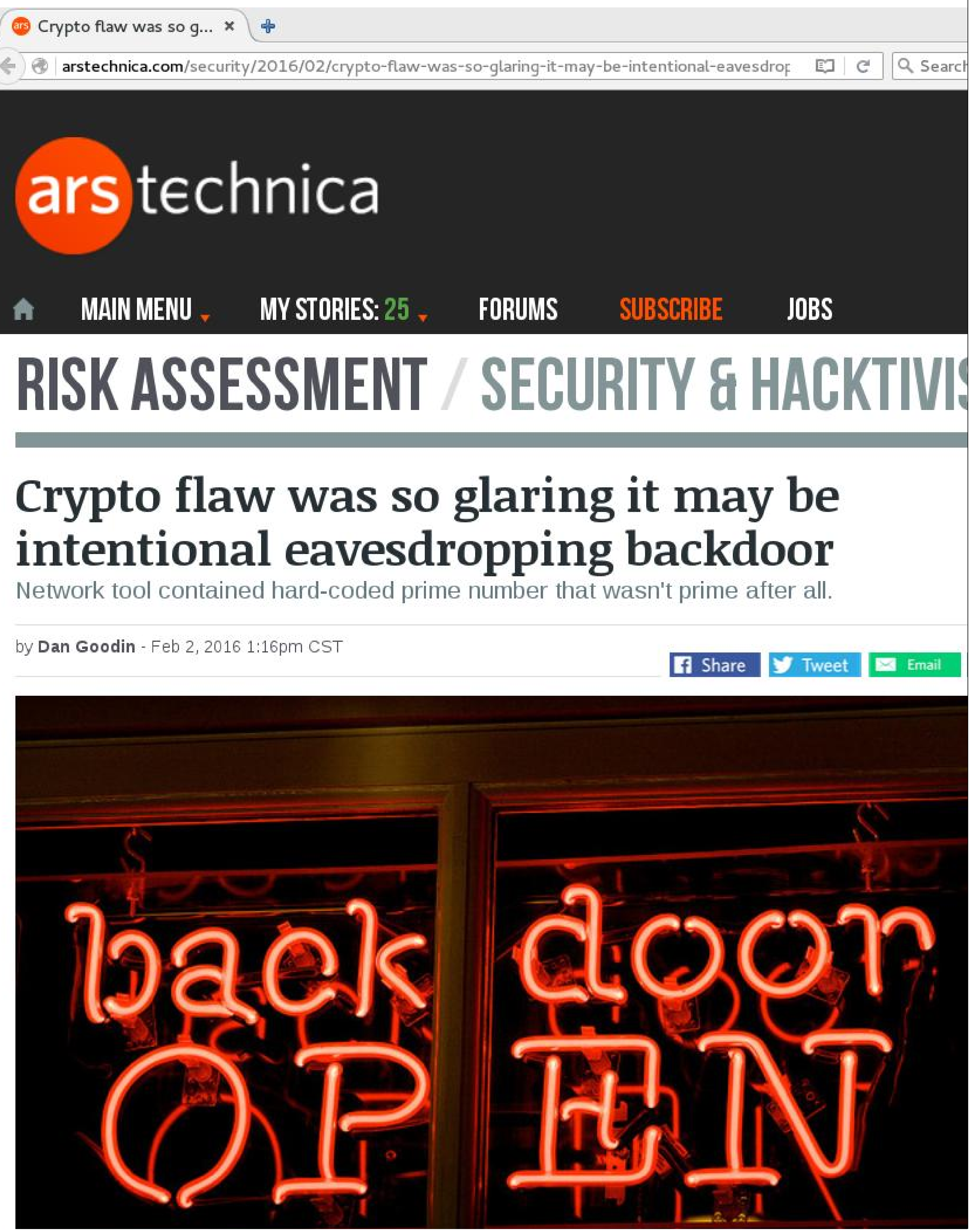



With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced
other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title
“9th International Conference on
Theory **and Practice** in Public-
Key Cryptography” justified
an occasional paper like this.

# 2016: Counterfeit “primes”.





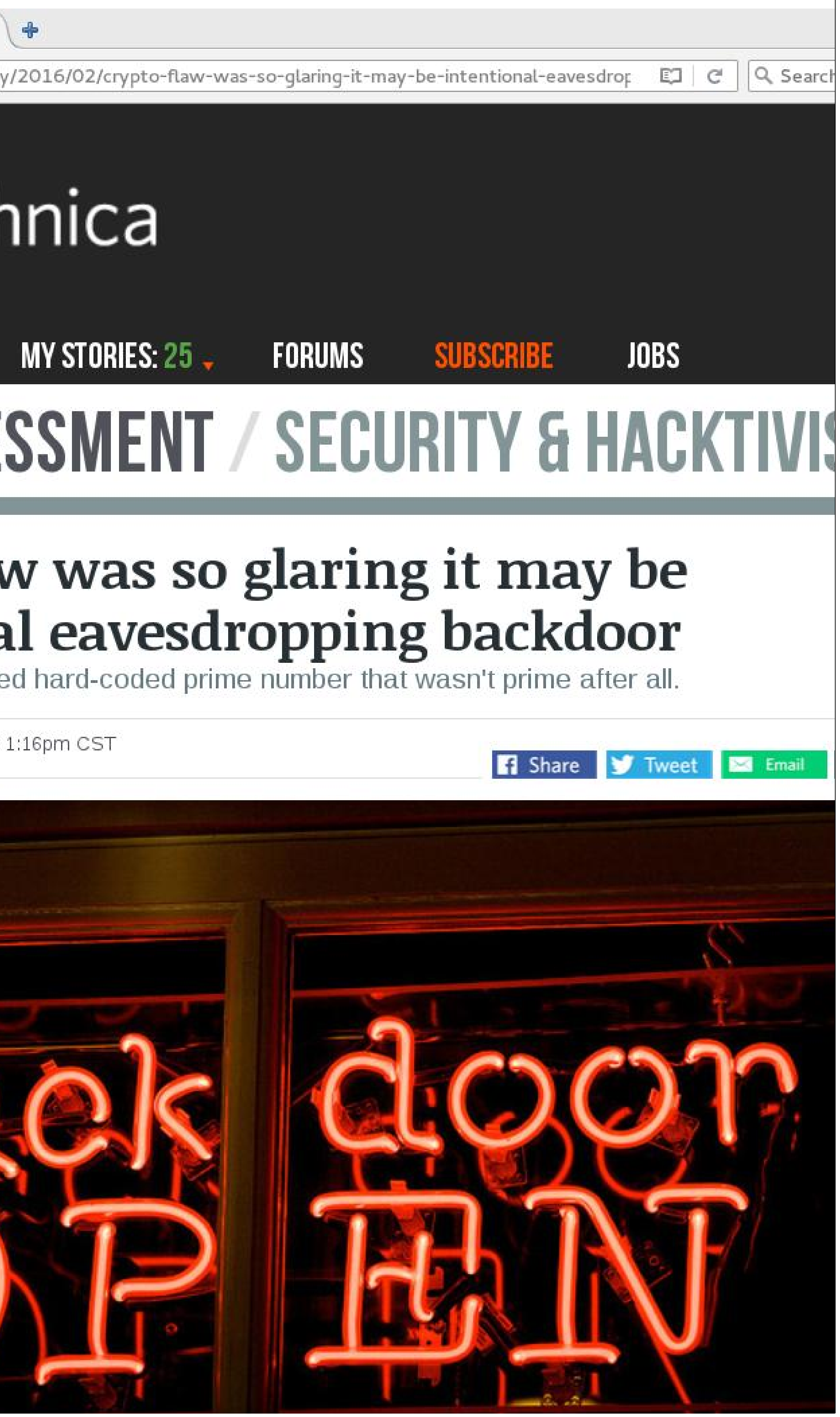

With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced
other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title
“9th International Conference on
Theory **and Practice** in Public-
Key Cryptography” justified
an occasional paper like this.

Note to young cryptographers:
Don’t let referees discourage you.

ounterfeit “primes”.






With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced
other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title
“9th International Conference on
Theory **and Practice** in Public-
Key Cryptography” justified
an occasional paper like this.

Note to young cryptographers:
Don’t let referees discourage you.

Edwards

2007 Ed
normal f

$x_3 =$

$y_3 =$

generica
$(x_1, y_1)$
on any e
$x^2 + y^2$

Euler+G
for one c

"primes".




With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced
other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title
"9th International Conference on
Theory **and Practice** in Public-
Key Cryptography" justified
an occasional paper like this.

Note to young cryptographers:
Don't let referees discourage you.

Edwards curves

2007 Edwards "A
normal form for el

$$x_3 = \frac{x_1 y_2 \cdots}{c(1 + x \cdots}$$

$$y_3 = \frac{y_1 y_2 \cdots}{c(1 - x \cdots}$$

generically defines
$(x_1, y_1) + (x_2, y_2)$
on any elliptic cur
$x^2 + y^2 = c^2(1 + \cdots$

Euler+Gauss defin
for one curve: $c^4 \cdots$




With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title "9th International Conference on Theory **and Practice** in Public-Key Cryptography" justified an occasional paper like this.

Note to young cryptographers:
Don't let referees discourage you.



Edwards curves

2007 Edwards "A normal form for elliptic curv

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition l
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
on any elliptic curve of the f
$x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this lav
for one curve: $c^4 = -1$.

With reviews like these,
how did PKC accept Curve25519?

Reviewer #4 was positive.
Maybe reviewer #4 convinced other people as part of discussion.
Or program chairs liked paper.

Maybe someone thought the title "9th International Conference on Theory **and Practice** in Public-Key Cryptography" justified an occasional paper like this.

Note to young cryptographers:
Don't let referees discourage you.



## Edwards curves

2007 Edwards "A normal form for elliptic curves":

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.

views like these,
PKC accept Curve25519?

r #4 was positive.
eviewer #4 convinced
ople as part of discussion.
ram chairs liked paper.

someone thought the title
ernational Conference on
**and Practice** in Public-
ptography" justified
sional paper like this.

young cryptographers:
t referees discourage you.



Edwards curves

2007 Edwards "A normal form for elliptic curves":

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.

2007 Be
addition
curves":
easily ge

$x_3$ =

$y_3$ =

on any e
$x^2 + y^2$

$d = c^4$ i
$d = 0$ is

these,
ept Curve25519?

positive.
4 convinced
rt of discussion.
liked paper.

hought the title
Conference on
**tice** in Public-
” justified
er like this.

ptographers:
discourage you.



## Edwards curves

2007 Edwards “A normal form for elliptic curves”:

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.

2007 Bernstein–La
addition and doub
curves”: Edwards
easily generalizes t

$$x_3 = \frac{x_1 y_2 +}{1 + d x_1}$$

$$y_3 = \frac{y_1 y_2 -}{1 - d x_1}$$

on any elliptic cur
$x^2 + y^2 = 1 + d x$

$d = c^4$ is original

$d = 0$ is circle, no

25519?

ed
ussion.
er.

e title
ce on
blic-

s.

rs:
e you.

Edwards curves

2007 Edwards "A normal form for elliptic curves":

$$x_3 = \frac{x_1y_2 + x_2y_1}{c(1 + x_1x_2y_1y_2)},$$

$$y_3 = \frac{y_1y_2 - x_1x_2}{c(1 - x_1x_2y_1y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.



2007 Bernstein–Lange "Fast addition and doubling on ell curves": Edwards addition l easily generalizes to

$$x_3 = \frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2},$$

$$y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

on any elliptic curve of the f $x^2 + y^2 = 1 + dx^2y^2$.

$d = c^4$ is original Edwards.

$d = 0$ is circle, non-elliptic.

## Edwards curves

2007 Edwards "A normal form for elliptic curves":

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.



2007 Bernstein–Lange "Faster addition and doubling on elliptic curves": Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + d x^2 y^2$.

$d = c^4$ is original Edwards.

$d = 0$ is circle, non-elliptic.

## Edwards curves

2007 Edwards "A normal form for elliptic curves":

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

generically defines addition law $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ on any elliptic curve of the form $x^2 + y^2 = c^2(1 + x^2 y^2)$.

Euler+Gauss defined this law for one curve: $c^4 = -1$.



2007 Bernstein–Lange "Faster addition and doubling on elliptic curves": Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + d x^2 y^2$.

$d = c^4$ is original Edwards.
$d = 0$ is circle, non-elliptic.
Surprise for non-square $d$: this addition law is **complete**!

curves

wards “A

orm for elliptic curves”:

$$= \frac{x_1 y_2 + x_2 y_1}{c(1 + x_1 x_2 y_1 y_2)},$$

$$= \frac{y_1 y_2 - x_1 x_2}{c(1 - x_1 x_2 y_1 y_2)}$$

lly defines addition law

$+ (x_2, y_2) = (x_3, y_3)$

elliptic curve of the form

$= c^2(1 + x^2 y^2)$.

auss defined this law

curve: $c^4 = -1$.



2007 Bernstein–Lange “Faster addition and doubling on elliptic curves”: Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + d x^2 y^2$.

$d = c^4$ is original Edwards.

$d = 0$ is circle, non-elliptic.

Surprise for non-square $d$:

this addition law is **complete**!

By easy

can writ

with nor

as a com

In partic

liptic curves":

$+ x_2 y_1$

$_1 x_2 y_1 y_2)$,

$- x_1 x_2$

$_1 x_2 y_1 y_2)$

addition law

$= (x_3, y_3)$

ve of the form

$x^2 y^2)$.

ned this law

$= -1$.



2007 Bernstein–Lange "Faster addition and doubling on elliptic curves": Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + d x^2 y^2$.

$d = c^4$ is original Edwards.

$d = 0$ is circle, non-elliptic.

Surprise for non-square $d$: this addition law is **complete**!

By easy change of

can write $y^2 = x^3$

with non-square $A$

as a complete Edw

In particular: Curv

es”:

aw

form

w



2007 Bernstein–Lange “Faster addition and doubling on elliptic curves”: Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + dx^2y^2$.

$d = c^4$ is original Edwards.

$d = 0$ is circle, non-elliptic.

Surprise for non-square $d$: this addition law is **complete**!



By easy change of coordinat

can write $y^2 = x^3 + Ax^2 +$

with non-square $A^2 - 4$

as a complete Edwards curv

In particular: Curve25519.

2007 Bernstein–Lange "Faster addition and doubling on elliptic curves": Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + dx^2y^2$.

$d = c^4$ is original Edwards.
$d = 0$ is circle, non-elliptic.
Surprise for non-square $d$: this addition law is **complete**!



By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve.
In particular: Curve25519.

2007 Bernstein–Lange "Faster addition and doubling on elliptic curves": Edwards addition law easily generalizes to

$$x_3 = \frac{x_1 y_2 + x_2 y_1}{1 + d x_1 x_2 y_1 y_2},$$

$$y_3 = \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}.$$

on any elliptic curve of the form $x^2 + y^2 = 1 + dx^2y^2$.

$d = c^4$ is original Edwards.
$d = 0$ is circle, non-elliptic.
Surprise for non-square $d$: this addition law is **complete**!



By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve.
In particular: Curve25519.

Curve arithmetic is very fast.

(After various followup papers: even faster!)

Almost as fast as Montgomery for $n, P \mapsto nP$ in DH.

New speed records for $m, n, P, Q \mapsto mP + nQ$ and other signature operations.

rnstein–Lange "Faster
and doubling on elliptic

Edwards addition law
eneralizes to

$$= \frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2},$$

$$= \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

elliptic curve of the form
$= 1 + dx^2y^2$.

s original Edwards.
circle, non-elliptic.
for non-square $d$:
ition law is **complete**!



By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve. In particular: Curve25519.

Curve arithmetic is very fast.

(After various followup papers: even faster!)

Almost as fast as Montgomery for $n, P \mapsto nP$ in DH.

New speed records for $m, n, P, Q \mapsto mP + nQ$ and other signature operations.

The Ed2

CHES 2
Lange–S

Start fro
Skip sign
Support
Use dou
include p
$SB = R$

Generate
as a secr
$\Rightarrow$ Avoic

Use Cur
"$-$1-twis

ange "Faster
ling on elliptic
addition law
to

$$\frac{\cdots x_2 y_1}{\cdots x_2 y_1 y_2},$$

$$\frac{\cdots x_1 x_2}{\cdots x_2 y_1 y_2}.$$

ve of the form
$^2y^2$.

Edwards.
n-elliptic.
quare $d$:
s **complete**!



By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve. In particular: Curve25519.

Curve arithmetic is very fast.

(After various followup papers: even faster!)

Almost as fast as Montgomery for $n, P \mapsto nP$ in DH.

New speed records for $m, n, P, Q \mapsto mP + nQ$ and other signature operations.



The Ed25519 sign

CHES 2011 Bernst
Lange–Schwabe–Y

Start from Schnor
Skip signature con
Support batch ver
Use double-size $H$
include public key
$SB = R + H(R, A$

Generate $R$ detern
as a secret hash o
$\Rightarrow$ Avoid PlayStat

Use Curve25519 in
"$-1$-twisted" Edw

ter
iptic
aw

form

te!



By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve. In particular: Curve25519.

Curve arithmetic is very fast.

(After various followup papers: even faster!)

Almost as fast as Montgomery for $n, P \mapsto nP$ in DH.

New speed records for $m, n, P, Q \mapsto mP + nQ$ and other signature operations.



The Ed25519 signature syst

CHES 2011 Bernstein–Duif–
Lange–Schwabe–Yang:

Start from Schnorr signature
Skip signature compression.
Support batch verification.
Use double-size $H$ output, a
include public key $A$ as inpu
$SB = R + H(R, A, M)A$.

Generate $R$ deterministically
as a secret hash of $M$.
$\Rightarrow$ Avoid PlayStation disaste

Use Curve25519 in complete
"$-$1-twisted" Edwards form.

By easy change of coordinates can write $y^2 = x^3 + Ax^2 + x$ with non-square $A^2 - 4$ as a complete Edwards curve. In particular: Curve25519.

Curve arithmetic is very fast.

(After various followup papers: even faster!)

Almost as fast as Montgomery for $n, P \mapsto nP$ in DH.

New speed records for $m, n, P, Q \mapsto mP + nQ$ and other signature operations.



## The Ed25519 signature system

CHES 2011 Bernstein–Duif–Lange–Schwabe–Yang:

Start from Schnorr signatures. Skip signature compression. Support batch verification. Use double-size $H$ output, and include public key $A$ as input: $SB = R + H(R, A, M)A$.

Generate $R$ deterministically as a secret hash of $M$. $\Rightarrow$ Avoid PlayStation disaster.

Use Curve25519 in complete "$-1$-twisted" Edwards form.

change of coordinates
e $y^2 = x^3 + Ax^2 + x$
n-square $A^2 - 4$
nplete Edwards curve.
cular: Curve25519.

rithmetic is very fast.

arious followup papers:
ter!)

as fast as Montgomery
$\mapsto nP$ in DH.

eed records for
$Q \mapsto mP + nQ$
er signature operations.



## The Ed25519 signature system

CHES 2011 Bernstein–Duif–Lange–Schwabe–Yang:

Start from Schnorr signatures.
Skip signature compression.
Support batch verification.
Use double-size $H$ output, and include public key $A$ as input:
$SB = R + H(R, A, M)A$.

Generate $R$ deterministically as a secret hash of $M$.
$\Rightarrow$ Avoid PlayStation disaster.

Use Curve25519 in complete "$-1$-twisted" Edwards form.

## Optimiz

2007 Ga
2009 Co
2011 Be
Schwabe
2012 Be
2014 La
2014 Ma
2014 Sa
2015 Ch
2015 Dü
Hutter–
microco
2015 Hu
Wieser:

f coordinates

$+ Ax^2 + x$

$^2 - 4$

vards curve.

ve25519.

s very fast.

owup papers:

Montgomery

DH.

s for

$+ nQ$

re operations.



## The Ed25519 signature system

CHES 2011 Bernstein–Duif–Lange–Schwabe–Yang:

Start from Schnorr signatures.
Skip signature compression.
Support batch verification.
Use double-size $H$ output, and include public key $A$ as input:
$SB = R + H(R, A, M)A$.

Generate $R$ deterministically as a secret hash of $M$.
$\Rightarrow$ Avoid PlayStation disaster.

Use Curve25519 in complete "$-1$-twisted" Edwards form.

## Optimizations for

2007 Gaudry–Tho

2009 Costigan–Sc

2011 Bernstein–D

Schwabe–Yang: N

2012 Bernstein–Sc

2014 Langley–Mo

2014 Mahé–Chau

2014 Sasdrich–Gü

2015 Chou: newer

2015 Düll–Haase–

Hutter–Paar–Sánc

microcontrollers.

2015 Hutter-Schill

Wieser: ASICs.

tes
x
e.
.
ers:
ery
ons.



## The Ed25519 signature system

CHES 2011 Bernstein–Duif–Lange–Schwabe–Yang:

Start from Schnorr signatures.
Skip signature compression.
Support batch verification.
Use double-size $H$ output, and include public key $A$ as input:
$SB = R + H(R, A, M)A$.

Generate $R$ deterministically as a secret hash of $M$.
$\Rightarrow$ Avoid PlayStation disaster.

Use Curve25519 in complete "$-1$-twisted" Edwards form.



## Optimizations for more platf

2007 Gaudry–Thomé: Core
2009 Costigan–Schwabe: Ce
2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.
2012 Bernstein–Schwabe: N
2014 Langley–Moon: newer
2014 Mahé–Chauvet: GPUs
2014 Sasdrich–Güneysu: FP
2015 Chou: newer Intel.
2015 Düll–Haase–Hinterwäl
Hutter–Paar–Sánchez–Schw
microcontrollers.
2015 Hutter-Schilling–Schwa
Wieser: ASICs.

## The Ed25519 signature system

CHES 2011 Bernstein–Duif–Lange–Schwabe–Yang:

Start from Schnorr signatures.
Skip signature compression.
Support batch verification.
Use double-size $H$ output, and include public key $A$ as input:
$SB = R + H(R, A, M)A$.

Generate $R$ deterministically as a secret hash of $M$.
$\Rightarrow$ Avoid PlayStation disaster.

Use Curve25519 in complete "$-1$-twisted" Edwards form.



## Optimizations for more platforms

2007 Gaudry–Thomé: Core 2.
2009 Costigan–Schwabe: Cell.
2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.
2012 Bernstein–Schwabe: NEON.
2014 Langley–Moon: newer Intel.
2014 Mahé–Chauvet: GPUs.
2014 Sasdrich–Güneysu: FPGAs.
2015 Chou: newer Intel.
2015 Düll–Haase–Hinterwälder–Hutter–Paar–Sánchez–Schwabe: microcontrollers.
2015 Hutter-Schilling–Schwabe–Wieser: ASICs.

25519 signature system

011 Bernstein–Duif–
Schwabe–Yang:

om Schnorr signatures.
nature compression.
batch verification.
ble-size $H$ output, and
public key $A$ as input:
$+ H(R, A, M)A$.

e $R$ deterministically
ret hash of $M$.
d PlayStation disaster.

ve25519 in complete
sted" Edwards form.



Optimizations for more platforms

2007 Gaudry–Thomé: Core 2.
2009 Costigan–Schwabe: Cell.
2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.
2012 Bernstein–Schwabe: NEON.
2014 Langley–Moon: newer Intel.
2014 Mahé–Chauvet: GPUs.
2014 Sasdrich–Güneysu: FPGAs.
2015 Chou: newer Intel.
2015 Düll–Haase–Hinterwälder–Hutter–Paar–Sánchez–Schwabe: microcontrollers.
2015 Hutter-Schilling–Schwabe–Wieser: ASICs.

Next-gen

NaCl: N
Cryptog
very sim
key auth

All-in-on
uses Cur
Salsa20
Poly130

More on
2011 Be
"The sec
new cryp

ature system

tein–Duif–
ang:

r signatures.
npression.
ification.
output, and
$A$ as input:
$, M)A$.

ministically
f $M$.
ion disaster.

n complete
ards form.



## Optimizations for more platforms

2007 Gaudry–Thomé: Core 2.
2009 Costigan–Schwabe: Cell.
2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.
2012 Bernstein–Schwabe: NEON.
2014 Langley–Moon: newer Intel.
2014 Mahé–Chauvet: GPUs.
2014 Sasdrich–Güneysu: FPGAs.
2015 Chou: newer Intel.
2015 Düll–Haase–Hinterwälder–Hutter–Paar–Sánchez–Schwabe: microcontrollers.
2015 Hutter-Schilling–Schwabe–Wieser: ASICs.

## Next-generation cr

NaCl: Networking
Cryptography libra
very simple new A
key authenticated

All-in-one crypto
uses Curve25519 f
Salsa20 for encryp
Poly1305 for authe

More on NaCl des
2011 Bernstein–La
"The security imp
new cryptographic

em

-

es.

nd

t:

y

er.

e

.



## Optimizations for more platforms

2007 Gaudry–Thomé: Core 2.

2009 Costigan–Schwabe: Cell.

2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.

2012 Bernstein–Schwabe: NEON.

2014 Langley–Moon: newer Intel.

2014 Mahé–Chauvet: GPUs.

2014 Sasdrich–Güneysu: FPGAs.

2015 Chou: newer Intel.

2015 Düll–Haase–Hinterwälder–Hutter–Paar–Sánchez–Schwabe: microcontrollers.

2015 Hutter-Schilling–Schwabe–Wieser: ASICs.

## Next-generation crypto libra

NaCl: Networking and Cryptography library provide very simple new API for pub key authenticated encryptio

All-in-one `crypto_box` func uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schw "The security impact of a new cryptographic library".

## Optimizations for more platforms

2007 Gaudry–Thomé: Core 2.
2009 Costigan–Schwabe: Cell.
2011 Bernstein–Duif–Lange–Schwabe–Yang: Nehalem.
2012 Bernstein–Schwabe: NEON.
2014 Langley–Moon: newer Intel.
2014 Mahé–Chauvet: GPUs.
2014 Sasdrich–Güneysu: FPGAs.
2015 Chou: newer Intel.
2015 Düll–Haase–Hinterwälder–Hutter–Paar–Sánchez–Schwabe: microcontrollers.
2015 Hutter-Schilling–Schwabe–Wieser: ASICs.



## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".

ations for more platforms

udry–Thomé: Core 2.
stigan–Schwabe: Cell.
rnstein–Duif–Lange–
e–Yang: Nehalem.
rnstein–Schwabe: NEON.
ngley–Moon: newer Intel.
ahé–Chauvet: GPUs.
sdrich–Güneysu: FPGAs.
ou: newer Intel.
ill–Haase–Hinterwälder–
Paar–Sánchez–Schwabe:
ntrollers.
tter-Schilling–Schwabe–
ASICs.



Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".

Simplicit

Curve25
advertise

2013 Be
Lange–S
reimplen
tweets.

more platforms

mé: Core 2.
hwabe: Cell.
uif–Lange–
lehalem.
chwabe: NEON.
on: newer Intel.
vet: GPUs.
neysu: FPGAs.
r Intel.
Hinterwälder–
hez–Schwabe:

ing–Schwabe–



## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".

## Simplicity

Curve25519 paper
advertised "short c

2013 Bernstein–Ja
Lange–Schwabe: "
reimplementing Na
tweets. Does spee

forms

2.

ell.

–

IEON.

Intel.

.

GAs.

der–

abe:

abe–



## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".

## Simplicity

Curve25519 paper advertised "short code."

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaC reimplementing NaCl in 100 tweets. Does speed matter?

## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".



## Simplicity

Curve25519 paper advertised "short code."

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaCl, reimplementing NaCl in 100 tweets. Does speed matter?

## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".



## Simplicity

Curve25519 paper advertised "short code."

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaCl, reimplementing NaCl in 100 tweets. Does speed matter?

Largest chunk of code: The hash function used inside signatures!

## Next-generation crypto library

NaCl: Networking and Cryptography library provides very simple new API for public-key authenticated encryption.

All-in-one `crypto_box` function uses Curve25519 for DH, Salsa20 for encryption, Poly1305 for authentication.

More on NaCl design: see 2011 Bernstein–Lange–Schwabe "The security impact of a new cryptographic library".



## Simplicity

Curve25519 paper advertised "short code."

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaCl, reimplementing NaCl in 100 tweets. Does speed matter?

Largest chunk of code: The hash function used inside signatures!

2014 Bernstein–van Gastel–Janssen–Lange–Schwabe–Smetsers: formal verification of some TweetNaCl properties.

neration crypto library

letworking and
raphy library provides
ple new API for public-
nenticated encryption.

ne `crypto_box` function
rve25519 for DH,
for encryption,
5 for authentication.

NaCl design: see
rnstein–Lange–Schwabe
curity impact of a
ptographic library".



Simplicity

Curve25519 paper
advertised "short code."

2013 Bernstein–Janssen–
Lange–Schwabe: TweetNaCl,
reimplementing NaCl in 100
tweets. Does speed matter?

Largest chunk of code: The hash
function used inside signatures!

2014 Bernstein–van Gastel–
Janssen–Lange–Schwabe–
Smetsers: formal verification of
some TweetNaCl properties.

2014 Ch
Tsai–Wa
Curve25
verificati
two high

Newer w
Russinof
surveyab
Curve25
Bernstei

Single-cu
*and* is th
towards

rypto library

and
ary provides
PI for public-
encryption.

_box function
or DH,
tion,
entication.

ign: see
ange–Schwabe
act of a
library".



Simplicity

Curve25519 paper
advertised "short code."

2013 Bernstein–Janssen–
Lange–Schwabe: TweetNaCl,
reimplementing NaCl in 100
tweets. Does speed matter?

Largest chunk of code: The hash
function used inside signatures!

2014 Bernstein–van Gastel–
Janssen–Lange–Schwabe–
Smetsers: formal verification of
some TweetNaCl properties.



2014 Chen–Hsu–L
Tsai–Wang–Yang–
Curve25519 softwa
verification of **cor**
two high-speed as

Newer work ongoi
Russinoff "A comp
surveyable proof o
Curve25519 group
Bernstein–Schwab

Single-curve code
*and* is the most p
towards bug-free E

ry

es
lic-
n.

tion

abe



## Simplicity

Curve25519 paper advertised “short code.”

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaCl, reimplementing NaCl in 100 tweets. Does speed matter?

Largest chunk of code: The hash function used inside signatures!

2014 Bernstein–van Gastel–Janssen–Lange–Schwabe–Smetsers: formal verification of some TweetNaCl properties.

2014 Chen–Hsu–Lin–Schwab
Tsai–Wang–Yang–Yang “Ve
Curve25519 software”: form
verification of **correctness** o
two high-speed asm main lo

Newer work ongoing: e.g., 2
Russinoff “A computationall
surveyable proof of the
Curve25519 group axioms”;
Bernstein–Schwabe `gfveri`

Single-curve code helps spee
*and* is the most promising a
towards bug-free ECC softwa

Simplicity

Curve25519 paper advertised "short code."

2013 Bernstein–Janssen–Lange–Schwabe: TweetNaCl, reimplementing NaCl in 100 tweets. Does speed matter?

Largest chunk of code: The hash function used inside signatures!

2014 Bernstein–van Gastel–Janssen–Lange–Schwabe–Smetsers: formal verification of some TweetNaCl properties.



2014 Chen–Hsu–Lin–Schwabe–Tsai–Wang–Yang–Yang "Verifying Curve25519 software": formal verification of **correctness** of two high-speed asm main loops.

Newer work ongoing: e.g., 2015 Russinoff "A computationally surveyable proof of the Curve25519 group axioms"; 2015 Bernstein–Schwabe `gfverif`.

Single-curve code helps speed *and* is the most promising avenue towards bug-free ECC software.

ty

519 paper
ed "short code."

rnstein–Janssen–
Schwabe: TweetNaCl,
nenting NaCl in 100
Does speed matter?

chunk of code: The hash
used inside signatures!

rnstein–van Gastel–
-Lange–Schwabe–
s: formal verification of
weetNaCl properties.

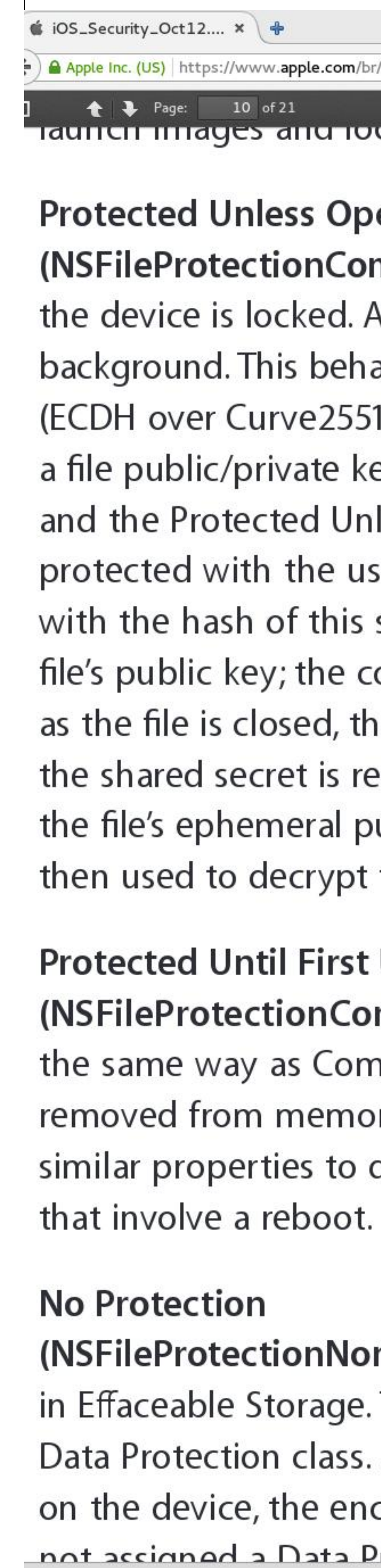


2014 Chen–Hsu–Lin–Schwabe–Tsai–Wang–Yang–Yang "Verifying Curve25519 software": formal verification of **correctness** of two high-speed asm main loops.

Newer work ongoing: e.g., 2015 Russinoff "A computationally surveyable proof of the Curve25519 group axioms"; 2015 Bernstein–Schwabe `gfverif`.

Single-curve code helps speed *and* is the most promising avenue towards bug-free ECC software.

2012: A


iOS_Security_Oct12....

Apple Inc. (US) https://www.apple.com/br/

Page: 10 of 21

launch images and loc

**Protected Unless Ope**
**(NSFileProtectionCom**
the device is locked. A
background. This beha
(ECDH over Curve2551
a file public/private ke
and the Protected Unl
protected with the us
with the hash of this s
file's public key; the co
as the file is closed, th
the shared secret is re
the file's ephemeral pu
then used to decrypt t

**Protected Until First**
**(NSFileProtectionCom**
the same way as Com
removed from memor
similar properties to d
that involve a reboot.

**No Protection**
**(NSFileProtectionNon**
in Effaceable Storage.
Data Protection class.
on the device, the enc
not assigned a Data P

code."

nssen–
TweetNaCl,
aCl in 100
d matter?

code: The hash
de signatures!

n Gastel–
chwabe–
verification of
properties.



2014 Chen–Hsu–Lin–Schwabe–Tsai–Wang–Yang–Yang "Verifying Curve25519 software": formal verification of **correctness** of two high-speed asm main loops.

Newer work ongoing: e.g., 2015 Russinoff "A computationally surveyable proof of the Curve25519 group axioms"; 2015 Bernstein–Schwabe `gfverif`.

Single-curve code helps speed *and* is the most promising avenue towards bug-free ECC software.

2012: Apple deplo


launch images and location data are also sto

**Protected Unless Open**
**(NSFileProtectionCompleteUnlessOpen):** So
the device is locked. A good example of this
background. This behavior is achieved by usin
(ECDH over Curve25519). Along with the usua
a file public/private key pair. A shared secret
and the Protected Unless Open class public k
protected with the user's passcode and the
with the hash of this shared secret and store
file's public key; the corresponding private ke
as the file is closed, the per-file key is also wi
the shared secret is re-created using the Prot
the file's ephemeral public key; its hash is use
then used to decrypt the file.

**Protected Until First User Authentication**
**(NSFileProtectionCompleteUntilFirstUserA**
the same way as Complete Protection, excep
removed from memory when the device is l
similar properties to desktop full-disk encryp
that involve a reboot.

**No Protection**
**(NSFileProtectionNone):** This class key is pro
in Effaceable Storage. This is the default class
Data Protection class. Since all the keys need
on the device, the encryption only affords th
not assigned a Data Protection class, it is still

l,

hash

res!

n of



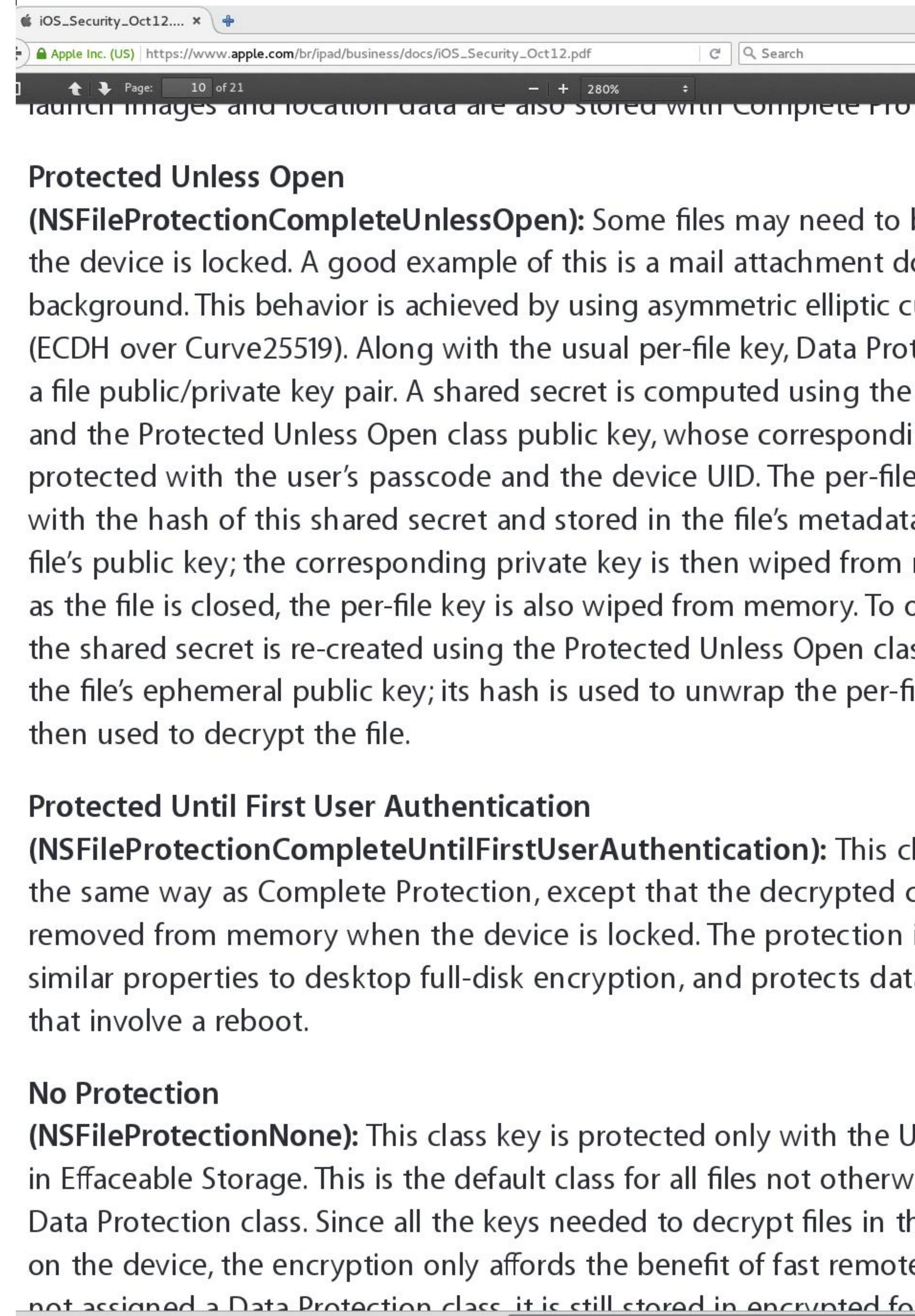
2014 Chen–Hsu–Lin–Schwabe–Tsai–Wang–Yang–Yang "Verifying Curve25519 software": formal verification of **correctness** of two high-speed asm main loops.

Newer work ongoing: e.g., 2015 Russinoff "A computationally surveyable proof of the Curve25519 group axioms"; 2015 Bernstein–Schwabe `gfverif`.

Single-curve code helps speed *and* is the most promising avenue towards bug-free ECC software.



2012: Apple deploys Curve2




**Protected Unless Open**

**(NSFileProtectionCompleteUnlessOpen):** Some files may need to b
the device is locked. A good example of this is a mail attachment do
background. This behavior is achieved by using asymmetric elliptic c
(ECDH over Curve25519). Along with the usual per-file key, Data Prot
a file public/private key pair. A shared secret is computed using the
and the Protected Unless Open class public key, whose correspondi
protected with the user's passcode and the device UID. The per-file
with the hash of this shared secret and stored in the file's metadata
file's public key; the corresponding private key is then wiped from
as the file is closed, the per-file key is also wiped from memory. To o
the shared secret is re-created using the Protected Unless Open clas
the file's ephemeral public key; its hash is used to unwrap the per-fi
then used to decrypt the file.

**Protected Until First User Authentication**

**(NSFileProtectionCompleteUntilFirstUserAuthentication):** This c
the same way as Complete Protection, except that the decrypted c
removed from memory when the device is locked. The protection i
similar properties to desktop full-disk encryption, and protects dat
that involve a reboot.

**No Protection**

**(NSFileProtectionNone):** This class key is protected only with the U
in Effaceable Storage. This is the default class for all files not otherw
Data Protection class. Since all the keys needed to decrypt files in th
on the device, the encryption only affords the benefit of fast remote

2014 Chen–Hsu–Lin–Schwabe–Tsai–Wang–Yang–Yang "Verifying Curve25519 software": formal verification of **correctness** of two high-speed asm main loops.

Newer work ongoing: e.g., 2015 Russinoff "A computationally surveyable proof of the Curve25519 group axioms"; 2015 Bernstein–Schwabe `gfverif`.

Single-curve code helps speed *and* is the most promising avenue towards bug-free ECC software.



## 2012: Apple deploys Curve25519

iOS_Security_Oct12....

Apple Inc. (US) https://www.apple.com/br/ipad/business/docs/iOS_Security_Oct12.pdf

Page: 10 of 21 280%

launch images and location data are also stored with Complete Protection.

**Protected Unless Open**

**(NSFileProtectionCompleteUnlessOpen):** Some files may need to be written while the device is locked. A good example of this is a mail attachment downloading in the background. This behavior is achieved by using asymmetric elliptic curve cryptography (ECDH over Curve25519). Along with the usual per-file key, Data Protection generates a file public/private key pair. A shared secret is computed using the file's private key and the Protected Unless Open class public key, whose corresponding private key is protected with the user's passcode and the device UID. The per-file key is wrapped with the hash of this shared secret and stored in the file's metadata along with the file's public key; the corresponding private key is then wiped from memory. As soon as the file is closed, the per-file key is also wiped from memory. To open the file again, the shared secret is re-created using the Protected Unless Open class's private key and the file's ephemeral public key; its hash is used to unwrap the per-file key, which is then used to decrypt the file.

**Protected Until First User Authentication**

**(NSFileProtectionCompleteUntilFirstUserAuthentication):** This class behaves in the same way as Complete Protection, except that the decrypted class key is not removed from memory when the device is locked. The protection in this class has similar properties to desktop full-disk encryption, and protects data from attacks that involve a reboot.

**No Protection**

**(NSFileProtectionNone):** This class key is protected only with the UID, and is kept in Effaceable Storage. This is the default class for all files not otherwise assigned to a Data Protection class. Since all the keys needed to decrypt files in this class are stored on the device, the encryption only affords the benefit of fast remote wipe. If a file is not assigned a Data Protection class, it is still stored in encrypted form (as is all data

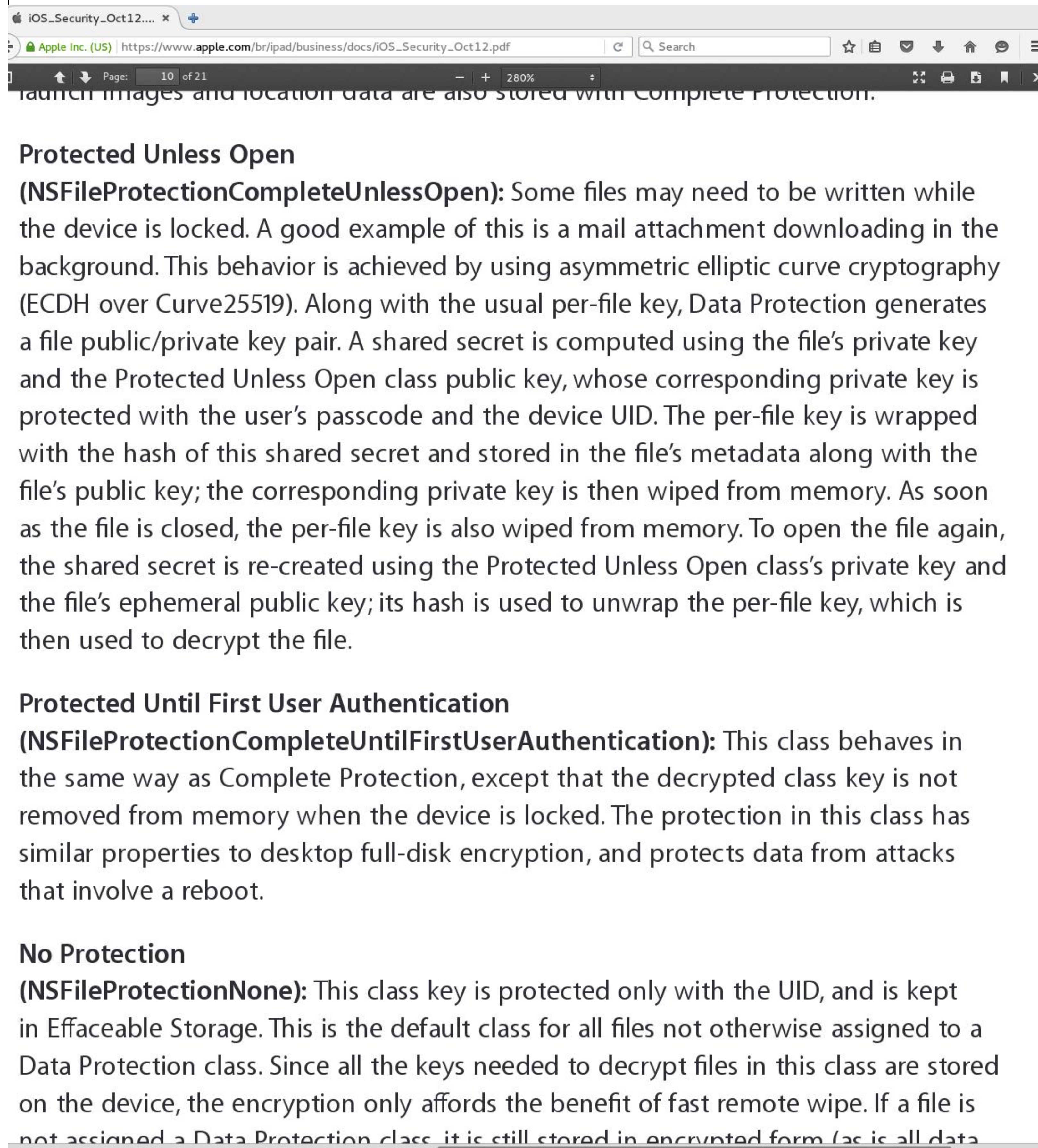
en–Hsu–Lin–Schwabe–
ang–Yang–Yang “Verifying
519 software”: formal
ion of **correctness** of
-speed asm main loops.

ork ongoing: e.g., 2015
ff “A computationally
le proof of the
519 group axioms”; 2015
n–Schwabe `gfverif`.

urve code helps speed
e most promising avenue
bug-free ECC software.



## 2012: Apple deploys Curve25519



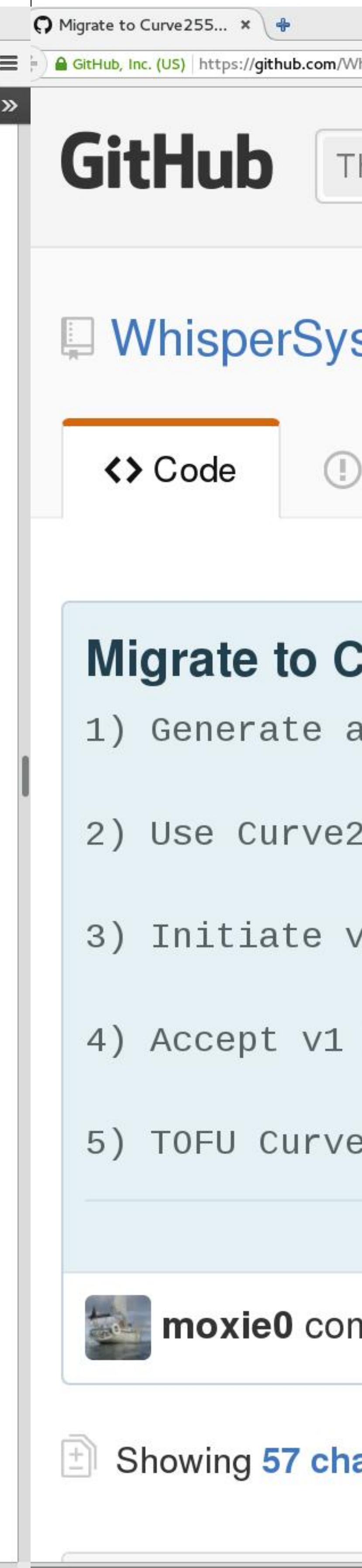

launch images and location data are also stored with Complete Protection.

**Protected Unless Open**
**(NSFileProtectionCompleteUnlessOpen):** Some files may need to be written while the device is locked. A good example of this is a mail attachment downloading in the background. This behavior is achieved by using asymmetric elliptic curve cryptography (ECDH over Curve25519). Along with the usual per-file key, Data Protection generates a file public/private key pair. A shared secret is computed using the file's private key and the Protected Unless Open class public key, whose corresponding private key is protected with the user's passcode and the device UID. The per-file key is wrapped with the hash of this shared secret and stored in the file's metadata along with the file's public key; the corresponding private key is then wiped from memory. As soon as the file is closed, the per-file key is also wiped from memory. To open the file again, the shared secret is re-created using the Protected Unless Open class's private key and the file's ephemeral public key; its hash is used to unwrap the per-file key, which is then used to decrypt the file.

**Protected Until First User Authentication**
**(NSFileProtectionCompleteUntilFirstUserAuthentication):** This class behaves in the same way as Complete Protection, except that the decrypted class key is not removed from memory when the device is locked. The protection in this class has similar properties to desktop full-disk encryption, and protects data from attacks that involve a reboot.

**No Protection**
**(NSFileProtectionNone):** This class key is protected only with the UID, and is kept in Effaceable Storage. This is the default class for all files not otherwise assigned to a Data Protection class. Since all the keys needed to decrypt files in this class are stored on the device, the encryption only affords the benefit of fast remote wipe. If a file is not assigned a Data Protection class, it is still stored in encrypted form (as is all data

## 2013: Si



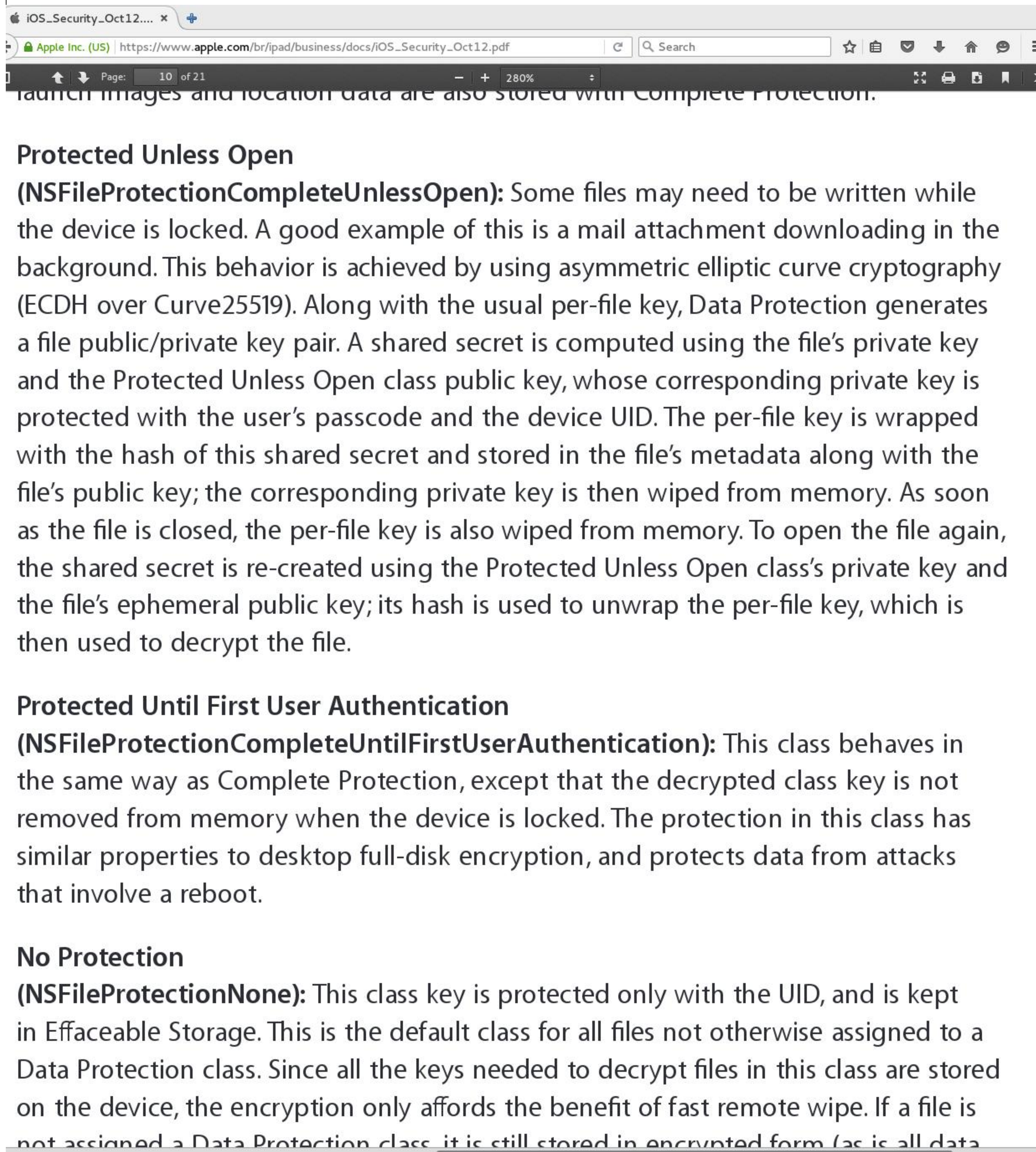
in–Schwabe–
-Yang "Verifying
are": formal
rectness of
m main loops.

ng: e.g., 2015
putationally
f the
axioms"; 2015
e gfverif.

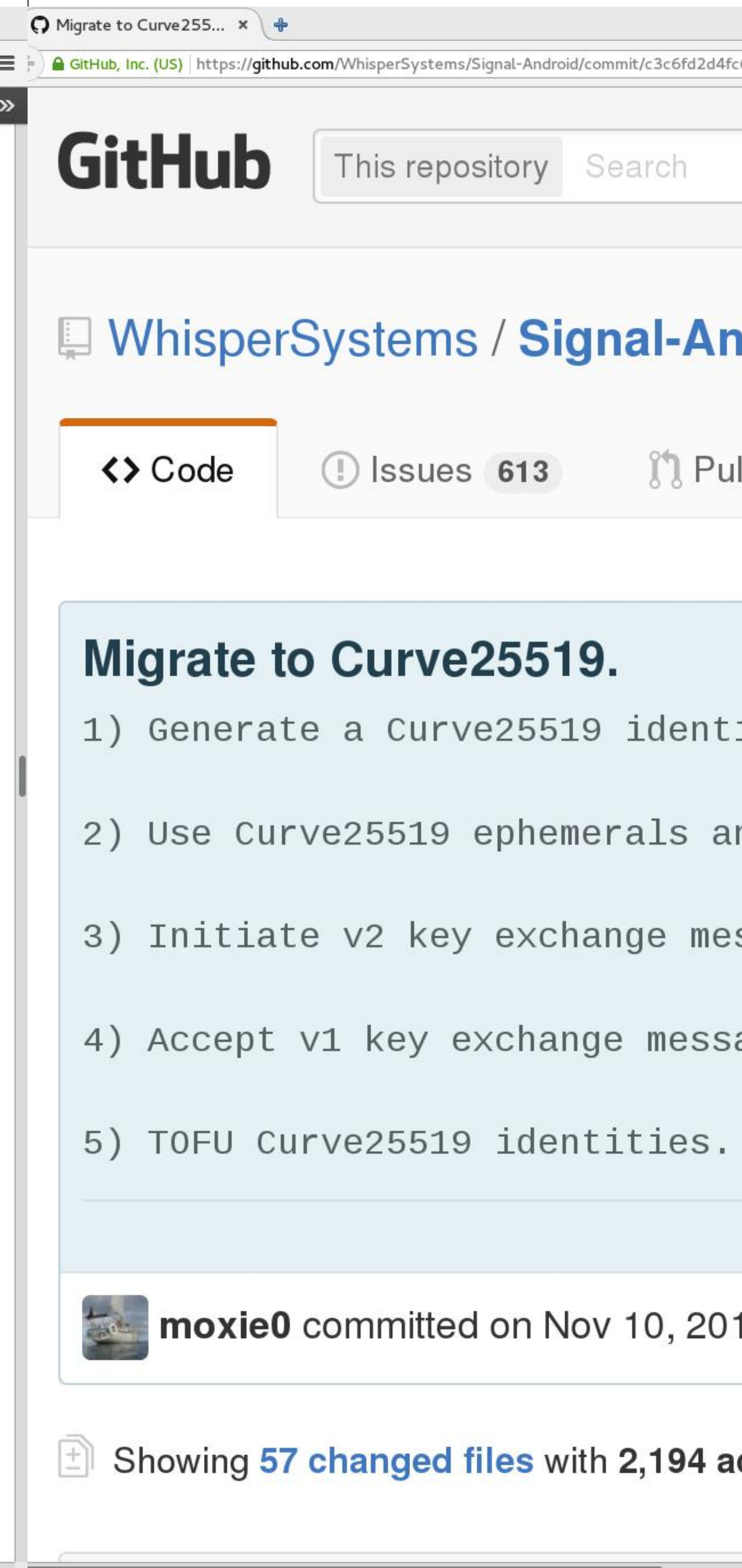
helps speed
romising avenue
ECC software.



# 2012: Apple deploys Curve25519



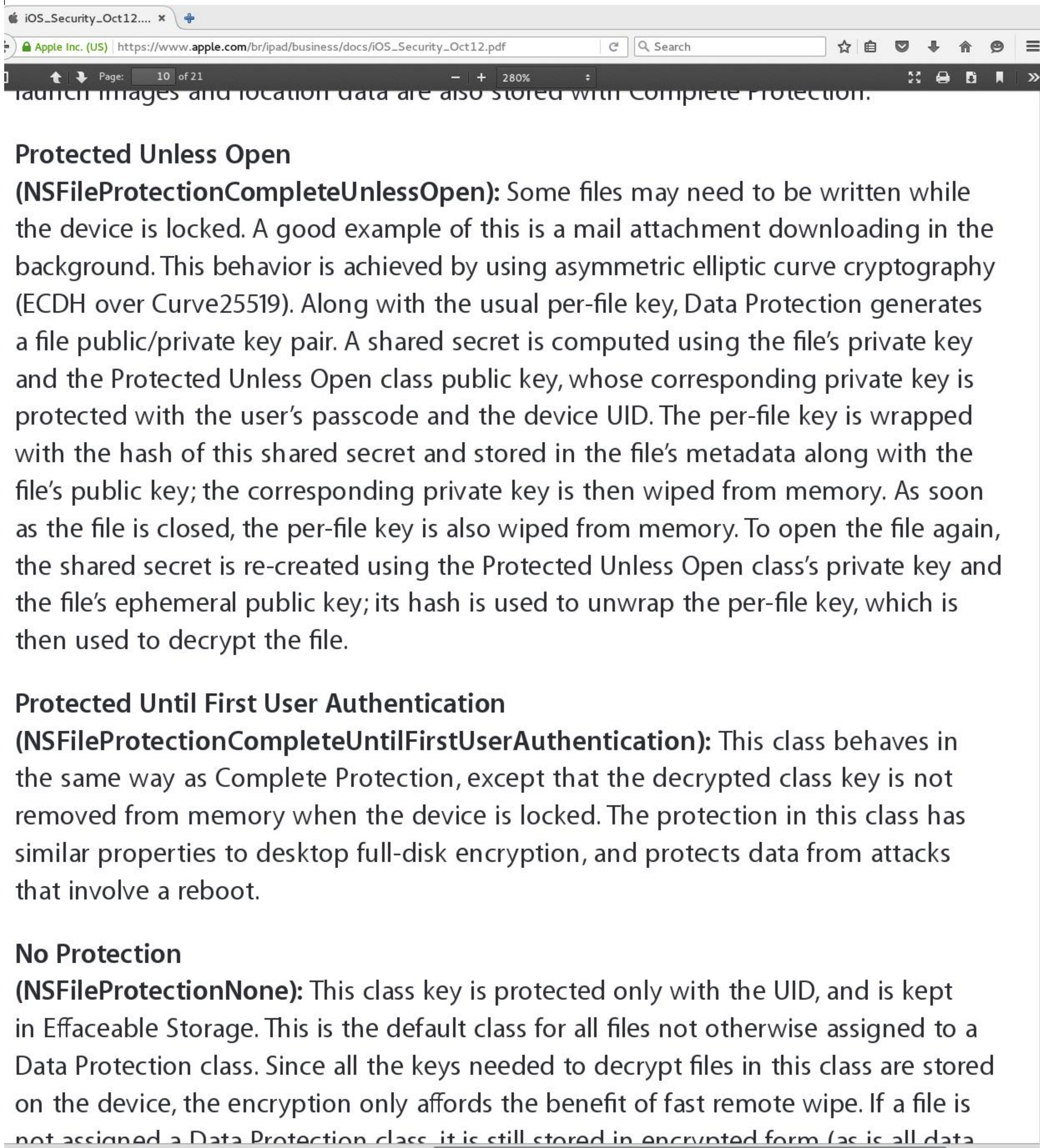

# 2013: Signal deplo



pe–
rifying
al
of
ops.
2015
y
2015
f.
ed
venue
are.



# 2012: Apple deploys Curve25519




launch images and location data are also stored with Complete Protection.

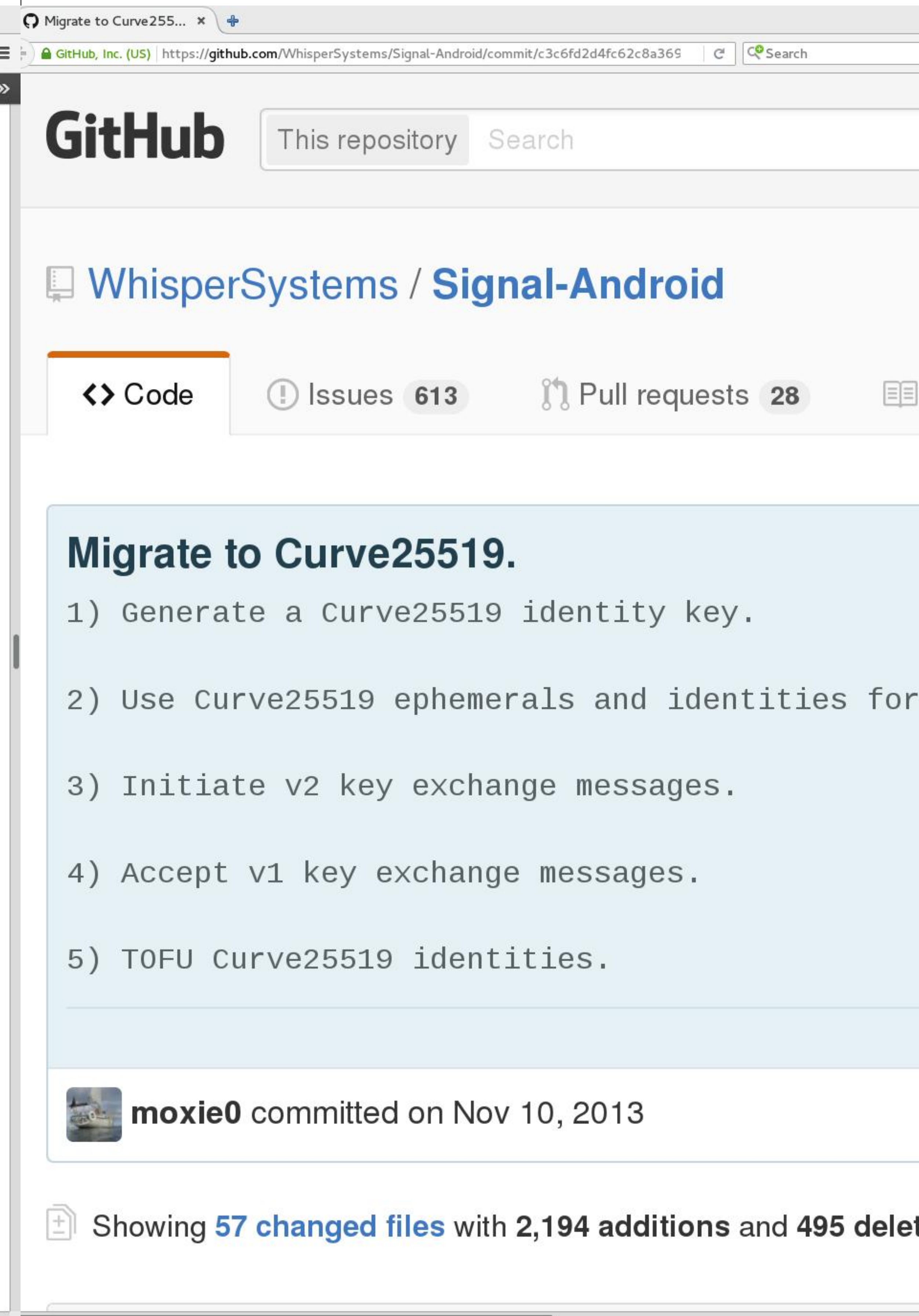
**Protected Unless Open (NSFileProtectionCompleteUnlessOpen):** Some files may need to be written while the device is locked. A good example of this is a mail attachment downloading in the background. This behavior is achieved by using asymmetric elliptic curve cryptography (ECDH over Curve25519). Along with the usual per-file key, Data Protection generates a file public/private key pair. A shared secret is computed using the file's private key and the Protected Unless Open class public key, whose corresponding private key is protected with the user's passcode and the device UID. The per-file key is wrapped with the hash of this shared secret and stored in the file's metadata along with the file's public key; the corresponding private key is then wiped from memory. As soon as the file is closed, the per-file key is also wiped from memory. To open the file again, the shared secret is re-created using the Protected Unless Open class's private key and the file's ephemeral public key; its hash is used to unwrap the per-file key, which is then used to decrypt the file.

**Protected Until First User Authentication (NSFileProtectionCompleteUntilFirstUserAuthentication):** This class behaves in the same way as Complete Protection, except that the decrypted class key is not removed from memory when the device is locked. The protection in this class has similar properties to desktop full-disk encryption, and protects data from attacks that involve a reboot.

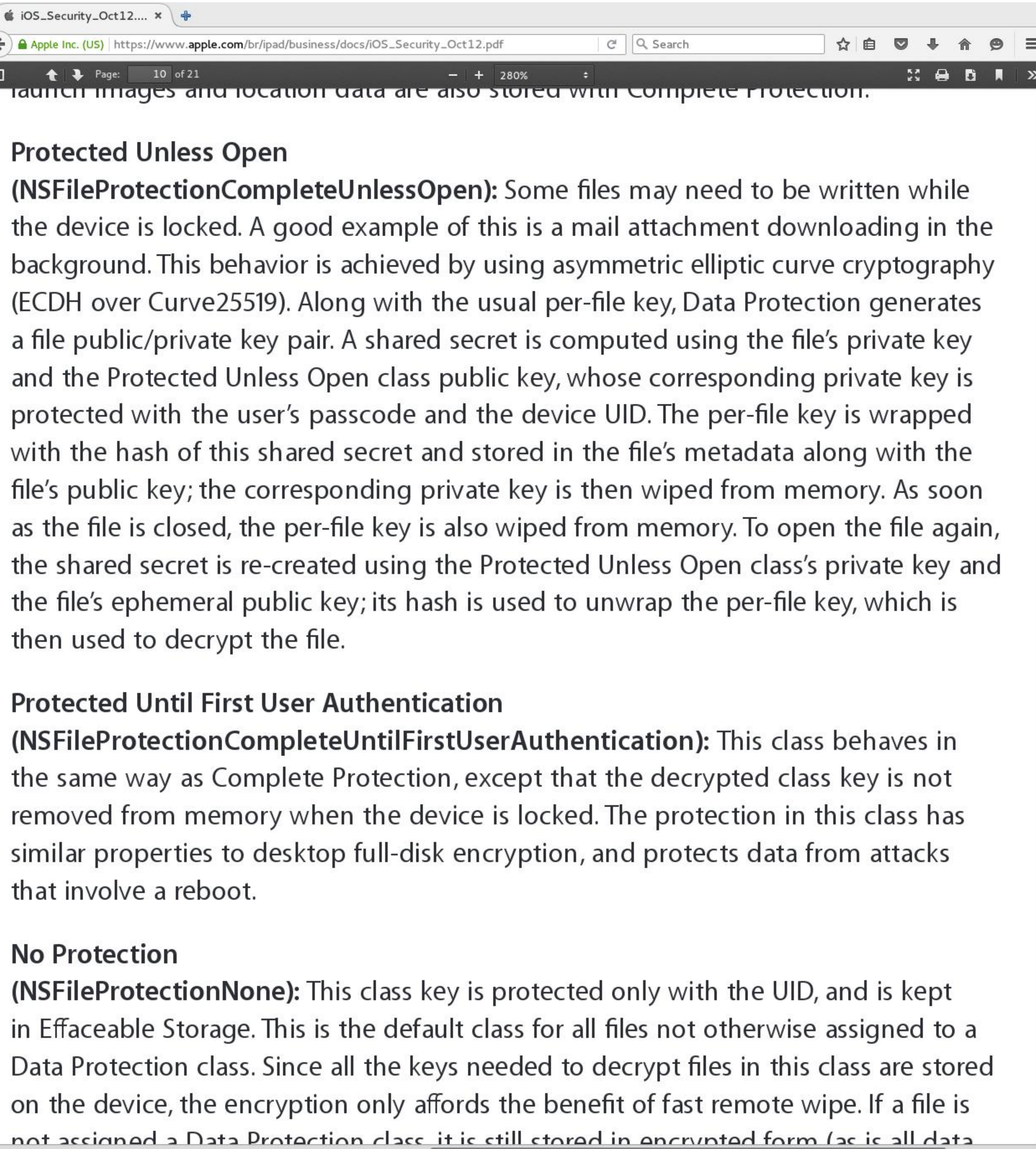
**No Protection (NSFileProtectionNone):** This class key is protected only with the UID, and is kept in Effaceable Storage. This is the default class for all files not otherwise assigned to a Data Protection class. Since all the keys needed to decrypt files in this class are stored on the device, the encryption only affords the benefit of fast remote wipe. If a file is not assigned a Data Protection class, it is still stored in encrypted form (as is all data



# 2013: Signal deploys Curve2



# 2012: Apple deploys Curve25519


launch images and location data are also stored with Complete Protection.

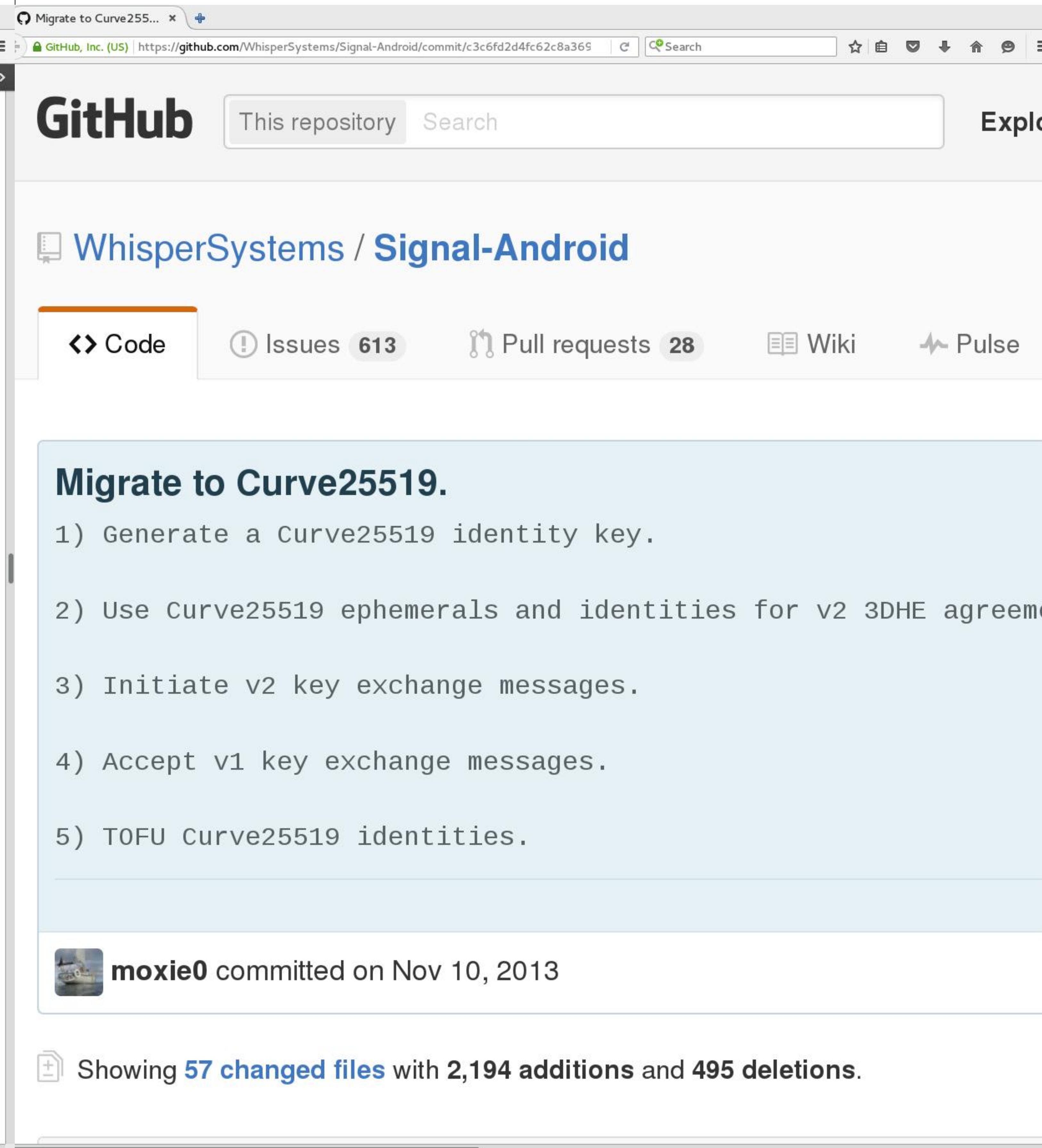
**Protected Unless Open**
**(NSFileProtectionCompleteUnlessOpen):** Some files may need to be written while the device is locked. A good example of this is a mail attachment downloading in the background. This behavior is achieved by using asymmetric elliptic curve cryptography (ECDH over Curve25519). Along with the usual per-file key, Data Protection generates a file public/private key pair. A shared secret is computed using the file's private key and the Protected Unless Open class public key, whose corresponding private key is protected with the user's passcode and the device UID. The per-file key is wrapped with the hash of this shared secret and stored in the file's metadata along with the file's public key; the corresponding private key is then wiped from memory. As soon as the file is closed, the per-file key is also wiped from memory. To open the file again, the shared secret is re-created using the Protected Unless Open class's private key and the file's ephemeral public key; its hash is used to unwrap the per-file key, which is then used to decrypt the file.

**Protected Until First User Authentication**
**(NSFileProtectionCompleteUntilFirstUserAuthentication):** This class behaves in the same way as Complete Protection, except that the decrypted class key is not removed from memory when the device is locked. The protection in this class has similar properties to desktop full-disk encryption, and protects data from attacks that involve a reboot.

**No Protection**
**(NSFileProtectionNone):** This class key is protected only with the UID, and is kept in Effaceable Storage. This is the default class for all files not otherwise assigned to a Data Protection class. Since all the keys needed to decrypt files in this class are stored on the device, the encryption only affords the benefit of fast remote wipe. If a file is not assigned a Data Protection class, it is still stored in encrypted form (as is all data



# 2013: Signal deploys Curve25519



# pple deploys Curve25519

cation data are also stored with Complete Protection.

en

**npleteUnlessOpen):** Some files may need to be written while
good example of this is a mail attachment downloading in the
vior is achieved by using asymmetric elliptic curve cryptography
9). Along with the usual per-file key, Data Protection generates
y pair. A shared secret is computed using the file's private key
ess Open class public key, whose corresponding private key is
er's passcode and the device UID. The per-file key is wrapped
hared secret and stored in the file's metadata along with the
orresponding private key is then wiped from memory. As soon
e per-file key is also wiped from memory. To open the file again,
-created using the Protected Unless Open class's private key and
ublic key; its hash is used to unwrap the per-file key, which is
the file.

**User Authentication**

**npleteUntilFirstUserAuthentication):** This class behaves in
plete Protection, except that the decrypted class key is not
ry when the device is locked. The protection in this class has
esktop full-disk encryption, and protects data from attacks

**ne):** This class key is protected only with the UID, and is kept
This is the default class for all files not otherwise assigned to a
Since all the keys needed to decrypt files in this class are stored
ryption only affords the benefit of fast remote wipe. If a file is
rotection class, it is still stored in encrypted form (as is all data



# 2013: Signal deploys Curve25519

GitHub This repository Search Explo

WhisperSystems / **Signal-Android**

Code Issues 613 Pull requests 28 Wiki Pulse

## Migrate to Curve25519.

```
1) Generate a Curve25519 identity key.

2) Use Curve25519 ephemerals and identities for v2 3DHE agreeme

3) Initiate v2 key exchange messages.

4) Accept v1 key exchange messages.

5) TOFU Curve25519 identities.
```

**moxie0** committed on Nov 10, 2013

Showing **57 changed files** with **2,194 additions** and **495 deletions**.

# 2014: O

```
Changes since OpenS
===================

This is a feature-f

New features:

 * ssh(1), sshd(8):
   Diffie Hellman i
   method is the de

 * ssh(1), sshd(8):
   Ed25519 is a ell
   better security
   used for both us

 * Add a new privat
   protect keys at
   Ed25519 keys, bu
   existing keys of
   We intend to mak
   Details of the n

 * ssh(1), sshd(8):
   "chacha20-poly13
   ChaCha20 stream
   encryption mode.

 * ssh(1), sshd(8):
   servers that use
   still be possibl
   DSA keys will be
   entirely in a fu

 * ssh(1), sshd(8):
   use a weaker key
```

# ys Curve25519

red with Complete Protection.

ome files may need to be written while
is a mail attachment downloading in the
ng asymmetric elliptic curve cryptography
al per-file key, Data Protection generates
is computed using the file's private key
key, whose corresponding private key is
device UID. The per-file key is wrapped
ed in the file's metadata along with the
ey is then wiped from memory. As soon
ped from memory. To open the file again,
ected Unless Open class's private key and
ed to unwrap the per-file key, which is

uthentication): This class behaves in
ot that the decrypted class key is not
ocked. The protection in this class has
otion, and protects data from attacks

otected only with the UID, and is kept
for all files not otherwise assigned to a
ed to decrypt files in this class are stored
e benefit of fast remote wipe. If a file is
stored in encrypted form (as is all data



# 2013: Signal deploys Curve25519

Migrate to Curve255...

GitHub, Inc. (US) https://github.com/WhisperSystems/Signal-Android/commit/c3c6fd2d4fc62c8a365

GitHub This repository Search Explo

WhisperSystems / **Signal-Android**

Code | Issues 613 | Pull requests 28 | Wiki | Pulse

**Migrate to Curve25519.**

```
1) Generate a Curve25519 identity key.

2) Use Curve25519 ephemerals and identities for v2 3DHE agreeme

3) Initiate v2 key exchange messages.

4) Accept v1 key exchange messages.

5) TOFU Curve25519 identities.
```

**moxie0** committed on Nov 10, 2013

Showing **57 changed files** with **2,194 additions** and **495 deletions**.

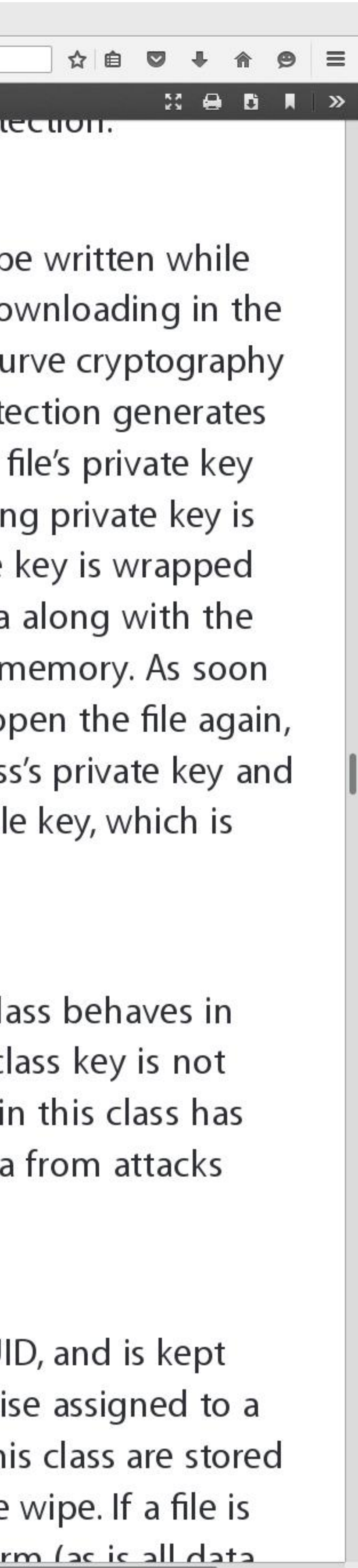
# 2014: OpenSSH d

http://www...lease-6.5

www.openssh.com/txt/release-6.5

```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for k
   Diffie Hellman in Daniel Bernstein
   method is the default when both th

 * ssh(1), sshd(8): Add support for E
   Ed25519 is a elliptic curve signat
   better security than ECDSA and DSA
   used for both user and host keys.

 * Add a new private key format that
   protect keys at rest. This format
   Ed25519 keys, but may be requested
   existing keys of other types via t
   We intend to make the new format t
   Details of the new format are in t

 * ssh(1), sshd(8): Add a new transpo
   "chacha20-poly1305@openssh.com" th
   ChaCha20 stream cipher and Poly130
   encryption mode. Details are in th

 * ssh(1), sshd(8): Refuse RSA keys f
   servers that use the obsolete RSA+
   still be possible to connect with
   DSA keys will be accepted, and Ope
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old propri
   use a weaker key exchange hash cal
```

5519


lection.

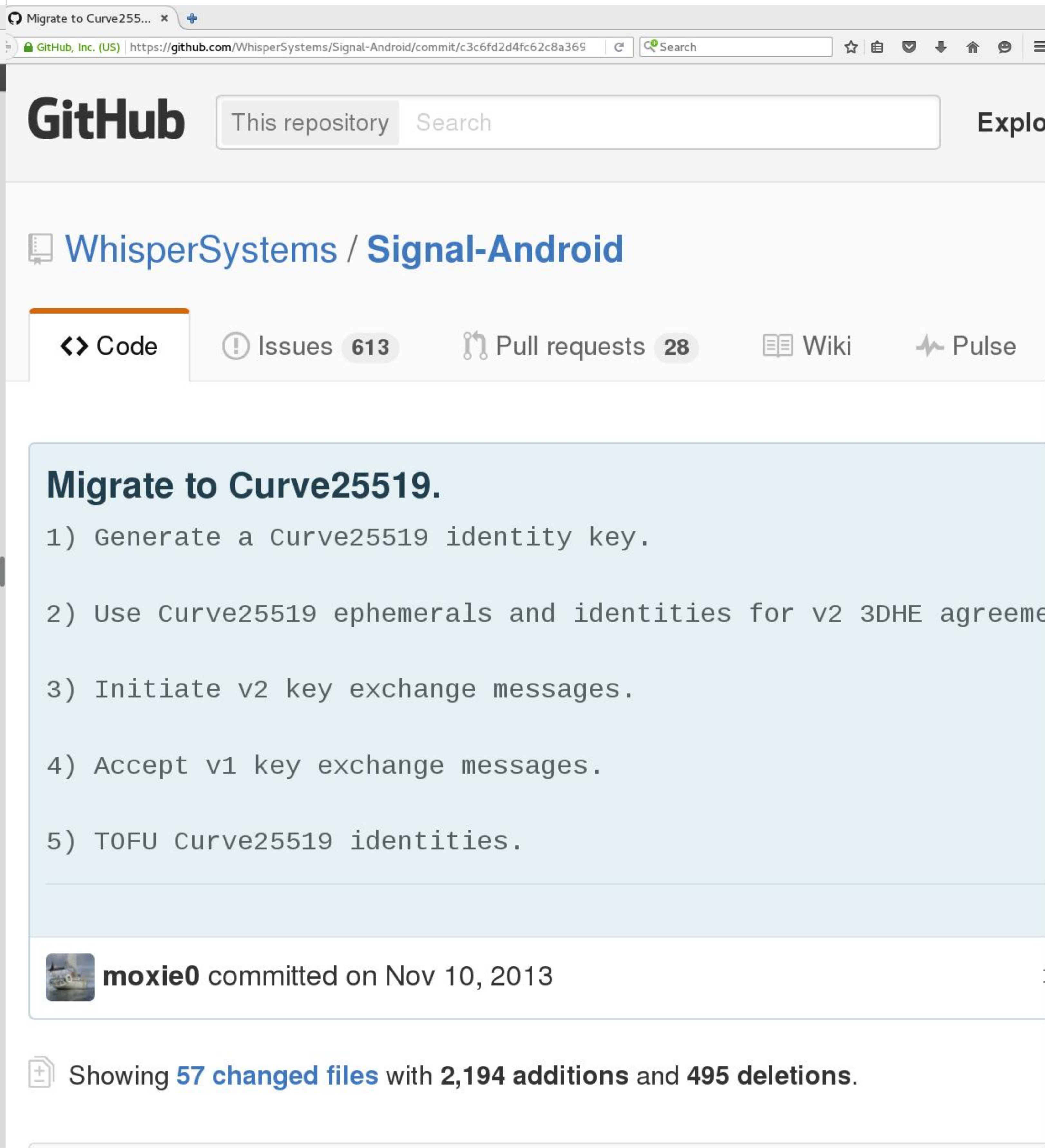
be written while
ownloading in the
urve cryptography
tection generates
file's private key
ng private key is
e key is wrapped
a along with the
memory. As soon
open the file again,
ss's private key and
le key, which is

lass behaves in
class key is not
in this class has
a from attacks

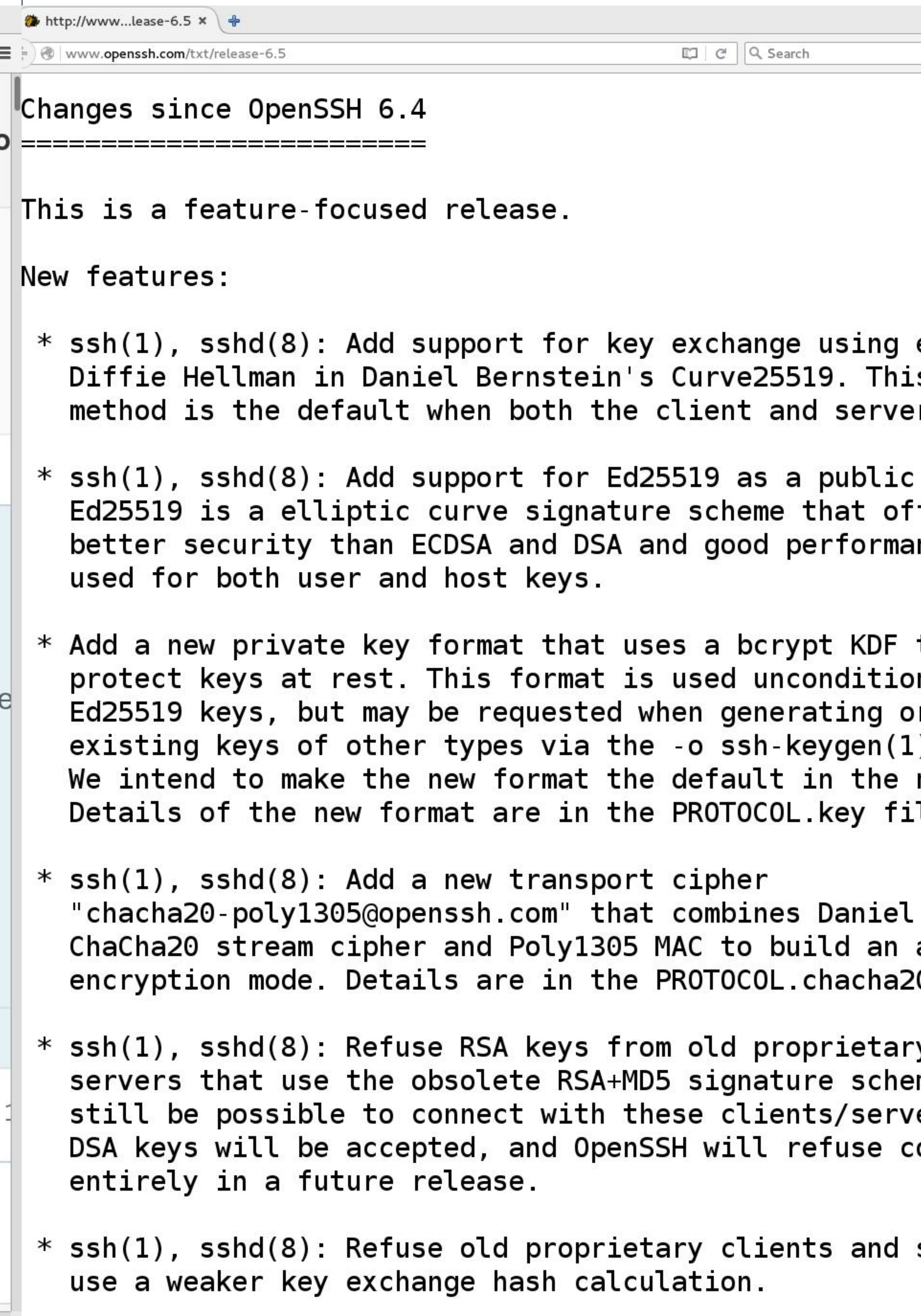
ID, and is kept
ise assigned to a
is class are stored
e wipe. If a file is
rm (as is all data



# 2013: Signal deploys Curve25519



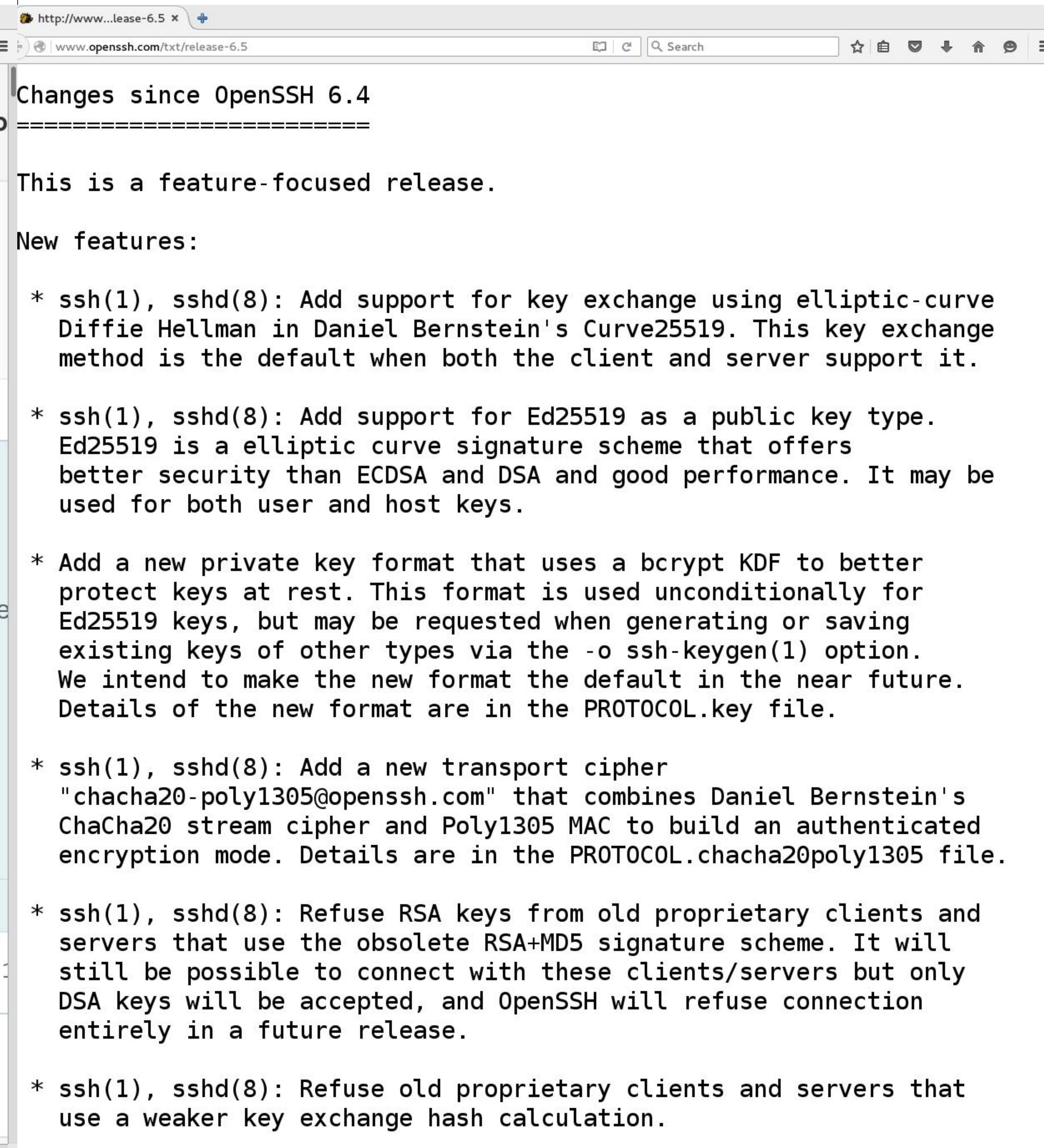

# 2014: OpenSSH deploys Cu



# 2013: Signal deploys Curve25519




# 2014: OpenSSH deploys Curve25519

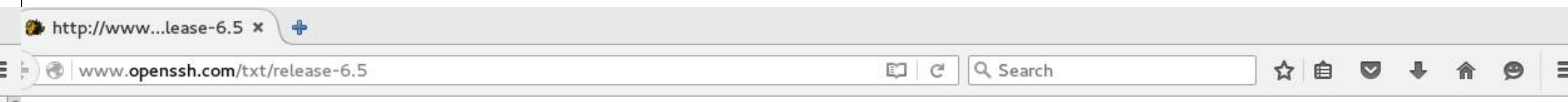


```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for key exchange using elliptic-curve
   Diffie Hellman in Daniel Bernstein's Curve25519. This key exchange
   method is the default when both the client and server support it.

 * ssh(1), sshd(8): Add support for Ed25519 as a public key type.
   Ed25519 is a elliptic curve signature scheme that offers
   better security than ECDSA and DSA and good performance. It may be
   used for both user and host keys.

 * Add a new private key format that uses a bcrypt KDF to better
   protect keys at rest. This format is used unconditionally for
   Ed25519 keys, but may be requested when generating or saving
   existing keys of other types via the -o ssh-keygen(1) option.
   We intend to make the new format the default in the near future.
   Details of the new format are in the PROTOCOL.key file.

 * ssh(1), sshd(8): Add a new transport cipher
   "chacha20-poly1305@openssh.com" that combines Daniel Bernstein's
   ChaCha20 stream cipher and Poly1305 MAC to build an authenticated
   encryption mode. Details are in the PROTOCOL.chacha20poly1305 file.

 * ssh(1), sshd(8): Refuse RSA keys from old proprietary clients and
   servers that use the obsolete RSA+MD5 signature scheme. It will
   still be possible to connect with these clients/servers but only
   DSA keys will be accepted, and OpenSSH will refuse connection
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old proprietary clients and servers that
   use a weaker key exchange hash calculation.
```

## ignal deploys Curve25519




## 2014: OpenSSH deploys Curve25519




```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for key exchange using elliptic-curve
   Diffie Hellman in Daniel Bernstein's Curve25519. This key exchange
   method is the default when both the client and server support it.

 * ssh(1), sshd(8): Add support for Ed25519 as a public key type.
   Ed25519 is a elliptic curve signature scheme that offers
   better security than ECDSA and DSA and good performance. It may be
   used for both user and host keys.

 * Add a new private key format that uses a bcrypt KDF to better
   protect keys at rest. This format is used unconditionally for
   Ed25519 keys, but may be requested when generating or saving
   existing keys of other types via the -o ssh-keygen(1) option.
   We intend to make the new format the default in the near future.
   Details of the new format are in the PROTOCOL.key file.

 * ssh(1), sshd(8): Add a new transport cipher
   "chacha20-poly1305@openssh.com" that combines Daniel Bernstein's
   ChaCha20 stream cipher and Poly1305 MAC to build an authenticated
   encryption mode. Details are in the PROTOCOL.chacha20poly1305 file.

 * ssh(1), sshd(8): Refuse RSA keys from old proprietary clients and
   servers that use the obsolete RSA+MD5 signature scheme. It will
   still be possible to connect with these clients/servers but only
   DSA keys will be accepted, and OpenSSH will refuse connection
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old proprietary clients and servers that
   use a weaker key exchange hash calculation.
```

2015.10

EdDSA–

for signa

X25519

2015.10

ECC sta

paving w

2015.11

X25519

These ar

Many m

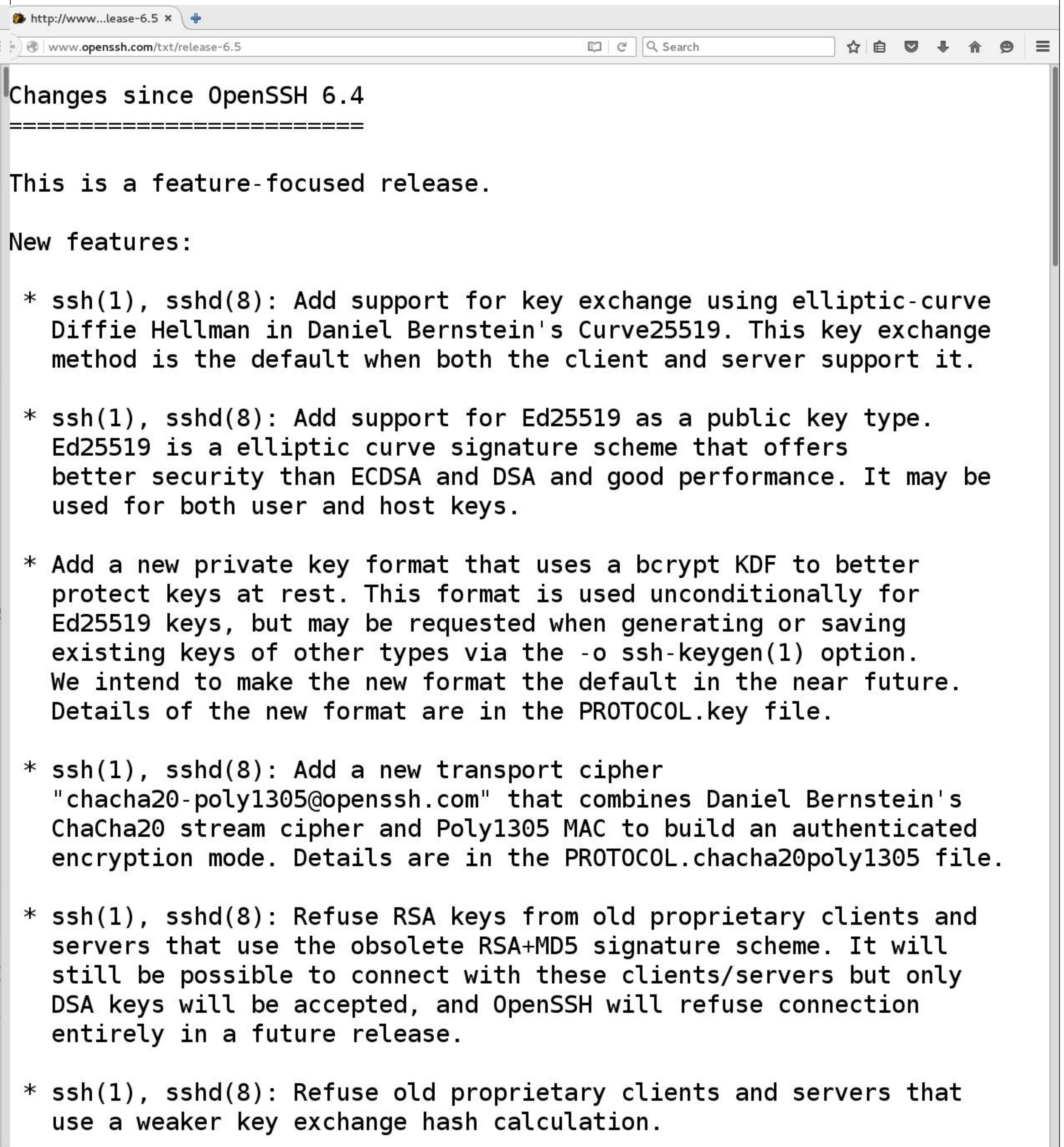
/curve2

and /ed

## oys Curve25519




## 2014: OpenSSH deploys Curve25519




```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for key exchange using elliptic-curve
   Diffie Hellman in Daniel Bernstein's Curve25519. This key exchange
   method is the default when both the client and server support it.

 * ssh(1), sshd(8): Add support for Ed25519 as a public key type.
   Ed25519 is a elliptic curve signature scheme that offers
   better security than ECDSA and DSA and good performance. It may be
   used for both user and host keys.

 * Add a new private key format that uses a bcrypt KDF to better
   protect keys at rest. This format is used unconditionally for
   Ed25519 keys, but may be requested when generating or saving
   existing keys of other types via the -o ssh-keygen(1) option.
   We intend to make the new format the default in the near future.
   Details of the new format are in the PROTOCOL.key file.

 * ssh(1), sshd(8): Add a new transport cipher
   "chacha20-poly1305@openssh.com" that combines Daniel Bernstein's
   ChaCha20 stream cipher and Poly1305 MAC to build an authenticated
   encryption mode. Details are in the PROTOCOL.chacha20poly1305 file.

 * ssh(1), sshd(8): Refuse RSA keys from old proprietary clients and
   servers that use the obsolete RSA+MD5 signature scheme. It will
   still be possible to connect with these clients/servers but only
   DSA keys will be accepted, and OpenSSH will refuse connection
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old proprietary clients and servers that
   use a weaker key exchange hash calculation.
```

2015.10: IRTF CF
EdDSA—Ed25519
for signatures. Alr
X25519 and X448

2015.10: NIST re
ECC standards for
paving way for nev

2015.11: BoringSS
X25519 and Ed25

These are just son
Many more: iani
/curve25519-dep
and /ed25519-de

## 25519

Explo

Wiki Pulse

v2 3DHE agreeme

ions.



## 2014: OpenSSH deploys Curve25519

http://www...lease-6.5 ×

www.openssh.com/txt/release-6.5

Search

```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for key exchange using elliptic-curve
   Diffie Hellman in Daniel Bernstein's Curve25519. This key exchange
   method is the default when both the client and server support it.

 * ssh(1), sshd(8): Add support for Ed25519 as a public key type.
   Ed25519 is a elliptic curve signature scheme that offers
   better security than ECDSA and DSA and good performance. It may be
   used for both user and host keys.

 * Add a new private key format that uses a bcrypt KDF to better
   protect keys at rest. This format is used unconditionally for
   Ed25519 keys, but may be requested when generating or saving
   existing keys of other types via the -o ssh-keygen(1) option.
   We intend to make the new format the default in the near future.
   Details of the new format are in the PROTOCOL.key file.

 * ssh(1), sshd(8): Add a new transport cipher
   "chacha20-poly1305@openssh.com" that combines Daniel Bernstein's
   ChaCha20 stream cipher and Poly1305 MAC to build an authenticated
   encryption mode. Details are in the PROTOCOL.chacha20poly1305 file.

 * ssh(1), sshd(8): Refuse RSA keys from old proprietary clients and
   servers that use the obsolete RSA+MD5 signature scheme. It will
   still be possible to connect with these clients/servers but only
   DSA keys will be accepted, and OpenSSH will refuse connection
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old proprietary clients and servers that
   use a weaker key exchange hash calculation.
```

2015.10: IRTF CFRG settle
EdDSA—Ed25519 and Ed44
for signatures. Already selec
X25519 and X448 for DH.

2015.10: NIST reopens its
ECC standards for comment
paving way for new curves.

2015.11: BoringSSL adds
X25519 and Ed25519.

These are just some highligh
Many more: ianix.com/pu
/curve25519-deployment
and /ed25519-deployment

# 2014: OpenSSH deploys Curve25519

http://www...lease-6.5

www.openssh.com/txt/release-6.5

```
Changes since OpenSSH 6.4
=========================

This is a feature-focused release.

New features:

 * ssh(1), sshd(8): Add support for key exchange using elliptic-curve
   Diffie Hellman in Daniel Bernstein's Curve25519. This key exchange
   method is the default when both the client and server support it.

 * ssh(1), sshd(8): Add support for Ed25519 as a public key type.
   Ed25519 is a elliptic curve signature scheme that offers
   better security than ECDSA and DSA and good performance. It may be
   used for both user and host keys.

 * Add a new private key format that uses a bcrypt KDF to better
   protect keys at rest. This format is used unconditionally for
   Ed25519 keys, but may be requested when generating or saving
   existing keys of other types via the -o ssh-keygen(1) option.
   We intend to make the new format the default in the near future.
   Details of the new format are in the PROTOCOL.key file.

 * ssh(1), sshd(8): Add a new transport cipher
   "chacha20-poly1305@openssh.com" that combines Daniel Bernstein's
   ChaCha20 stream cipher and Poly1305 MAC to build an authenticated
   encryption mode. Details are in the PROTOCOL.chacha20poly1305 file.

 * ssh(1), sshd(8): Refuse RSA keys from old proprietary clients and
   servers that use the obsolete RSA+MD5 signature scheme. It will
   still be possible to connect with these clients/servers but only
   DSA keys will be accepted, and OpenSSH will refuse connection
   entirely in a future release.

 * ssh(1), sshd(8): Refuse old proprietary clients and servers that
   use a weaker key exchange hash calculation.
```



2015.10: IRTF CFRG settles on EdDSA—Ed25519 and Ed448—for signatures. Already selected X25519 and X448 for DH.

2015.10: NIST reopens its ECC standards for comment, paving way for new curves.

2015.11: BoringSSL adds X25519 and Ed25519.

These are just some highlights. Many more: `ianix.com/pub/curve25519-deployment.html` and `/ed25519-deployment.html`.