

# 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).”

10 years of Curve25519

Bernstein

University of Illinois at Chicago &  
Radboud University Eindhoven

---

2019: [Seminar talk](#);

software close to done.

2015: [Software online](#).

2020: [Invited talk at ECC](#).

2015: [Paper online](#);

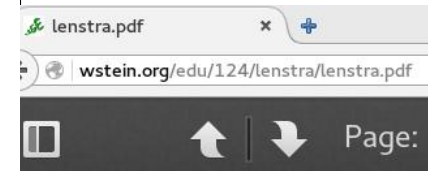
presented to PKC 2006.

1

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).”

2

Elliptic-c



Factor

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

Acknowledg

1

of Curve25519

n

is at Chicago &  
siteit Eindhoven

nar talk;

lose to done.

ware online.

d talk at ECC.

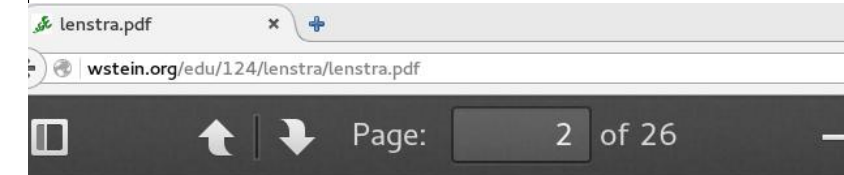
r online;

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).”

2

Elliptic-curve com



Annals of Mathematics,

## Factoring integers v

By H. W. LE

Abstr

This paper is devoted to the description of an algorithm for factoring positive integers. It depends on the fact that the least prime factor  $p$  of a positive integer  $n$  is obtained from Pollard's  $(p-1)$ -method (1974, 521–528) by replacing the multiplication by a random elliptic curve. It is conjectured that the expected time to find a non-trivial divisor of a composite number  $n$  is  $O(K(p)(\log n)^2)$ , where  $p$  is the least prime factor of  $n$ , which  $\log K(x) = \sqrt{(2 + o(1)) \log x \log \log x}$  when  $n$  is the product of two primes of size  $\sqrt{n}$ . This is faster than the  $\exp((1 + o(1))\sqrt{\log n \log \log n})$  (for  $n \rightarrow \infty$ ) algorithms of which the conjectural expected time is given by the above formula. However, these algorithms have a time complexity independent of the size of the prime factors. The new method is substantially faster for small  $p$ .

Acknowledgements. This paper was

5519

ago &  
hoven

ne.

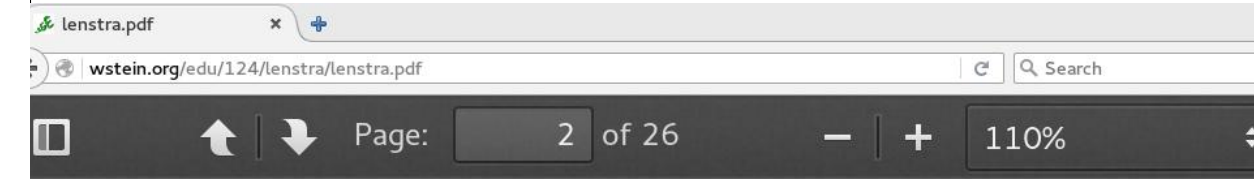
ECC.

1

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).”

2

# 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 for factoring positive integers. It depends on the use of elliptic curves. The algorithm is obtained from Pollard’s  $(p - 1)$ -method (Proc. Cambridge Math. Soc. (2) 77 (1974), 521–528) by replacing the multiplicative group by the group of points on a random elliptic curve. It is conjectured that the algorithm finds a non-trivial divisor of a composite number  $n$  in expected time  $O(K(p)(\log n)^2)$ , where  $p$  is the least prime dividing  $n$  and  $K$  is a constant which  $\log K(x) = \sqrt{(2 + o(1)) \log x \log \log x}$  for  $x \rightarrow \infty$ . In particular, when  $n$  is the product of two primes of the same order of magnitude, the running time is  $O(\exp((1 + o(1))\sqrt{\log n \log \log n}))$  (for  $n \rightarrow \infty$ ). There are several other algorithms of which the conjectural expected running time is given by a similar formula. However, these algorithms have a running time that is independent of the size of the prime factors of  $n$ , whereas the new method is substantially faster for small  $p$ .

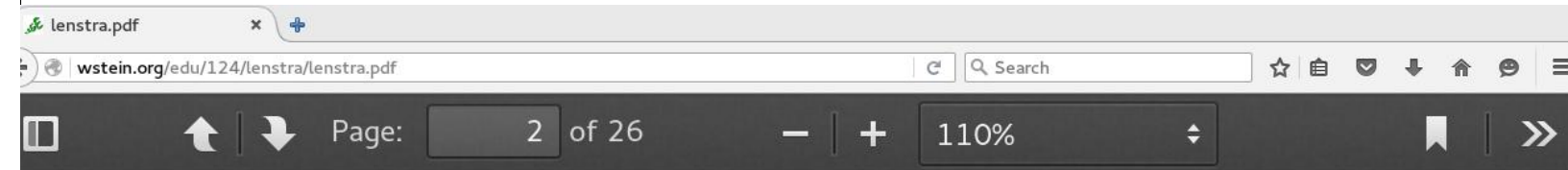
Acknowledgements. This paper was written at the Mathematisch Instituut, Universiteit Utrecht.

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).”

2

## Elliptic-curve computations

3



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

### Factoring integers with elliptic curves

By H. W. LENSTRA, JR.

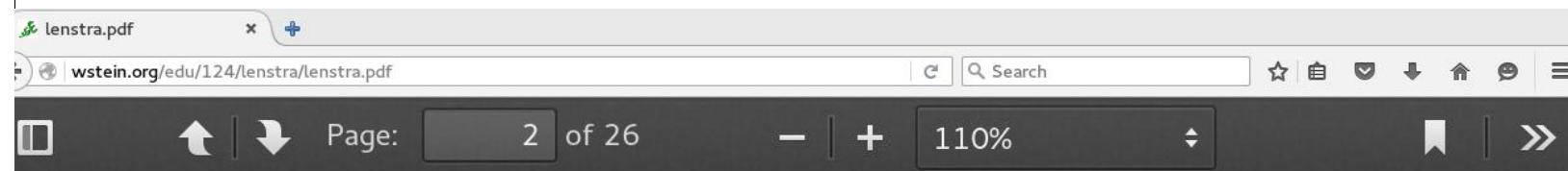
#### Abstract

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  $x \rightarrow \infty$ . 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  $n \rightarrow \infty$ ). 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

2  
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).”

## 3 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  $x \rightarrow \infty$ . 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  $n \rightarrow \infty$ ). 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

3  
1987 (di  
ECM, th  
of factor  
1985 Bo  
Kilian, 1  
Chudnov  
elliptic-c  
1985/6  
and inde  
1987 (di  
ECC—u  
to avoid

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

2

## Elliptic-curve computations

3

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

Page: 2 of 26 110%

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

### Factoring integers with elliptic curves

By H. W. LENSTRA, JR.

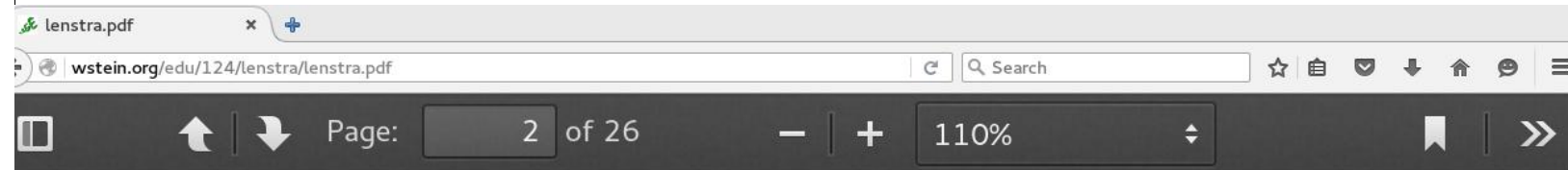
#### Abstract

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  $x \rightarrow \infty$ . 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  $n \rightarrow \infty$ ). 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 integers  
1985 Bosma, 1986  
Kilian, 1986 Chud  
Chudnovsky, 1988  
elliptic-curve prima  
1985/6 (distributed  
and independently  
1987 (distributed  
ECC—use elliptic  
to avoid index-cal

# 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  $x \rightarrow \infty$ . 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  $n \rightarrow \infty$ ). 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 1984) Lenstra  
ECM, the elliptic-curve method  
of factoring integers.

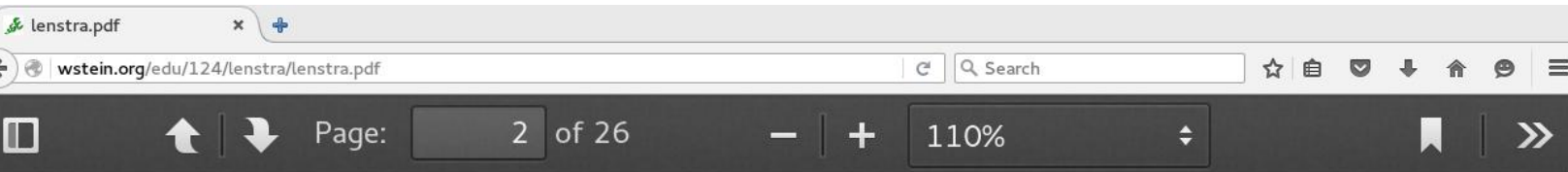
1985 Bosma, 1986 Goldwasser  
Kilian, 1986 Chudnovsky–  
Chudnovsky, 1988 Atkin: ECC  
elliptic-curve primality proving

1985/6 (distributed 1984) Montgomery  
and independently

1987 (distributed 1984) Kobayashi  
ECC—use elliptic curves in factoring  
to avoid index-calculus attacks



# 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  $x \rightarrow \infty$ . 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  $n \rightarrow \infty$ ). 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.

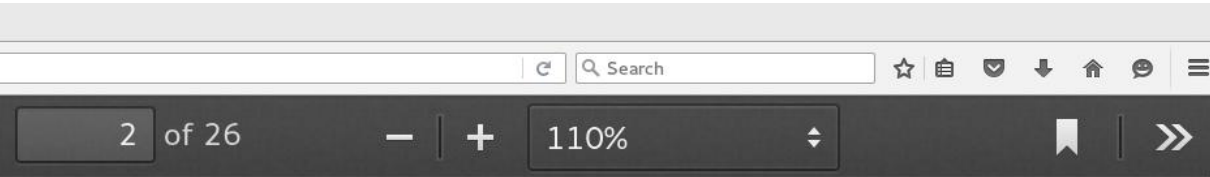
1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

1985/6 (distributed 1984) Miller, and independently

1987 (distributed 1984) Koblitz: ECC—use elliptic curves in DH to avoid index-calculus attacks.

# Curve Computations

3



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

## Factoring integers with elliptic curves

By H. W. LENSTRA, JR.

### Abstract

devoted to the description and analysis of a new algorithm to factor integers. It depends on the use of elliptic curves. The new method is based on Pollard's  $(p - 1)$ -method (Proc. Cambridge Philos. Soc. 76) but replaces the multiplicative group by the group of points on an elliptic curve. It is conjectured that the algorithm determines a prime factor of a composite number  $n$  in expected time at most  $O(\sqrt{(2 + o(1)) \log x \log \log x})$  for  $x \rightarrow \infty$ . In the worst case, for the product of two primes of the same order of magnitude, this is  $O(\sqrt{n \log \log n})$  (for  $n \rightarrow \infty$ ). There are several other factoring algorithms for which the conjectural expected running time is given by the latter expression. For these algorithms the running time is basically proportional to the size of the prime factors of  $n$ , whereas the new elliptic curve algorithm is potentially faster for small  $p$ .

Comments: This paper was written at the Mathematical Sciences

4

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

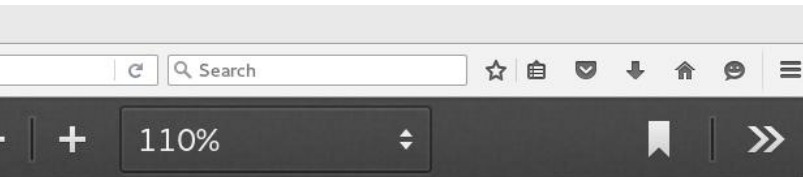
1985 Bosma, 1986 Goldwasser–Kilian, 1986 Chudnovsky–Chudnovsky, 1988 Atkin: ECPP, elliptic-curve primality proving.

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: ways to optimize ECM

# Computations



126 (1987), 649–673

## with elliptic curves

LENSTRA, JR.

Abstract

tion and analysis of a new algorithm to use of elliptic curves. The new method (Proc. Cambridge Philos. Soc. 76) multiplicative group by the group of points on a curve. It is proved that the algorithm determines a factor of  $n$  in expected time at most  $O(\sqrt{n})$  for  $n$  dividing  $n$  and  $K$  is a function for  $K = O(\log x)$  for  $x \rightarrow \infty$ . In the worst case, the running time is of the same order of magnitude, this is  $O(\sqrt{n})$ . There are several other factoring algorithms. The expected running time is given by the latter algorithm. They have a running time that is basically  $O(\sqrt{n})$  for factors of  $n$ , whereas the new elliptic curve

written at the Mathematical Sciences

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: a way to represent  $n$  and optimize # field o

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.

curves

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

Mathematical Sciences

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”.

(distributed 1984) Lenstra:  
the elliptic-curve method  
for factoring integers.

1986 Goldwasser–

1986 Chudnovsky–

Chudnovsky, 1988 Atkin: ECPP,  
elliptic curve primality proving.

(distributed 1984) Miller,  
independently

(distributed 1984) Koblitz:  
use elliptic curves in DH  
to resist index-calculus attacks.

4

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”.

5

Did Chu  
actually  
What ab  
What ab



4

1984) Lenstra:  
curve method  
rs.

5 Goldwasser–  
novsky–

Atkin: ECPP,  
ality proving.

and 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”.

5

Did Chudnovsky a  
actually recommen  
What about Mont  
What about paper

4

stra:  
 mod  
 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”.

5

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.

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

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

2000s: ANSI/IEEE/NIST  
 standards specify  $y^2 = x^3 - 3x + b$   
 in Weierstrass coordinates,  
 Chudnovsky–Chudnovsky.  
 Motivation: “the fastest  
 algorithm 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.

... laws on  $E$  exist. ...  
 ... consisting of bihomomorphisms ...  
 ... explicitly by Lang ...  
 ... there are complete ...  
 ... addition laws in ...

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

We can describe ...  
 ... the zero addition ...  
 ... call two addition ...

-Chudnovsky,  
analyze several  
elliptic curves;  
operations.

, for ECM:

$$y^2 = x^3 + Ax^2 + x,$$

$(-2)/4$  small.

/IEEE/NIST

$$y^2 = x^3 - 3x + b$$

nates,

-Chudnovsky.

n: "the fastest  
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.

Complete Systems of  
for Elliptic

W. BO

*Department of Pure Mathem  
Sydney, New South W*

AN

H. W. LEN

*Department of Mathematics  
Berkeley, Califor*

---

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

**THEOREM 1.** *The smallest cardinality of  
laws on  $E$  equals two, and if two addition  
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.

## 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 consisting of bihomogeneous polynomials of bidegree  $(2, 2)$ , was explicitly by Lange and Ruppert [2; cf. 1]. In the present paper there are complete systems consisting of *two* addition laws, and addition laws in such a system are necessarily of bidegree  $(2, 2)$ .

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

We can describe all addition laws of bidegree  $(2, 2)$ . To do this we consider the zero addition law, for which *all* pairs  $P_1, P_2$  are exceptional. We 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.

## 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  $d \in k$

---

Chudnovsky and Chudnovsky  
 recommend this?  
 about Montgomery?  
 about papers after 1987?

all known options

computing  $n, P \mapsto nP$

derivative elliptic curves.

Montgomery ladder is the fastest.

Elliptic-curve formulas

have exceptional cases.

Montgomery derives formulas for

inputs; for crypto we need

algorithms that *always* work.

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

Crypto 1

secret br

this leak

and Chudnovsky  
 and this?  
 Montgomery?  
 after 1987?  
 options  
 $P \mapsto nP$   
 elliptic curves.  
 er is the fastest.  
 curve formulas  
 tional cases.  
 es formulas for  
 crypto we need  
 ways work.

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  $E$  exist. 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 Koch  
 secret branches af  
 this leaks your sec

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.<sup>†</sup>

*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.

## 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.<sup>†</sup>

*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.

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.<sup>†</sup>

*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–Miyachi:  
timing attacks on DES.

### 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*

Indeed, a complete system of three addition laws, each con-  
sisting of homogeneous polynomials of bidegree  $(2, 2)$ , was exhibited  
by Lenstra and Ruppert [2; cf. 1]. In the present paper we show that  
complete systems consisting of two addition laws, and that both  
of which are necessarily of bidegree  $(2, 2)$ .

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

We consider all addition laws of bidegree  $(2, 2)$ . To do this, we omit  
the case of a law, for which all pairs  $P_1, P_2$  are exceptional, and we  
define two 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–Miyachi:  
timing attacks on DES.

“Guaran  
load ent

Two Addition Laws  
on Curves

OSMA\*

atics, University of Sydney,  
Wales 2006, Australia

D

STRA, JR.†

s, University of California,  
California 94720-3840

em 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  
pairs  $P_1, P_2$  are exceptional, and we  
ere 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–Miyachi:  
timing attacks on DES.

“Guaranteed” cou  
load entire table in



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–Miyachi:  
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–Miyachi:  
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–Miyachi:  
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–Miyachi:  
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:

branches affect timing;

is your secret key.

mentioned by Kocher

ESORICS 1998 Kelsey–

–Wagner–Hall:

array indices can affect

via cache misses.

ge, CHES 2003 Tsunoo–

uzaki–Shigeri–Miyauchi:

attacks on DES.

8

“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.

9

2016: O

CacheBleed: A Timing... x

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

CacheBleed  
Open

Yuva

Yaro

The Univer

Adelaide

NICTA

## Overview

**CacheBleed** is a side-channel attack that exploits cache-bank conflicts to leak secret data through minute timing variations in cryptographic processes running on modern processors (e.g., 2048-bit and 4096-bit RSA signatures). This is a carefully designed tool (and other) side-channel

While the possibility of this attack was speculated, this is the first technical documentation. However, these were common cryptographic countermeasures to

## Paper

Latest version can be

se!

er:

fect timing;

ret key.

by Kocher

1998 Kelsey–

-Hall:

s can affect

misses.

2003 Tsunoo–

geri–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

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

## CacheBleed: A Timing Attack on OpenSSL Constant Time

Yuval  
Yarom

The University of  
Adelaide and  
NICTA

Dan  
Genkin

Technion  
Aviv Univ

### Overview

**CacheBleed** is a side-channel attack that exploits cache-bank conflicts in Intel processors. By measuring minute timing variations, we are able to recover secret keys from processes running on the same machine. Our attack recovers 2048-bit and 4096-bit RSA secret keys from Intel Bridge processors after observing only 16,000 signatures (and other side-channel attacks).

While the possibility of an attack based on cache-bank conflicts was speculated, this is the first practical demonstration. Our technical documentation describes cache-bank conflicts. However, these were not widely thought to be a common cryptographic software developers’ countermeasures to this attack.

### Paper

Latest version can be downloaded [here](#).

“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 Attack on OpenSSL Constant Time

Yuval Yarom  
The University of Adelaide and NICTA

Daniel Genkin  
Technion and Tel Aviv University

Nathaniel Heninger  
University of Pennsylvania

## Overview

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

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

## Paper

Latest version can be downloaded [here](#).

“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.

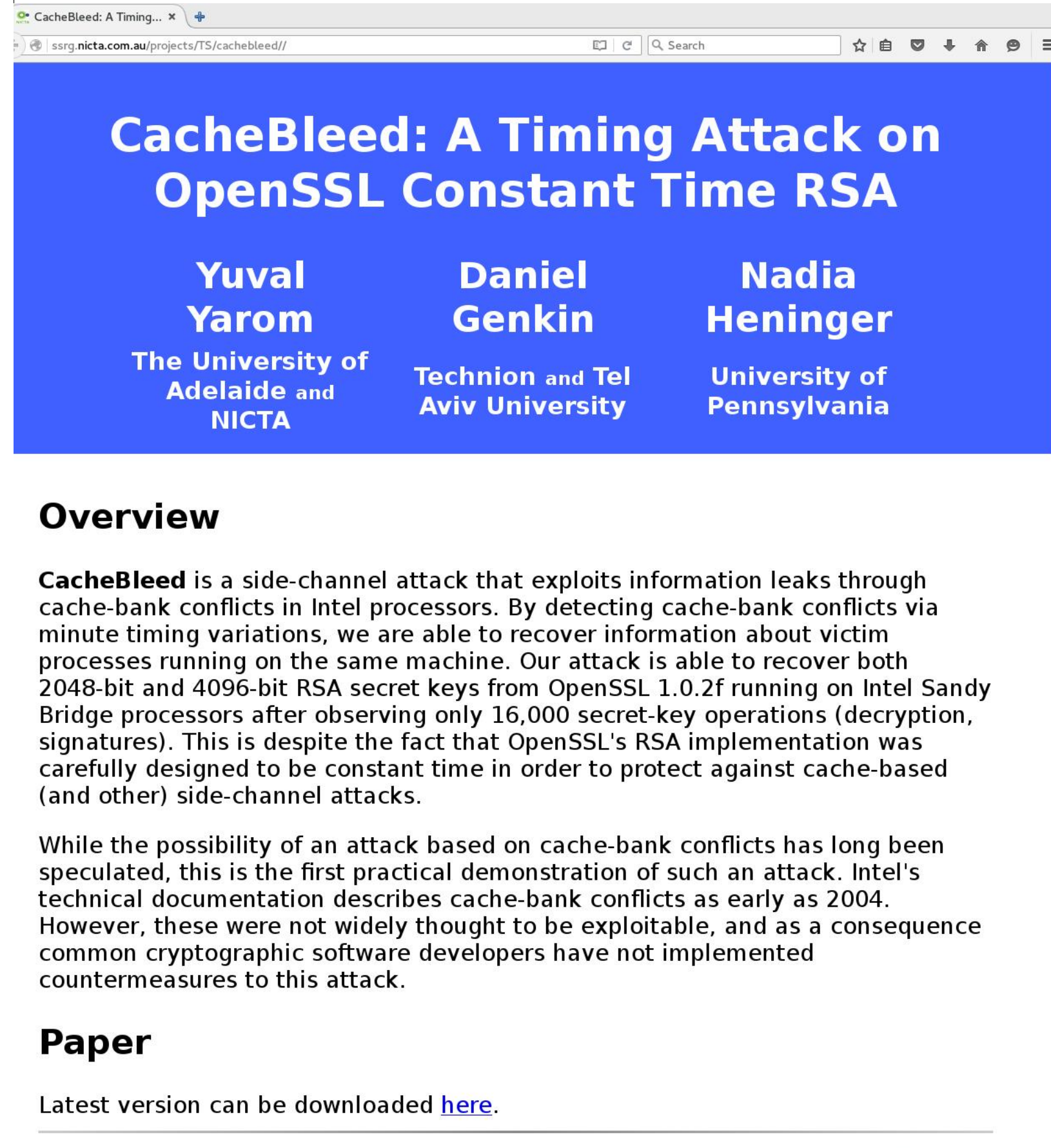
2013 Bernstein–Schwabe

“A word of warning” :

Cheaper countermeasure

recommended by Intel isn't safe.

2016: OpenSSL **didn't listen.**



CacheBleed: A Timing Attack on  
OpenSSL Constant Time RSA

<b>Yuval Yarom</b>	<b>Daniel Genkin</b>	<b>Nadia Heninger</b>
The University of Adelaide and NICTA	Technion and Tel Aviv University	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](#).



“steed” 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.

9

2016: OpenSSL didn't listen.

10

CacheBleed: A Timing Attack on  
OpenSSL Constant Time RSA

<b>Yuval Yarom</b> The University of Adelaide and NICTA	<b>Daniel Genkin</b> Technion and Tel Aviv University	<b>Nadia Heninger</b> 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 Cur

Avoid “a  
branches  
indices,  
with inp

intermeasure:

nto cache.

Bernstein:

AES.

sn't safe;

ndices can affect

ank collisions.

all data flow

ray indices.

chwabe

g":

measure

ntel isn't safe.

2016: OpenSSL didn't listen.

The Curve25519 p

Avoid "all input-d

branches, all input

indices, and other

with input-depend

CacheBleed: A Timing Attack on  
OpenSSL Constant Time RSA

<b>Yuval Yarom</b> The University of Adelaide and NICTA	<b>Daniel Genkin</b> Technion and Tel Aviv University	<b>Nadia Heninger</b> 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](#).

2016: OpenSSL **didn't listen**.

The Curve25519 paper

CacheBleed: A Timing Attack on  
OpenSSL Constant Time RSA

<b>Yuval Yarom</b>	<b>Daniel Genkin</b>	<b>Nadia Heninger</b>
The University of Adelaide and NICTA	Technion and Tel Aviv University	University of Pennsylvania

Avoid “all input-dependent branches, all input-dependent indices, and other instructions with input-dependent timing”

## Overview

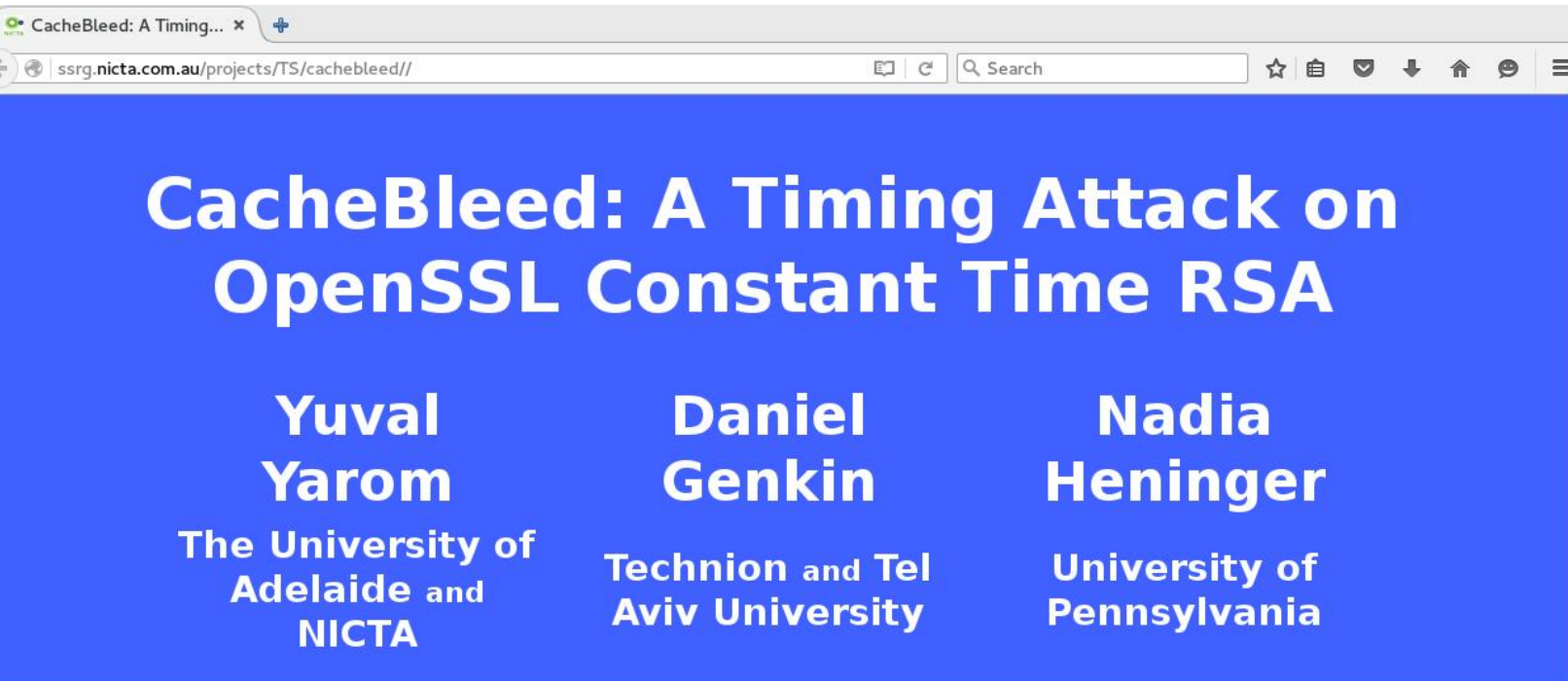
**CacheBleed** is a side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via minute timing variations, we are able to recover information about victim processes running on the same machine. Our attack is able to recover 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](#).

# 2016: OpenSSL didn't listen.



CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

<b>Yuval Yarom</b> The University of Adelaide and NICTA	<b>Daniel Genkin</b> Technion and Tel Aviv University	<b>Nadia Heninger</b> 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.

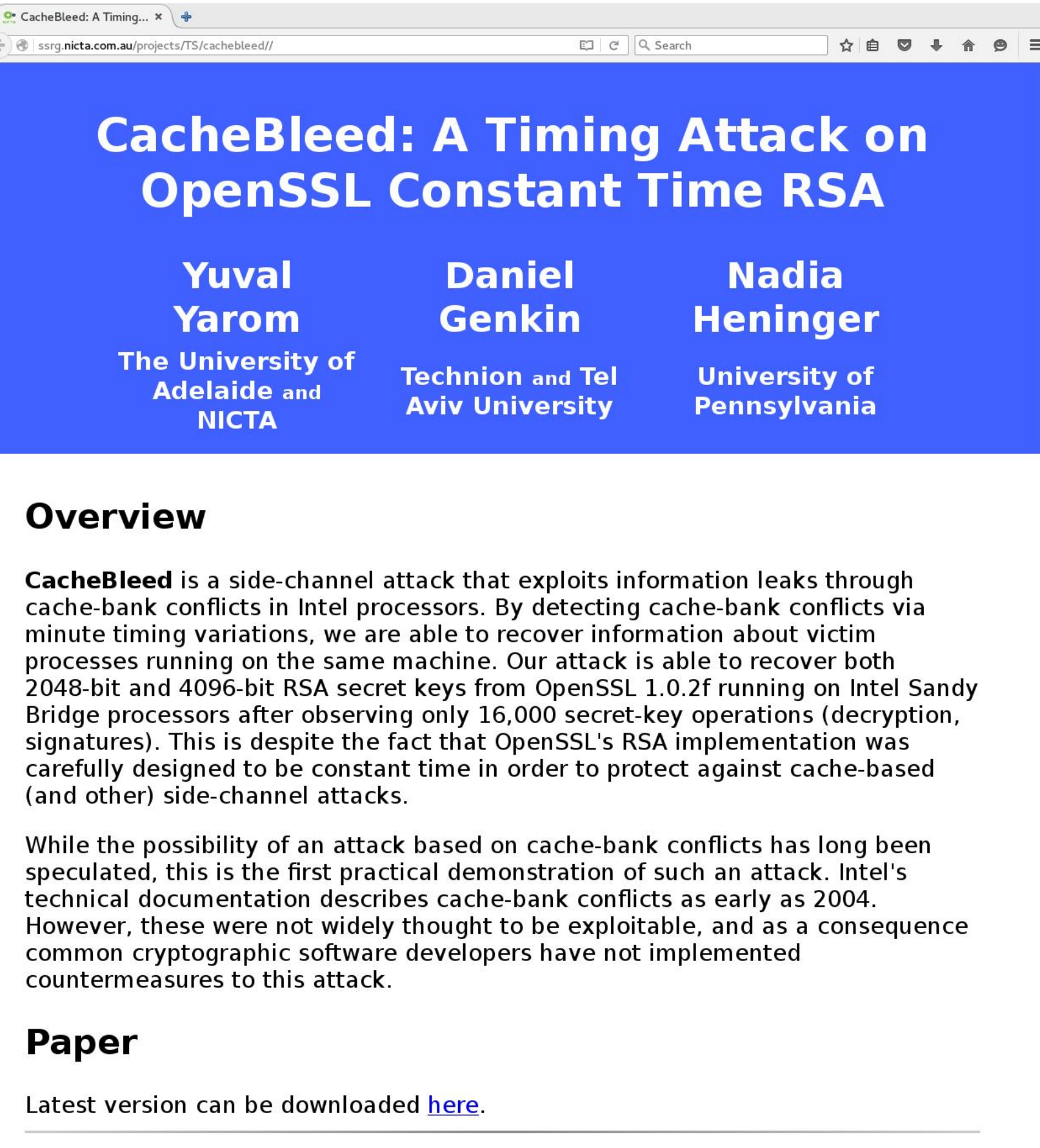
## Paper

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”.

# 2016: OpenSSL didn't listen.



CacheBleed: A Timing Attack on OpenSSL Constant Time RSA

<b>Yuval Yarom</b> The University of Adelaide and NICTA	<b>Daniel Genkin</b> Technion and Tel Aviv University	<b>Nadia Heninger</b> 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**

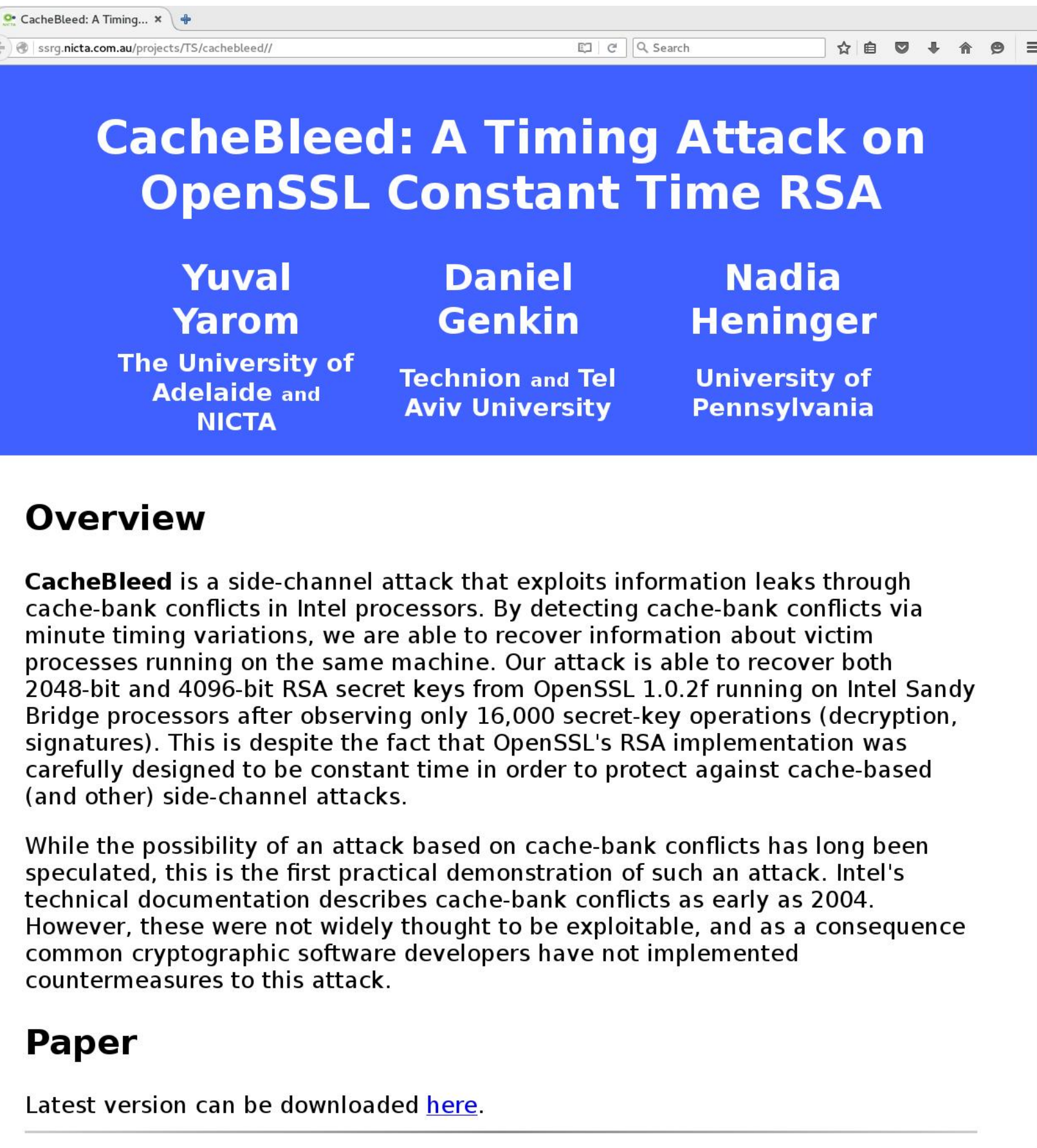
Latest version can be downloaded [here](#).

## 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 Attack on OpenSSL Constant Time RSA

<b>Yuval Yarom</b> The University of Adelaide and NICTA	<b>Daniel Genkin</b> Technion and Tel Aviv University	<b>Nadia Heninger</b> 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

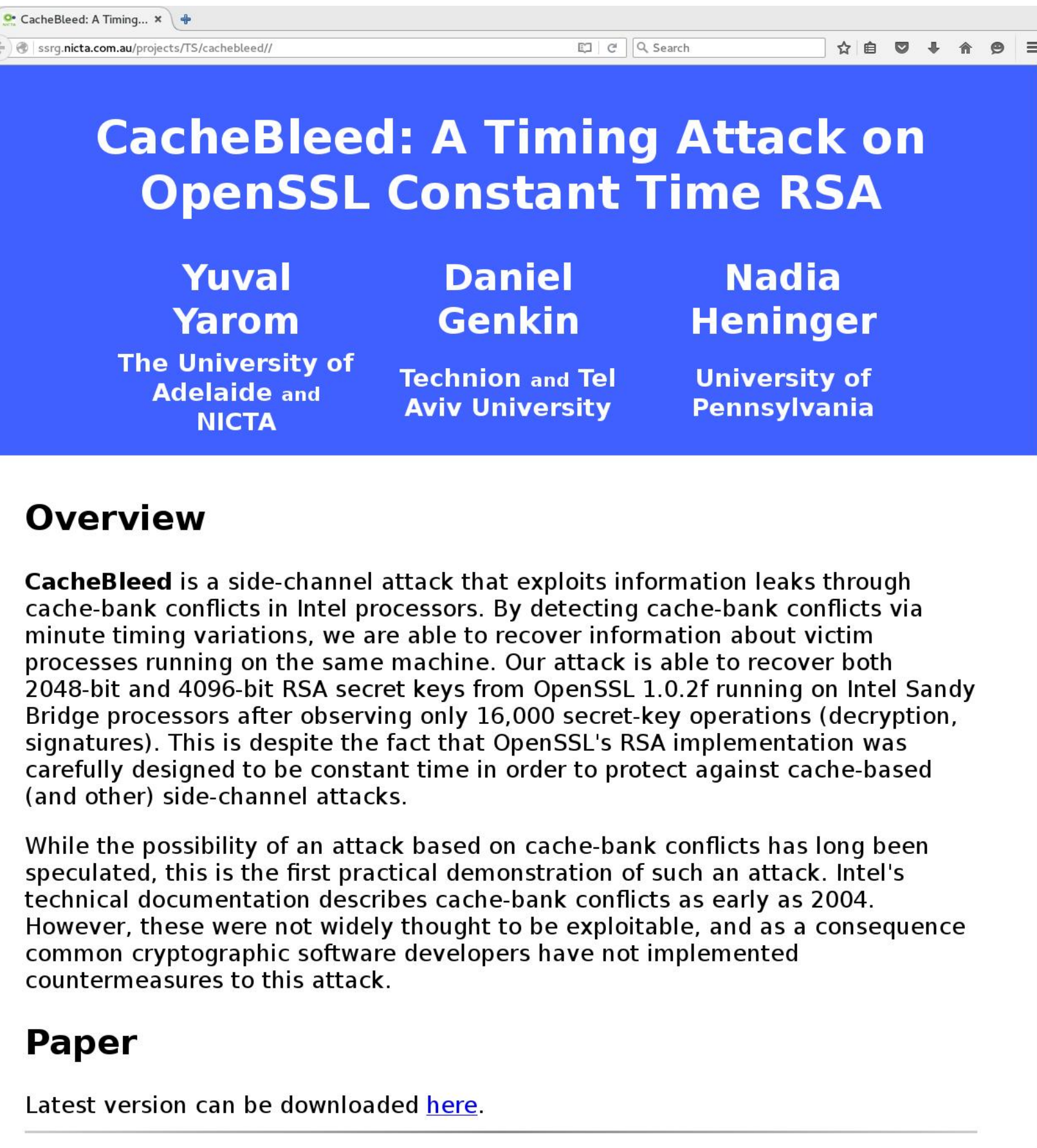
Avoid “all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings”.

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”.

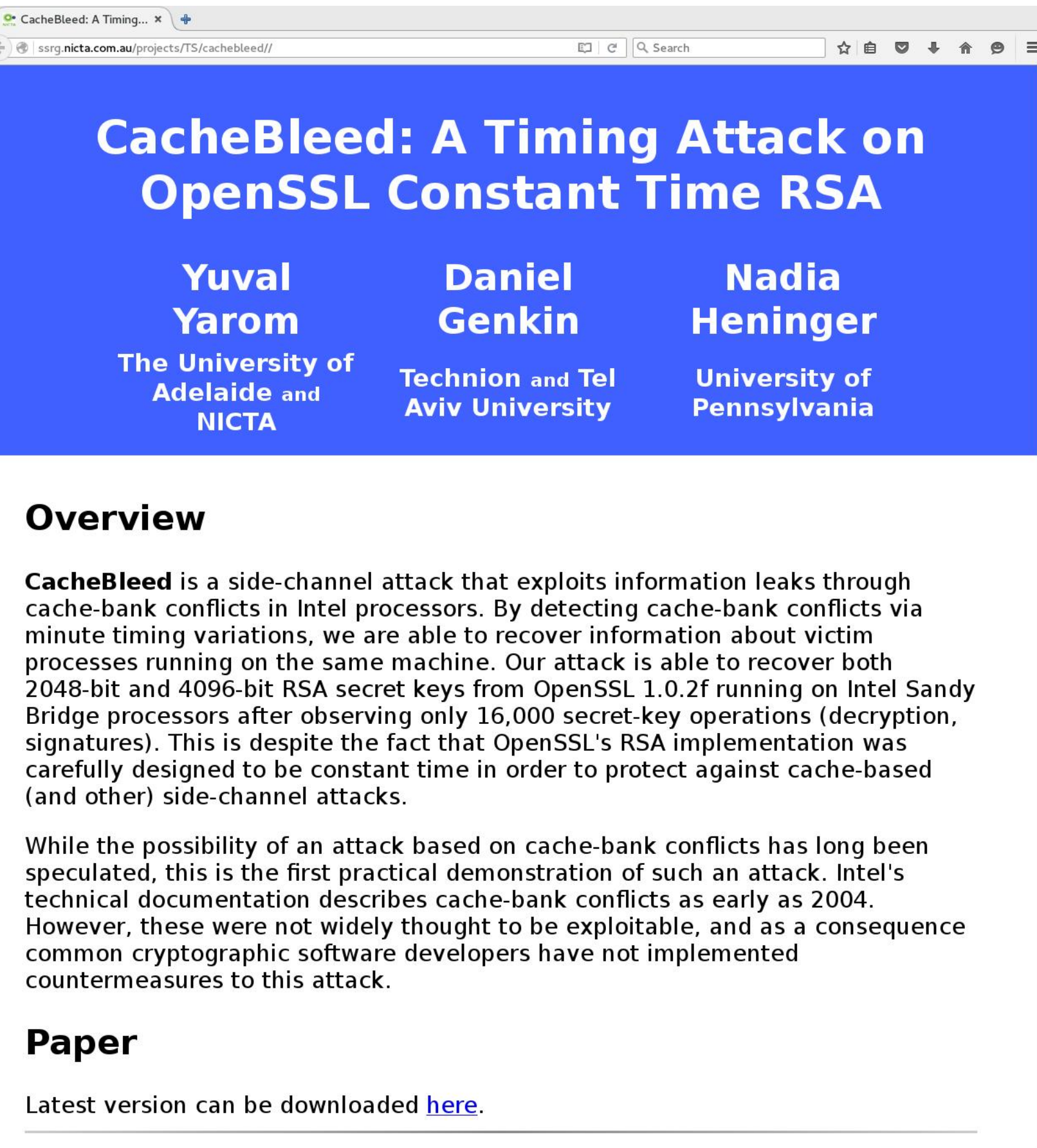
Choose a curve  $y^2 = x^3 + Ax^2 + x$  where  $A^2 - 4$  is not a square.

$\approx 25\%$  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.**

# 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  $X_0(nP)$ .



OpenSSL didn't listen.

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

**Daniel Genkin** (Technion and Tel Aviv University) and **Nadia Heninger** (University of Pennsylvania)

Side-channel attack that exploits information leaks through cache-bank conflicts in Intel processors. By detecting cache-bank conflicts via branch mispredictions, we are able to recover information about victim RSA secret keys from OpenSSL 1.0.2f running on Intel Sandy Bridge processors observing only 16,000 secret-key operations (decryption), despite the fact that OpenSSL's RSA implementation was designed to be constant time in order to protect against cache-based timing attacks.

This is the first practical demonstration of such an attack. Intel's documentation describes cache-bank conflicts as early as 2004. It was not widely thought to be exploitable, and as a consequence cryptographic software developers have not implemented defenses against this attack.

Full paper available [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  $X_0(nP)$ .

```
x2, z2, x3
for i in range(1, n):
    bit = (n - i) >> 1
    x2, x3 = x2 * x3
    z2, z3 = z2 * z3
    x3, z3 = x3 * z3
    x2, z2 = x2 * z2
    x2, z2 = x2 * z2 + 4 * x3
    x2, x3 = x2 * x3
    z2, z3 = z2 * z3
return x2, z2
```

didn't listen.

**Timing Attack on Constant Time RSA**

**Daniel Klein** and **Nadia Heninger**

University of Pennsylvania

exploits information leaks through detecting cache-bank conflicts via cover information about victim our attack is able to recover both private and public keys

OpenSSL 1.0.2f running on Intel Sandy Bridge processors

100 secret-key operations (decryption, encryption)

OpenSSL's RSA implementation was discovered in order to protect against cache-based attacks

Cache-bank conflicts has long been a source of information leakage

Discovery of such an attack. Intel's processors have not implemented mitigations for such attacks as early as 2004.

Cache-bank conflicts can be exploitable, and as a consequence mitigations have not implemented

## 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-
```

## 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) ^
              x1*(x2*z3 - z2*x3) ^
              x2, z2 = ((x2^2 - z2^2) ^ 2,
              4*x2*z2*(x2^2 + A*x2*z2) ^ 2)
    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.

≈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)
```

Montgon

dependin

11

paper

dependent

t-dependent array

instructions

ent timings".

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

ot a square.

c curves.

$$x; X_0(\infty) = 0.$$

nt  $P$  as  $X_0(P)$ .

ery ladder

a tests.

is  $X_0(nP)$ .

12

$$x2, z2, x3, z3 = 1, 0, x1, 1$$

```
for i in reversed(range(255)):
```

$$\text{bit} = 1 \ \& \ (n \gg i)$$

$$x2, x3 = \text{cswap}(x2, x3, \text{bit})$$

$$z2, z3 = \text{cswap}(z2, z3, \text{bit})$$

$$x3, z3 = ((x2*x3 - z2*z3)^2, \\ x1*(x2*z3 - z2*x3)^2)$$

$$x2, z2 = ((x2^2 - z2^2)^2,$$

$$4*x2*z2*(x2^2 + A*x2*z2 + z2^2))$$

$$x2, x3 = \text{cswap}(x2, x3, \text{bit})$$

$$z2, z3 = \text{cswap}(z2, z3, \text{bit})$$

```
return x2*z2^(p-2)
```

Montgomery has v

depending on top

11

```

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)

```

12

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 input

12

```

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.

13

“Hey, you forgot to  
the input is on the

(55)) :

)

)

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 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!”

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.

Publications - Ruhr-U... x

https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/

RUHR-UNIVERSITÄT BOCHUM A-Z | OVERVIEW | SEARCH | CONTACT

CHAIR FOR NETWORK AND DATA SECURITY

ETI 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 different 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

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

519: Change initialization  
to handle leading 0 bits.

constant #loops.

use scalars  $n$

that have leading 0 bits,

and a Montgomery ladder

that runs in constant time.

is symmetric to compute

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.

Curve25519

“free key

eliminate

No cost

no code

Publications - Ruhr-U... x

https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/

RUHR-UNIVERSITÄT BOCHUM A-Z | OVERVIEW | SEARCH | CONTACT

CHAIR FOR NETWORK AND DATA SECURITY

ETI FORSCHUNG

RUB

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 different 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

variable #loops,  
bit of  $n$ .

ange initialization  
bits.

ops.

$n$

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 validation  
eliminates these at  
No cost for checki  
no code to forget.

Publications - Ruhr-U... x

https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/

RUHR-UNIVERSITÄT BOCHUM A-Z | OVERVIEW | SEARCH | CONTACT

CHAIR FOR NETWORK AND DATA SECURITY

ETI 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 different 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

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.

The screenshot shows a web browser window with the URL <https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/>. The page header includes 'RUHR-UNIVERSITÄT BOCHUM' and 'CHAIR FOR NETWORK AND DATA SECURITY'. The main content area displays the title 'PRACTICAL INVALID CURVE ATTACKS ON TLS-ECDH' by Tibor Jager, Jörg Schwenk, and Juraj Somorovsky, awarded at ESORICS 2015. An abstract is visible, starting with 'Elliptic Curve Cryptography (ECC) is based on cyclic groups...'. A sidebar on the left lists navigation options like 'LEHRSTUHL', 'LEHRE', and 'courses'.

“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.

Publications - Ruhr-U... x

https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/

RUHR-UNIVERSITÄT BOCHUM A-Z | OVERVIEW | SEARCH | CONTACT

CHAIR FOR NETWORK AND DATA SECURITY

ETI 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 different 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

“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.

The screenshot shows a web browser window with the URL <https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/>. The page header includes 'RUHR-UNIVERSITÄT BOCHUM' and 'CHAIR FOR NETWORK AND DATA SECURITY'. The main content area displays the title 'PRACTICAL INVALID CURVE ATTACKS ON TLS-ECDH' and the authors 'Tibor Jager, Jörg Schwenk, Juraj Somorovsky'. It also mentions 'ESORICS 2015' and an 'ABSTRACT' section. On the left side, there is a navigation menu with items like 'LEHRSTUHL', 'LEHRE', 'Best Student Paper Award', 'HackerPraktikum', 'HackPra Allstars', 'Former speakers', and 'courses'.

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... x

https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/

RUHR-UNIVERSITÄT BOCHUM

CHAIR FOR NETWORK AND DATA SECURITY

FORSCHUNG

A-Z | OVERVIEW | SEARCH | CONTACT

RUB

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 different 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.

The screenshot shows a web browser window with the URL <https://www.nds.ruhr-uni-bochum.de/research/publications/ESORICS15/>. The page header includes 'RUHR-UNIVERSITÄT BOCHUM' and 'CHAIR FOR NETWORK AND DATA SECURITY'. The main content area displays the title 'PRACTICAL INVALID CURVE ATTACKS ON TLS-ECDH' and the authors 'Tibor Jager, Jörg Schwenk, Juraj Somorovsky'. It also mentions 'ESORICS 2015' and an 'ABSTRACT' section. A sidebar on the left lists navigation options like 'LEHRSTUHL', 'LEHRE', and 'courses'.

Curve25519 paper:

“free key validation”

eliminates these attacks.

No cost for checking input;  
no code to forget.

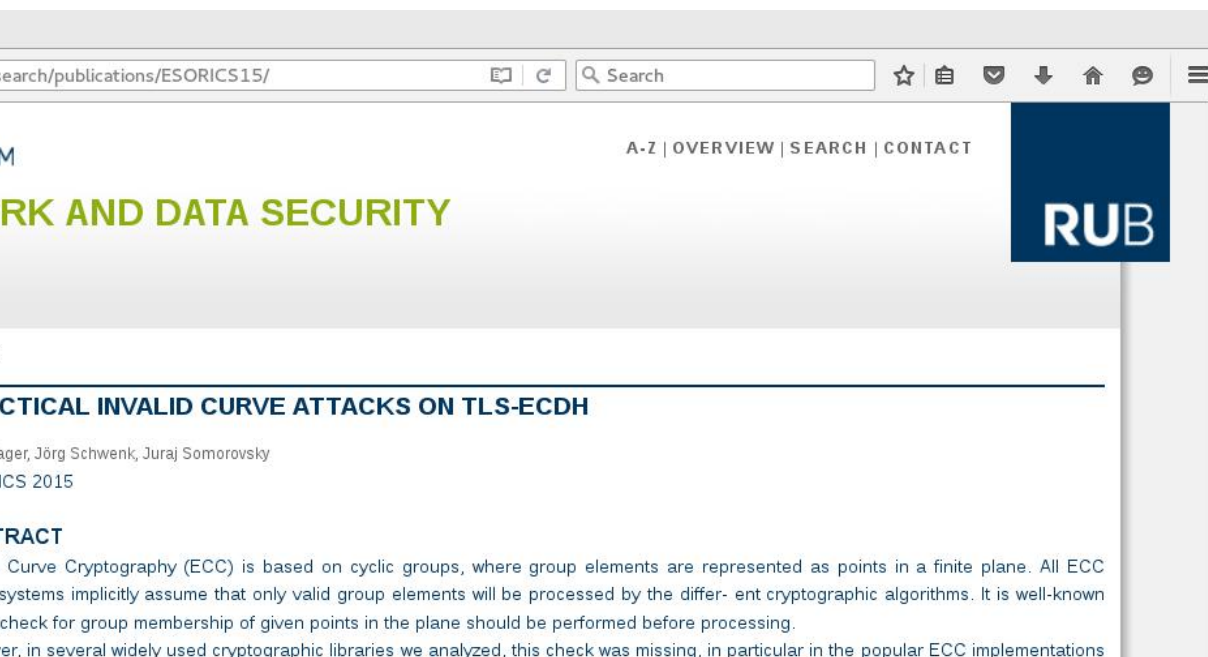
1. Montgomery naturally follows 1986 Miller compression: send only  $x$ -coordinate, not  $(x, y)$ . Forces input onto “curve” or “twist”. (Bonus: 32-byte keys!)
2. Montgomery ladder works correctly for inputs on twist.
3. Choose twist-secure curve.



you forgot to check that  
it 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.

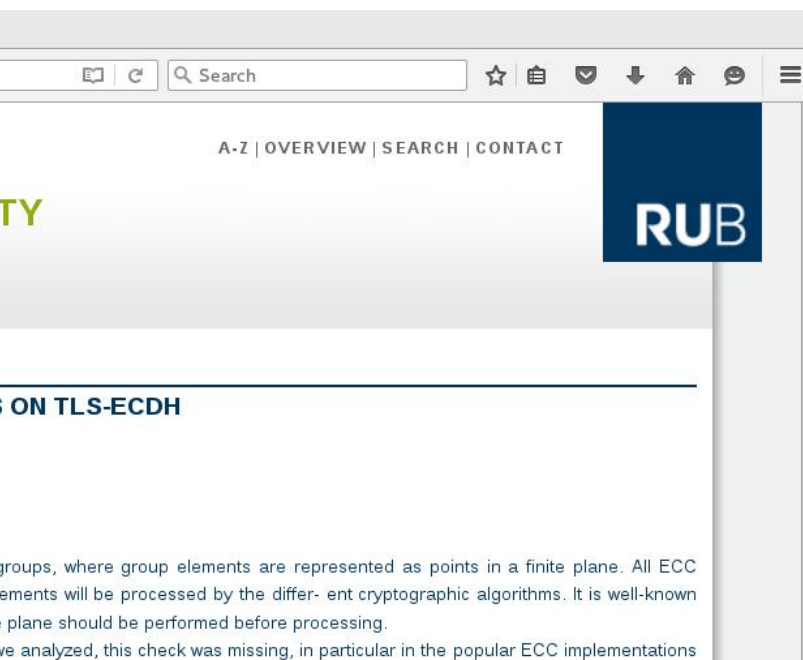
1. Montgomery naturally follows 1986 Miller compression: send only  $x$ -coordinate, not  $(x, y)$ . Forces input onto “curve” or “twist”. (Bonus: 32-byte keys!)
2. Montgomery ladder works correctly for inputs on twist.
3. Choose twist-secure curve.

Longest  
paper: f  
improvin  
from 199

to check that  
the curve!”

om: Important  
e broken by  
–Meyer–Müller.

ger–Schwenk–  
successful 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 algo  
from 1999–2004 B

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 arithm

improving on algorithm design

from 1999–2004 Bernstein.

mat

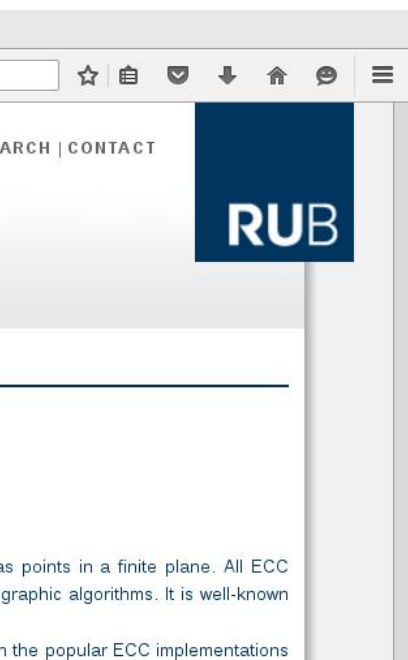
rtant

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 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:

“validation”

uses these attacks.

for checking input;

to forget.

Montgomery naturally

1986 Miller compression:

only  $x$ -coordinate, not  $(x, y)$ .

input onto “curve” or

(Bonus: 32-byte keys!)

Montgomery ladder works

for inputs on twist.

use 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”  
 attacks.  
 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 sys  
 1976 Diffie–Hellma  
 1999 Rescorla “sta  
 mode”; 2006 NIST



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  
fast finite-field arithmetic,  
on algorithm designs  
1999–2004 Bernstein.

mentioned in paper:  
programming language.

prime  $2^{255} - 19$ .

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

fields also have  
of minimizing the  
of security concerns for  
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.

2015: B

Weak Diffie-Hellman and... x

https://weakdh.org

## Weak D Logjam

Good News! You

Diffie-Hellman key  
allows Internet pro  
connection. It is fun  
IPsec, SMTPS, and

We have uncovered  
exchange has been

1. **Logjam attack**  
man-in-the-r  
connections  
attacker to r  
The attack is  
the TLS prot  
attacks a Dif  
exchange. Th

Curve25519  
field arithmetic,  
algorithm designs  
Bernstein.

in paper:  
language.

19.

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

have

minimizing the  
concerns for  
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: Beware batch

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

## Weak Diffie-He Logjam Attack

Good News! Your browser is safe again

Diffie-Hellman key exchange is a popular protocol that allows Internet protocols to agree on a shared secret for a connection. It is fundamental to many protocols, including IPsec, SMTPS, and protocols that rely on it.

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

1. **Logjam attack against the TLS protocol**  
A man-in-the-middle attacker to downgrade connections to 512-bit export-grade Diffie-Hellman, allowing the attacker to read and modify any data. The attack is reminiscent of the Logjam attack on the TLS protocol rather than an attack on the Diffie-Hellman key exchange. The attack affects any

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 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 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 it targets the TLS protocol rather than an implementation vulnerability. Logjam attacks a Diffie-Hellman key exchange rather than an implementation of the key exchange. The attack affects any server that supports

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 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 DHF 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... x  
 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 DHF EXPORT

Paper sk  
 attack m  
 composi  
 multi-us  
 (as in, e  
 “public-l  
 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:  
 tolerably loose.  
 leas: rho etc.  
 attacks.  
 hardware:  
 breakable.  
 attacks,  
 acks, etc.

## 2015: Beware batch attacks.

Weak Diffie-Hellman and... x

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 DHF EXPORT

Paper sketched co  
 attack model, incl  
 composition with s  
 multi-user secret-k  
 (as in, e.g., 2001  
 “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

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 DH 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 2001 Freire–Hofheinz–Kiltz–Paterdishonest 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... x

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 DH\_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.

# Diffie-Hellman and the Attack

your browser is safe against the Logjam attack.

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

There are several weaknesses in how Diffie-Hellman key exchange is deployed:

**Downgrade attack against the TLS protocol.** The Logjam attack allows a 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 Diffie-Hellman key exchange rather than an implementation vulnerability, and affects Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports DH\_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).

PKC 2006: Main Page x +

pkc06.cs.columbia.edu

**PKC 2006**

April 24-26, 2006  
New York City, USA



[front page](#)

[call for papers](#)

[local information](#)

[registration](#)

[program](#)

[contact](#)

golden sponsors




silver sponsors






ch attacks.

llman and the

ainst the Logjam attack.

lar cryptographic algorithm that  
a shared key and negotiate a secure  
protocols including HTTPS, SSH,  
on TLS.

es in how Diffie-Hellman key

protocol. The Logjam attack allows a  
downgrade vulnerable TLS  
grade cryptography. This allows the  
data passed over the connection.  
FREAK attack, but is due to a flaw in  
implementation vulnerability, and  
change rather than an RSA key  
server that supports DHF 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).

PKC 2006: Main Page x +  
pkc06.cs.columbia.edu

PKC 2006

April 24-26, 2006  
New York City, USA



[front page](#)

[call for papers](#)

[local information](#)

[registration](#)

[program](#)

[contact](#)

golden sponsors



Morgan Stanley

silver sponsors



GEMPLUS™

Google™

Microsoft

9th INTERNATIONAL CONFERENCE ON  
PUBLIC KEY CRYPTOGRAPHY

NEW YORK



The International Conference on Cryptography (PKC) has been held on all aspects of public-key cryptography. The proceedings are published by Springer-Verlag in the LNCS series.

PKC'06 will be hosted by Columbia University in the Davis Auditorium on the 4th Floor of the 48th Building at Columbia University.

and the

ck.

gorithm that  
negotiate a secure  
HTTPS, SSH,

man key


attack allows a  
le TLS  
This allows the  
e connection.  
s due to a flaw in  
nerability, and  
n RSA key  
ts DHF 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).







PKC 2006: Main Page x

pkc06.cs.columbia.edu Search


**PKC 2006**  
April 24-26, 2006  
New York City, USA



[front page](#)  
[call for papers](#)  
[local information](#)  
[registration](#)  
[program](#)  
[contact](#)  
[golden sponsors](#)

  
  
silver sponsors  
  
  
  


**PKC 2006**  
**9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF PUBLIC KEY CRYPTOGRAPHY**  
**NEW YORK**




The International Conference on Theory and Practice of Cryptography (PKC) has been the main IACR annual conference on all aspects of public-key cryptography. PKC has a long history of attracting world-renowned scientists in the area. The Proceedings of PKC are published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place in the Davis Auditorium on the 4th floor (campus level) of the [Davis Building](#) at Columbia University, in New York City.







[ [Sponsors](#) ]

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).

PKC 2006  
April 24-26, 2006  
New York City, USA




[front page](#)  
[call for papers](#)  
[local information](#)  
[registration](#)  
[program](#)  
[contact](#)  
golden sponsors

  
  
silver sponsors  
  
  
  


# PKC 2006

9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
**PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**



The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the [Schapiro CEPSR Building](#) at Columbia University, in New York City.


[\[ Sponsors \]](#)

sketched common-sense model, 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);  
 st key registrations  
 .g., Eurocrypt 2008  
 ltz–Shoup);  
 strings (allows modeling,  
 00 Biehl–Meyer–Müller).

PKC 2006: Main Page x +

pkc06.cs.columbia.edu Search

PKC 2006  
 April 24-26, 2006  
 New York City, USA




[front page](#)  
[call for papers](#)  
[local information](#)  
[registration](#)  
[program](#)  
[contact](#)  
[golden sponsors](#)

**PKC 2006**

**9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
 PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**



The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the [Schapiro CEPSR Building](#) at Columbia University, in New York City.

[ [Sponsors](#) ]

**EADS**  
**Morgan Stanley**  
 silver sponsors  
**NTT DoCoMo**  
**GEMPLUS**  
**Google**  
**Microsoft**

Email from  
 \_\_\_\_\_  
 It is my  
 that you  
 new Diff  
 records  
 PKC'06.



Common-sense

uding

subsequent

key system

Bernstein

nticators” );

key system

ven for

n PKC 2013

Kiltz–Paterson);

strations

rypt 2008

);

lows modeling,

Meyer–Müller).

PKC 2006

April 24-26, 2006  
New York City, USA



[front page](#)

[call for papers](#)

[local information](#)

[registration](#)

[program](#)

[contact](#)

[golden sponsors](#)



Morgan Stanley

silver sponsors



GEMPLUS™

Google™

Microsoft

# PKC 2006

9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
**PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK**

**APRIL 24-26**



The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the Schapiro CEPSR Building at Columbia University, in New York City.

[ [Sponsors](#) ]

Email from progra

It is my pleasur

that your paper

new Diffie–Hellm


records" was acc

PKC'06. Congratu

PKC 2006: Main Page x


pkc06.cs.columbia.edu Search

**PKC 2006**  
April 24-26, 2006  
New York City, USA



**PKC 2006**  
**9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**



[front page](#)  
[call for papers](#)  
[local information](#)  
[registration](#)  
[program](#)  
[contact](#)  
golden sponsors

**EADS**

**Morgan Stanley**

silver sponsors

**NTT DoCoMo**

**GEMPLUS**

**Google**

**Microsoft**

The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the Schapiro CEPSR Building at Columbia University, in New York City.

[ [Sponsors](#) ]

Email from program chairs:

---

It is my pleasure to inform you that your paper "Curve25519: a new Diffie-Hellman speed record" 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!


---

PKC 2006: Main Page

pkc06.cs.columbia.edu

PKC 2006


April 24-26, 2006  
New York City, USA



**PKC 2006**

**9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**



[front page](#)

[call for papers](#)


[local information](#)


[registration](#)

[program](#)


[contact](#)


[golden sponsors](#)


 EADS


 Morgan Stanley

[silver sponsors](#)

 NTT DoCoMo

 GEMPLUS

 Google

 Microsoft

The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the Schapiro CEPSR Building at Columbia University, in New York City.

[\[ Sponsors \]](#)

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.


---

PKC 2006: Main Page

pkc06.cs.columbia.edu

PKC 2006


April 24-26, 2006  
New York City, USA



**PKC 2006**

**9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**



[front page](#)

[call for papers](#)


[local information](#)


[registration](#)

[program](#)


[contact](#)


[golden sponsors](#)


 EADS


 Morgan Stanley

[silver sponsors](#)

 NTT DoCoMo

 GEMPLUS

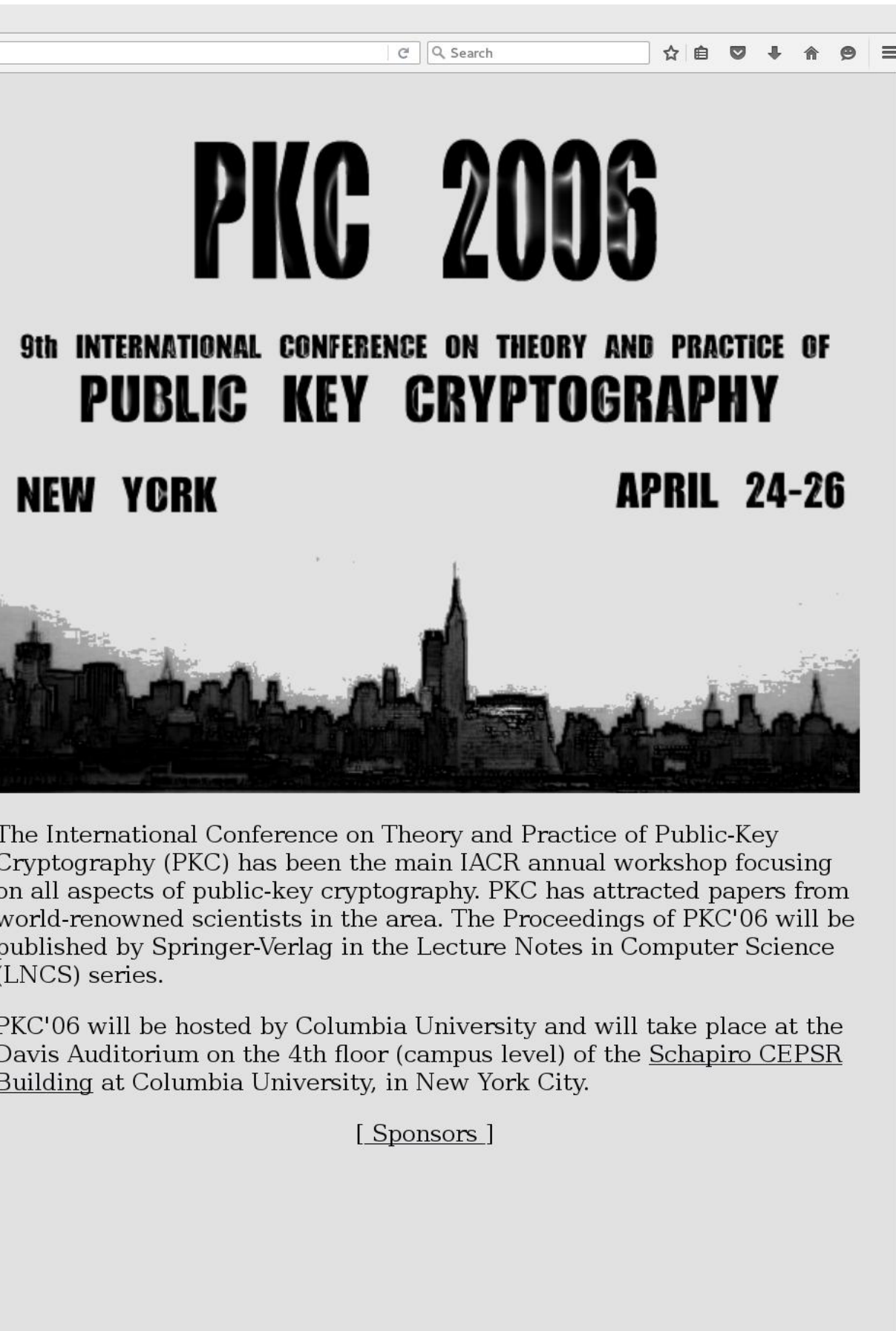
 Google

 Microsoft

The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the [Schapiro CEPSR Building](#) at Columbia University, in New York City.

[\[ Sponsors \]](#)



PKC 2006

9th INTERNATIONAL CONFERENCE ON THEORY AND PRACTICE OF  
**PUBLIC KEY CRYPTOGRAPHY**

**NEW YORK** **APRIL 24-26**

The International Conference on Theory and Practice of Public-Key Cryptography (PKC) has been the main IACR annual workshop focusing on all aspects of public-key cryptography. PKC has attracted papers from world-renowned scientists in the area. The Proceedings of PKC'06 will be published by Springer-Verlag in the Lecture Notes in Computer Science (LNCS) series.

PKC'06 will be hosted by Columbia University and will take place at the Davis Auditorium on the 4th floor (campus level) of the [Schapiro CEPSR Building](#) at Columbia University, in New York City.

[\[ Sponsors \]](#)

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.

---

Reviewers

---

While I  
this is  
I think  
"real"  
I don't  
correct  
the app  
paper to  
So engin

PKC 2006  
CONFERENCE ON THEORY AND PRACTICE OF  
PUBLIC-KEY CRYPTOGRAPHY  
APRIL 24-26

Conference on Theory and Practice of Public-Key  
Cryptography is the main IACR annual workshop focusing  
on public-key cryptography. PKC has attracted papers from  
all over the world in the area. The Proceedings of PKC'06 will be  
published in the Lecture Notes in Computer Science  
series by Springer-Verlag.

The conference is held at Columbia University and will take place at the  
11th floor (campus level) of the [Schapiro CEPSR](#)  
Building at Columbia University, in New York City.

[\[ Sponsors \]](#)

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) that this is a nice engineering paper, 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

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?

Reviewer

---

... bene  
 against  
 apparren  
 and very

On the r  
 does not  
 nor does  
 things :

"conject  
 through  
 a consio  
 achieven

---

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.

---

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 apparent patent infringement 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.

---

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.

---

Reviewer #1:

---

think (frankly) that  
a nice engineering work,  
that this is not a  
research paper.

question the  
ness but I question  
appropriateness of the  
to the conference.

---

Engineering isn't research?

Reviewer #2:

---

... benefits including protection  
against timing attacks, no  
apparent 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. "Bro  
function  
the share  
public ke  
extremel  
attack is  
performi  
on a typ  
cipher. .  
have ord  
a margin  
same co  
but this  
extra sp

---

rankly) that  
engineering work,  
is not a  
paper.

the  
I question  
ess of the  
ference.

---

't research?

Reviewer #2:

---

... benefits including protection  
against timing attacks, no  
apparent 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 exam  
the shared secret f  
public keys—is con  
extremely difficult.  
attack is more exp  
performing a brute  
on a typical 128-b  
cipher. ... Curves  
have order divisibl  
a marginally largen  
same conjectured  
but this is outweig  
extra speed of cur

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

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:

---

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

negative side, the paper  
 t introduce novel ideas,  
 s it attempt to prove  
 rigorously (the word  
 "structure" 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."

Reviewer

---

... The  
 hardware  
 which lo  
 if chang  
 ... My r  
 paper a  
 as low  
 mostly  
 very str  
 therefor  
 The pap

---

cluding protection  
attacks, 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

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 (mostly about one curve), is very strangely written, and is therefore unpleasant to read. 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

breaking the Curve25519  
 —for example, computing  
 ed secret from the two  
 keys—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  
 der divisible by 4, requiring  
 nally larger prime for the  
conjectured 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  
 incoheren  
 rewriting  
 to make  
 signific  
 someone  
 I'm not  
 be done  
 the con  
 "results  
 stated :  
 trivial  
 signific

e Curve25519  
 mple, computing  
 from the two  
conjectured to be  
 . Every known  
 expensive 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

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

significant (though easy for someone willing to do it)

I'm not optimistic that it can be done by the deadline,

but the content (I can't say "results" since there are

stated results, other than trivial mathematical results

significant enough to justify

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,  
 leaves 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.  
 appendic  
 For exam  
 discuss  
 either l  
 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

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 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 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 sho  
 prime sh  
 The pap  
 interest  
 Diffie-  
 curves.  
 the exp  
 y-coord  
 being us  
 ECC pro

a rambling  
 . . . The  
 ould be required  
 er readable is  
 ough 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

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  
le 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 great  
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.

mples, the irrelevant

tion of patents should

be removed, or rephrased

purely scientific

tion and not a patent

tion, 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



**RISK ASSE**

**Crypto flaw  
intentional**

Network tool contained

by Dan Goodin - Feb 2, 2016



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

The screenshot shows a web browser window with the URL `arstechnica.com/security/2016/02/crypto-flaw-was-so`. The page features the Ars Technica logo and navigation links for 'MAIN MENU' and 'MY STORIES: 25'. The main article title is 'Crypto flaw was so gl intentional eavesdro' with a subtitle 'Network tool contained hard-coded prime n'. The author is 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. At the bottom of the screenshot is a photograph of a neon sign that reads 'back OP' in a stylized, glowing orange font.

27  
 28  
 29  
 30  
 31  
 32  
 33  
 34  
 35  
 36  
 37  
 38  
 39  
 40  
 41  
 42  
 43  
 44  
 45  
 46  
 47  
 48  
 49  
 50  
 51  
 52  
 53  
 54  
 55  
 56  
 57  
 58  
 59  
 60  
 61  
 62  
 63  
 64  
 65  
 66  
 67  
 68  
 69  
 70  
 71  
 72  
 73  
 74  
 75  
 76  
 77  
 78  
 79  
 80  
 81  
 82  
 83  
 84  
 85  
 86  
 87  
 88  
 89  
 90  
 91  
 92  
 93  
 94  
 95  
 96  
 97  
 98  
 99  
 100  
 101  
 102  
 103  
 104  
 105  
 106  
 107  
 108  
 109  
 110  
 111  
 112  
 113  
 114  
 115  
 116  
 117  
 118  
 119  
 120  
 121  
 122  
 123  
 124  
 125  
 126  
 127  
 128  
 129  
 130  
 131  
 132  
 133  
 134  
 135  
 136  
 137  
 138  
 139  
 140  
 141  
 142  
 143  
 144  
 145  
 146  
 147  
 148  
 149  
 150  
 151  
 152  
 153  
 154  
 155  
 156  
 157  
 158  
 159  
 160  
 161  
 162  
 163  
 164  
 165  
 166  
 167  
 168  
 169  
 170  
 171  
 172  
 173  
 174  
 175  
 176  
 177  
 178  
 179  
 180  
 181  
 182  
 183  
 184  
 185  
 186  
 187  
 188  
 189  
 190  
 191  
 192  
 193  
 194  
 195  
 196  
 197  
 198  
 199  
 200  
 201  
 202  
 203  
 204  
 205  
 206  
 207  
 208  
 209  
 210  
 211  
 212  
 213  
 214  
 215  
 216  
 217  
 218  
 219  
 220  
 221  
 222  
 223  
 224  
 225  
 226  
 227  
 228  
 229  
 230  
 231  
 232  
 233  
 234  
 235  
 236  
 237  
 238  
 239  
 240  
 241  
 242  
 243  
 244  
 245  
 246  
 247  
 248  
 249  
 250  
 251  
 252  
 253  
 254  
 255  
 256  
 257  
 258  
 259  
 260  
 261  
 262  
 263  
 264  
 265  
 266  
 267  
 268  
 269  
 270  
 271  
 272  
 273  
 274  
 275  
 276  
 277  
 278  
 279  
 280  
 281  
 282  
 283  
 284  
 285  
 286  
 287  
 288  
 289  
 290  
 291  
 292  
 293  
 294  
 295  
 296  
 297  
 298  
 299  
 300  
 301  
 302  
 303  
 304  
 305  
 306  
 307  
 308  
 309  
 310  
 311  
 312  
 313  
 314  
 315  
 316  
 317  
 318  
 319  
 320  
 321  
 322  
 323  
 324  
 325  
 326  
 327  
 328  
 329  
 330  
 331  
 332  
 333  
 334  
 335  
 336  
 337  
 338  
 339  
 340  
 341  
 342  
 343  
 344  
 345  
 346  
 347  
 348  
 349  
 350  
 351  
 352  
 353  
 354  
 355  
 356  
 357  
 358  
 359  
 360  
 361  
 362  
 363  
 364  
 365  
 366  
 367  
 368  
 369  
 370  
 371  
 372  
 373  
 374  
 375  
 376  
 377  
 378  
 379  
 380  
 381  
 382  
 383  
 384  
 385  
 386  
 387  
 388  
 389  
 390  
 391  
 392  
 393  
 394  
 395  
 396  
 397  
 398  
 399  
 400  
 401  
 402  
 403  
 404  
 405  
 406  
 407  
 408  
 409  
 410  
 411  
 412  
 413  
 414  
 415  
 416  
 417  
 418  
 419  
 420  
 421  
 422  
 423  
 424  
 425  
 426  
 427  
 428  
 429  
 430  
 431  
 432  
 433  
 434  
 435  
 436  
 437  
 438  
 439  
 440  
 441  
 442  
 443  
 444  
 445  
 446  
 447  
 448  
 449  
 450  
 451  
 452  
 453  
 454  
 455  
 456  
 457  
 458  
 459  
 460  
 461  
 462  
 463  
 464  
 465  
 466  
 467  
 468  
 469  
 470  
 471  
 472  
 473  
 474  
 475  
 476  
 477  
 478  
 479  
 480  
 481  
 482  
 483  
 484  
 485  
 486  
 487  
 488  
 489  
 490  
 491  
 492  
 493  
 494  
 495  
 496  
 497  
 498  
 499  
 500  
 501  
 502  
 503  
 504  
 505  
 506  
 507  
 508  
 509  
 510  
 511  
 512  
 513  
 514  
 515  
 516  
 517  
 518  
 519  
 520  
 521  
 522  
 523  
 524  
 525  
 526  
 527  
 528  
 529  
 530  
 531  
 532  
 533  
 534  
 535  
 536  
 537  
 538  
 539  
 540  
 541  
 542  
 543  
 544  
 545  
 546  
 547  
 548  
 549  
 550  
 551  
 552  
 553  
 554  
 555  
 556  
 557  
 558  
 559  
 560  
 561  
 562  
 563  
 564  
 565  
 566  
 567  
 568  
 569  
 570  
 571  
 572  
 573  
 574  
 575  
 576  
 577  
 578  
 579  
 580  
 581  
 582  
 583  
 584  
 585  
 586  
 587  
 588  
 589  
 590  
 591  
 592  
 593  
 594  
 595  
 596  
 597  
 598  
 599  
 600  
 601  
 602  
 603  
 604  
 605  
 606  
 607  
 608  
 609  
 610  
 611  
 612  
 613  
 614  
 615  
 616  
 617  
 618  
 619  
 620  
 621  
 622  
 623  
 624  
 625  
 626  
 627  
 628  
 629  
 630  
 631  
 632  
 633  
 634  
 635  
 636  
 637  
 638  
 639  
 640  
 641  
 642  
 643  
 644  
 645  
 646  
 647  
 648  
 649  
 650  
 651  
 652  
 653  
 654  
 655  
 656  
 657  
 658  
 659  
 660  
 661  
 662  
 663  
 664  
 665  
 666  
 667  
 668  
 669  
 670  
 671  
 672  
 673  
 674  
 675  
 676  
 677  
 678  
 679  
 680  
 681  
 682  
 683  
 684  
 685  
 686  
 687  
 688  
 689  
 690  
 691  
 692  
 693  
 694  
 695  
 696  
 697  
 698  
 699  
 700  
 701  
 702  
 703  
 704  
 705  
 706  
 707  
 708  
 709  
 710  
 711  
 712  
 713  
 714  
 715  
 716  
 717  
 718  
 719  
 720  
 721  
 722  
 723  
 724  
 725  
 726  
 727  
 728  
 729  
 730  
 731  
 732  
 733  
 734  
 735  
 736  
 737  
 738  
 739  
 740  
 741  
 742  
 743  
 744  
 745  
 746  
 747  
 748  
 749  
 750  
 751  
 752  
 753  
 754  
 755  
 756  
 757  
 758  
 759  
 760  
 761  
 762  
 763  
 764  
 765  
 766  
 767  
 768  
 769  
 770  
 771  
 772  
 773  
 774  
 775  
 776  
 777  
 778  
 779  
 780  
 781  
 782  
 783  
 784  
 785  
 786  
 787  
 788  
 789  
 790  
 791  
 792  
 793  
 794  
 795  
 796  
 797  
 798  
 799  
 800  
 801  
 802  
 803  
 804  
 805  
 806  
 807  
 808  
 809  
 810  
 811  
 812  
 813  
 814  
 815  
 816  
 817  
 818  
 819  
 820  
 821  
 822  
 823  
 824  
 825  
 826  
 827  
 828  
 829  
 830  
 831  
 832  
 833  
 834  
 835  
 836  
 837  
 838  
 839  
 840  
 841  
 842  
 843  
 844  
 845  
 846  
 847  
 848  
 849  
 850  
 851  
 852  
 853  
 854  
 855  
 856  
 857  
 858  
 859  
 860  
 861  
 862  
 863  
 864  
 865  
 866  
 867  
 868  
 869  
 870  
 871  
 872  
 873  
 874  
 875  
 876  
 877  
 878  
 879  
 880  
 881  
 882  
 883  
 884  
 885  
 886  
 887  
 888  
 889  
 890  
 891  
 892  
 893  
 894  
 895  
 896  
 897  
 898  
 899  
 900  
 901  
 902  
 903  
 904  
 905  
 906  
 907  
 908  
 909  
 910  
 911  
 912  
 913  
 914  
 915  
 916  
 917  
 918  
 919  
 920  
 921  
 922  
 923  
 924  
 925  
 926  
 927  
 928  
 929  
 930  
 931  
 932  
 933  
 934  
 935  
 936  
 937  
 938  
 939  
 940  
 941  
 942  
 943  
 944  
 945  
 946  
 947  
 948  
 949  
 950  
 951  
 952  
 953  
 954  
 955  
 956  
 957  
 958  
 959  
 960  
 961  
 962  
 963  
 964  
 965  
 966  
 967  
 968  
 969  
 970  
 971  
 972  
 973  
 974  
 975  
 976  
 977  
 978  
 979  
 980  
 981  
 982  
 983  
 984  
 985  
 986  
 987  
 988  
 989  
 990  
 991  
 992  
 993  
 994  
 995  
 996  
 997  
 998  
 999  
 1000  
 1001  
 1002  
 1003  
 1004  
 1005  
 1006  
 1007  
 1008  
 1009  
 1010  
 1011  
 1012  
 1013  
 1014  
 1015  
 1016  
 1017  
 1018  
 1019  
 1020  
 1021  
 1022  
 1023  
 1024  
 1025  
 1026  
 1027  
 1028  
 1029  
 1030  
 1031  
 1032  
 1033  
 1034  
 1035  
 1036  
 1037  
 1038  
 1039  
 1040  
 1041  
 1042  
 1043  
 1044  
 1045  
 1046  
 1047  
 1048  
 1049  
 1050  
 1051  
 1052  
 1053  
 1054  
 1055  
 1056  
 1057  
 1058  
 1059  
 1060  
 1061  
 1062  
 1063  
 1064  
 1065  
 1066  
 1067  
 1068  
 1069  
 1070  
 1071  
 1072  
 1073  
 1074  
 1075  
 1076  
 1077  
 1078  
 1079  
 1080  
 1081  
 1082  
 1083  
 1084  
 1085  
 1086  
 1087  
 1088  
 1089  
 1090  
 1091  
 1092  
 1093  
 1094  
 1095  
 1096  
 1097  
 1098  
 1099  
 1100  
 1101  
 1102  
 1103  
 1104  
 1105  
 1106  
 1107  
 1108  
 1109  
 1110  
 1111  
 1112  
 1113  
 1114  
 1115  
 1116  
 1117  
 1118  
 1119  
 1120  
 1121  
 1122  
 1123  
 1124  
 1125  
 1126  
 1127  
 1128  
 1129  
 1130  
 1131  
 1132  
 1133  
 1134  
 1135  
 1136  
 1137  
 1138  
 1139  
 1140  
 1141  
 1142  
 1143  
 1144  
 1145  
 1146  
 1147  
 1148  
 1149  
 1150  
 1151  
 1152  
 1153  
 1154  
 1155  
 1156  
 1157  
 1158  
 1159  
 1160  
 1161  
 1162  
 1163  
 1164  
 1165  
 1166  
 1167  
 1168  
 1169  
 1170  
 1171  
 1172  
 1173  
 1174  
 1175  
 1176  
 1177  
 1178  
 1179  
 1180  
 1181  
 1182  
 1183  
 1184  
 1185  
 1186  
 1187  
 1188  
 1189  
 1190  
 1191  
 1192  
 1193  
 1194  
 1195  
 1196  
 1197  
 1198  
 1199  
 1200  
 1201  
 1202  
 1203  
 1204  
 1205  
 1206  
 1207  
 1208  
 1209  
 1210  
 1211  
 1212  
 1213  
 1214  
 1215  
 1216  
 1217  
 1218  
 1219  
 1220  
 1221  
 1222  
 1223  
 1224  
 1225  
 1226  
 1227  
 1228  
 1229  
 1230  
 1231  
 1232  
 1233  
 1234  
 1235  
 1236  
 1237  
 1238  
 1239  
 1240  
 1241  
 1242  
 1243  
 1244  
 1245  
 1246  
 1247  
 1248  
 1249  
 1250  
 1251  
 1252  
 1253  
 1254  
 1255  
 1256  
 1257  
 1258  
 1259  
 1260  
 1261  
 1262  
 1263  
 1264  
 1265  
 1266  
 1267  
 1268  
 1269  
 1270  
 1271  
 1272  
 1273  
 1274  
 1275  
 1276  
 1277  
 1278  
 1279  
 1280  
 1281  
 1282  
 1283  
 1284  
 1285  
 1286  
 1287  
 1288  
 1289  
 1290  
 1291  
 1292  
 1293  
 1294  
 1295  
 1296  
 1297  
 1298  
 1299  
 1300  
 1301  
 1302  
 1303  
 1304  
 1305  
 1306  
 1307  
 1308  
 1309  
 1310  
 1311  
 1312  
 1313  
 1314  
 1315  
 1316  
 1317  
 1318  
 1319  
 1320  
 1321  
 1322  
 1323  
 1324  
 1325  
 1326  
 1327  
 1328  
 1329  
 1330  
 1331  
 1332  
 1333  
 1334  
 1335  
 1336  
 1337  
 1338  
 1339  
 1340  
 1341  
 1342  
 1343  
 1344  
 1345  
 1346  
 1347  
 1348  
 1349  
 1350  
 1351  
 1352  
 1353  
 1354  
 1355  
 1356  
 1357  
 1358  
 1359  
 1360  
 1361  
 1362  
 1363  
 1364  
 1365  
 1366  
 1367  
 1368  
 1369  
 1370  
 1371  
 1372  
 1373  
 1374  
 1375  
 1376  
 1377  
 1378  
 1379  
 1380  
 1381  
 1382  
 1383  
 1384  
 1385  
 1386  
 1387  
 1388  
 1389  
 1390  
 1391  
 1392  
 1393  
 1394  
 1395  
 1396  
 1397  
 1398  
 1399  
 1400  
 1401  
 1402  
 1403  
 1404  
 1405  
 1406  
 1407  
 1408  
 1409  
 1410  
 1411  
 1412  
 1413  
 1414  
 1415  
 1416  
 1417  
 1418  
 1419  
 1420  
 1421  
 1422  
 1423  
 1424  
 1425  
 1426  
 1427  
 1428  
 1429  
 1430  
 1431  
 1432  
 1433  
 1434  
 1435  
 1436  
 1437  
 1438  
 1439  
 1440  
 1441  
 1442  
 1443  
 1444  
 1445  
 1446  
 1447  
 1448  
 1449  
 1450  
 1451  
 1452  
 1453  
 1454  
 1455  
 1456  
 1457  
 1458  
 1459  
 1460  
 1461  
 1462  
 1463  
 1464  
 1465  
 1466  
 1467  
 1468  
 1469  
 1470  
 1471  
 1472  
 1473  
 1474  
 1475  
 1476  
 1477  
 1478  
 1479  
 1480  
 1481  
 1482  
 1483  
 1484  
 1485  
 1486  
 1487  
 1488  
 1489  
 1490  
 1491  
 1492  
 1493  
 1494  
 1495  
 1496  
 1497  
 1498  
 1499  
 1500  
 1501  
 1502  
 1503  
 1504  
 1505  
 1506  
 1507  
 1508  
 1509  
 1510  
 1511  
 15

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” .

The image is a screenshot of a web browser displaying an article on the Ars Technica website. The browser's address bar shows the URL: `arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor`. The page features the Ars Technica logo at the top left, followed by navigation links for 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. The main heading of the article is 'RISK ASSESSMENT / SECURITY & HACKTIVISM'. The article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor', with a sub-headline: 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is identified as 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. Social media sharing buttons for Facebook, Twitter, and Email are visible. At the bottom of the screenshot, there is a photograph of a neon sign that reads 'back door OPEN' in a stylized, glowing orange-red font.

ows that 3 numbers are  
 should 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  
 protocols. ... The paper is  
 bly free of grammatical

2016: Counterfeit “primes”.

With rev  
 how did

The image is a screenshot of a web browser displaying an article on the Ars Technica website. The browser's address bar shows the URL: `arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor`. The page features the Ars Technica logo at the top left, followed by navigation links: 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. Below the navigation is a section header: 'RISK ASSESSMENT / SECURITY & HACKTIVISM'. The main article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor', with a subtitle: 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is listed as 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. There are social media sharing buttons for Facebook, Twitter, and Email. At the bottom of the screenshot is a photograph of a neon sign that reads 'back door OPEN' in a stylized, glowing orange-red font.

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".

With reviews like t  
 how did PKC acce

Crypto flaw was so g... x

arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropp... Search

ars technica


MAIN MENU MY STORIES: 25 FORUMS SUBSCRIBE JOBS

RISK ASSESSMENT / SECURITY & HACKTIVISM

**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



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

# 2016: Counterfeit "primes".

With reviews like these,  
how did PKC accept Curve2

The image is a screenshot of a web browser displaying an article on the Ars Technica website. The browser's address bar shows the URL: `arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropp`. The page features the Ars Technica logo at the top left, with navigation links for 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. The article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor', with a subtitle: 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is listed as 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. Social media sharing buttons for Facebook, Twitter, and Email are visible. At the bottom of the screenshot is a photograph of a neon sign that reads 'back door OPEN' in a stylized, glowing orange font.



2016: Counterfeit “primes” .

Crypto flaw was so g... x

arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropp... Search

ars technica

MAIN MENU MY STORIES: 25 FORUMS SUBSCRIBE JOBS

RISK ASSESSMENT / SECURITY & HACKTIVISM

**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 Curve25519?



2016: Counterfeit “primes” .

The screenshot shows the top portion of an Ars Technica article. The browser address bar displays the URL: [arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor/](http://arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor/). The Ars Technica logo is visible in the top left. The navigation bar includes links for 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. The article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor', with a subtitle: 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. Social sharing buttons for Facebook, Twitter, and Email are present at the bottom of the article header.



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” .

The screenshot shows the top portion of an Ars Technica article. The browser address bar displays the URL: `arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor`. The Ars Technica logo is visible in the top left. The navigation bar includes links for 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. The article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor', with a subtitle: 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. Social sharing buttons for Facebook, Twitter, and Email are present.



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” .

The screenshot shows the top portion of an Ars Technica article. The browser address bar displays the URL: [arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor/](http://arstechnica.com/security/2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor/). The Ars Technica logo is visible in the top left. The navigation bar includes links for 'MAIN MENU', 'MY STORIES: 25', 'FORUMS', 'SUBSCRIBE', and 'JOBS'. Below the navigation bar, the article title is 'Crypto flaw was so glaring it may be intentional eavesdropping backdoor' with a subtitle 'Network tool contained hard-coded prime number that wasn't prime after all.' The author is 'Dan Goodin' and the date is 'Feb 2, 2016 1:16pm CST'. Social sharing buttons for Facebook, Twitter, and Email are present at the bottom of the article header.



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.

counterfeit “primes”.

2016/02/crypto-flaw-was-so-glaring-it-may-be-intentional-eavesdropping-backdoor

MY STORIES: 25 FORUMS SUBSCRIBE JOBS

ASSESSMENT / SECURITY & HACKTIVISM

Crypto-flaw was so glaring it may be intentional eavesdropping backdoor  
Hard-coded prime number that wasn't prime after all.

1:16pm CST

Share Tweet Email



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” .

glaring-it-may-be-intentional-eavesdropping-backdoor

FORUMS SUBSCRIBE JOBS

SECURITY & HACKTIVISM

glaring it may be  
eavesdropping backdoor  
number that wasn't prime after all.

Share Tweet Email



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

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

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

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

Euler+Gauss definition  
for one curve:  $c^4$

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:  $c^4 = -1$ .

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.  
 reviewer #4 convinced  
 people as part of discussion.  
 ram chairs liked paper.  
 someone thought the title  
 International Conference on  
**Standards and Practice** in Public-  
 Cryptography” 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  
 $x^2 + y^2$

$$d = c^4 i$$

$$d = 0 \text{ is}$$

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–Lauterbach  
 addition and doubling  
 curves”: Edwards  
 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  
 $x^2 + y^2 = 1 + d x^2 y^2$   
 $d = c^4$  is original  
 $d = 0$  is circle, no

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 “Fast 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.

## 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

form 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

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

Gauss defined this law

$$\text{curve: } c^4 = -1.$$

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 non

as a com

In partic

elliptic curves”:

$$\left( \frac{x_2 y_1 + x_1 y_2}{1 - d x_1 x_2 y_1 y_2}, \frac{x_1 x_2 - y_1 y_2}{1 + d x_1 x_2 y_1 y_2} \right)$$

addition law

$$= (x_3, y_3)$$

curve of the form

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

used 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  $x^2 + y^2 = 1 + dx^2 y^2$ .

$d = c^4$  is original Edwards.

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

Surprise for non-square  $d$ :

this addition law is **complete!**

By easy change of variables we can write  $y^2 = x^3 + Ax + B$  with non-square  $A$  and  $B$  as a complete Edwards curve. In particular: Curve

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 coordinates can write  $y^2 = x^3 + Ax^2 + Bx + C$  with non-square  $A^2 - 4B$  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  $x^2 + y^2 = 1 + d x^2 y^2$ .

$d = c^4$  is original Edwards.

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

Surprise for non-square  $d$ :

this addition law is **complete!**

By easy change of 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  $x^2 + y^2 = 1 + d x^2 y^2$ .

$d = c^4$  is original Edwards.

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

Surprise for non-square  $d$ :

this addition law is **complete!**

By easy change of 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

generalizes to

$$= \frac{x_1 y_2 + x_2 y_1}{1 + dx_1 x_2 y_1 y_2},$$

$$= \frac{y_1 y_2 - x_1 x_2}{1 - dx_1 x_2 y_1 y_2}.$$

elliptic curve of the form

$$= 1 + dx^2 y^2.$$

s original Edwards.

circle, non-elliptic.

for non-square  $d$ :

dition 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 20

Lange–S

Start fro

Skip sign

Support

Use dou

include p

$SB = R$

Generate

as a secr

$\Rightarrow$  Avoid

Use Curv

“–1-tw

change “Faster  
 adding on elliptic

addition law

to

$$\frac{-x_2y_1}{x_2y_1y_2},$$

$$\frac{-x_1x_2}{x_2y_1y_2}.$$

ve of the form  
 $y^2$ .

Edwards.

n-elliptic.

square  $d$ :

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 Berns

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  $R$  determ

as a secret hash o

$\Rightarrow$  Avoid PlayStat

Use Curve25519 in

“-1-twisted” Edw

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 signature  
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.

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

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

discriminant-square  $A^2 - 4$

complete Edwards curve.

example: Curve25519.

arithmetic is very fast.

various followup papers:

(later!)

as fast as Montgomery

$\mapsto nP$  in DH.

need records for

$$Q \mapsto mP + nQ$$

for 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.

## Optimization

2007 Ga

2009 Co

2011 Be

Schwabe

2012 Be

2014 La

2014 Ma

2014 Sa

2015 Ch

2015 Dü

Hutter–

microco

2015 Hu

Wieser:

coordinates

$$+ Ax^2 + x$$

$$^2 - 4$$

wards 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  $M$ .

⇒ Avoid PlayStation disaster.

Use Curve25519 in complete

“–1-twisted” Edwards form.

## Optimizations for

2007 Gaudry–Tho

2009 Costigan–Sch

2011 Bernstein–D

Schwabe–Yang: N

2012 Bernstein–Sc

2014 Langley–Mod

2014 Mahé–Chauv

2014 Sasdrich–Gü

2015 Chou: newer

2015 Düll–Haase–

Hutter–Paar–Sánc

microcontrollers.

2015 Hutter–Schill

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  $M$ .

⇒ 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–Hinterwäld

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  $M$ .

⇒ 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.

ve25519 signature system

2011 Bernstein–Duif–

Schwabe–Yang:

from Schnorr signatures.

signature compression.

batch verification.

variable-size  $H$  output, and

public key  $A$  as input:

$$r + H(R, A, M)A.$$

where  $R$  deterministically

is the secret hash of  $M$ .

the PlayStation disaster.

ve25519 in complete

“compressed” 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-generation

NaCl: NaCl

Cryptographic

very simple

key authentication

All-in-one

uses Curve25519

Salsa20

Poly1305

More on

2011 Bernstein

“The second

new cryptographic

Signature system

Stein–Duif–

Yang:

for signatures.

compression.

ification.

output, and

$A$  as input:

$(A, M)A$ .

ministically

of  $M$ .

ion disaster.

n complete

wards 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 auth

More on NaCl des

2011 Bernstein–La

“The security imp

new cryptographic

## 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 provides a 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”.

## 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” .

Implementations for more platforms

[Audry–Thomé](#): Core 2.

[Bernstein–Schwabe](#): Cell.

[Bernstein–Duif–Lange–](#)

[Yang](#): Nehalem.

[Bernstein–Schwabe](#): NEON.

[Langley–Moon](#): newer Intel.

[Fahé–Chauvet](#): GPUs.

[Sdrich–Güneysu](#): FPGAs.

[Liu](#): newer Intel.

[Bill–Haase–Hinterwälder–](#)

[Paar–Sánchez–Schwabe](#):

controllers.

[Winter–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”](#).

Simplicity

Curve25519  
advertised

2013 Bernstein–Lange–Schwabe  
reimplemented  
tweets.

more platforms

omé: Core 2.

hwabe: Cell.

uif–Lange–

lehalem.

chwabe: NEON.

on: newer Intel.

vet: GPUs.

neysu: FPGAs.

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



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?

## 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.

generation crypto library

networking and

graphy library provides

simple new API for public-  
authenticated encryption.

the `crypto_box` function

Curve25519 for DH,

for encryption,

5 for authentication.

NaCl design: see

Bernstein–Lange–Schwabe

curity impact of a

otographic 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-c

*and* is th

towards

crypto library

and

ary provides

API for public-

encryption.

\_box function

for 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 software](#)

verification of **cor**

two high-speed as

Newer work ongoing

Russinoff “A comp

[surveyable proof o](#)

[Curve25519 group](#)

Bernstein–Schwab

Single-curve code

*and* is the most pr

towards bug-free E

## 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–  
Smetzers](#): 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](#)”;  
Bernstein–Schwabe [gfverify](#)

Single-curve code helps speed  
*and* is the most promising approach  
towards bug-free ECC software

## 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](#),

menting 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](#)

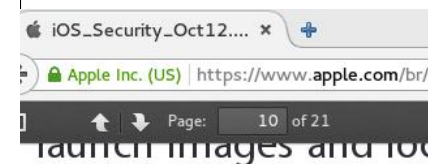
[Curve25519 group axioms](#)”; 2015

Bernstein–Schwabe [gfverif](#).

Single-curve code helps speed

*and* is the most promising avenue  
towards bug-free ECC software.

2012: A



Protected Unless Op

(NSFileProtectionCon

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

Protected Until First

(NSFileProtectionCon

the same way as Com

removed from memor

similar properties to c

that involve a reboot.

No Protection

(NSFileProtectionNor

in Effaceable Storage.

Data Protection class.

on the device, the enc

not assigned a Data P

code.”

nssen–

[TweetNaCl](#),

NaCl in 100

and matter?

code: The hash

de signatures!

[in 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 [depl](#)



#### Protected Unless Open

**(NSFileProtectionCompleteUnlessOpen):** So the device is locked. A good example of this background. This behavior is achieved by using (ECDH over Curve25519). Along with the usual a file public/private key pair. A shared secret and the Protected Unless Open class public key protected with the user’s passcode and the with the hash of this shared secret and stored file’s public key; the corresponding private key as the file is closed, the per-file key is also with the shared secret is re-created using the Protected Unless Open class public key; its hash is used then used to decrypt the file.

#### Protected Until First User Authentication

**(NSFileProtectionCompleteUntilFirstUserAuthentication):** the same way as Complete Protection, except removed from memory when the device is locked. Similar properties to desktop full-disk encryption that involve a reboot.

#### No Protection

**(NSFileProtectionNone):** This class key is provided in Effaceable Storage. This is the default class for Data Protection class. Since all the keys need to be on the device, the encryption only affords the protection. not assigned a Data Protection class, it is still

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



### Protected Unless Open

**(NSFileProtectionCompleteUnlessOpen):** Some files may need to be accessible even when the device is locked. A good example of this is a mail attachment downloaded 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 also uses a file public/private key pair. A shared secret is computed using the per-file public 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 encrypted with the hash of this shared secret and stored in the file’s metadata. When the file is opened, the file’s public key; the corresponding private key is then wiped from memory. As the file is closed, the per-file key is also wiped from memory. To open the file, the shared secret is re-created using the Protected Unless Open class public 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 of protection works the same way as Complete Protection, except that the decrypted content is removed from memory when the device is locked. The protection has similar properties to desktop full-disk encryption, and protects data that survive a reboot.

### No Protection

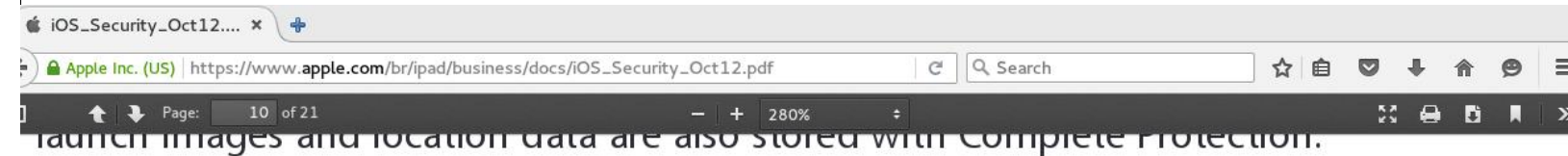
**(NSFileProtectionNone):** This class key is protected only with the user’s passcode in Effaceable Storage. This is the default class for all files not otherwise protected by a Data Protection class. Since all the keys needed to decrypt files in this class are on the device, the encryption only affords the benefit of fast remote wipe. Files in this class, not assigned a Data Protection class, it is still stored in encrypted form.

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



### 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  
n-speed asm main loops.  
work ongoing: e.g., 2015  
f “A computationally  
le proof of the  
519 group axioms”; 2015  
n-Schwabe [gfverif](#).  
urve code helps speed  
ne most promising avenue  
bug-free ECC software.

38

## 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

39

## 2013: S

**GitHub**

WhisperSystem

Code

**Migrate to Curve25519**

- 1) Generate a
- 2) Use Curve25519
- 3) Initiate v2
- 4) Accept v1 k
- 5) TOFU Curve2

moxie0 comm

Showing 57 chan

in-Schwabe-  
-Yang “Verifying  
are”: formal  
correctness of  
m main loops.  
ng: e.g., 2015  
putationally  
f the  
axioms”; 2015  
e `gfverif`.  
helps speed  
promising avenue  
ECC software.

38

## 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

39

## 2013: Signal deplo

**GitHub** This repository Search

WhisperSystems / Signal-Android

Code Issues 613 Pull Requests

**Migrate to Curve25519.**

- 1) Generate a Curve25519 identity.
- 2) Use Curve25519 ephemerals and
- 3) Initiate v2 key exchange mes
- 4) Accept v1 key exchange messa
- 5) TOFU Curve25519 identities.

**moxie0** committed on Nov 10, 2013

Showing 57 changed files with 2,194 additions and 2,194 deletions

## 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

Migrate to Curve25519

GitHub

This repository Search

WhisperSystems / Signal-Android

Code

Issues 613

Pull requests 28

### Migrate to Curve25519.

- 1) Generate a Curve25519 identity key.
- 2) Use Curve25519 ephemerals and identities for
- 3) Initiate v2 key exchange messages.
- 4) Accept v1 key exchange messages.
- 5) T0FU Curve25519 identities.

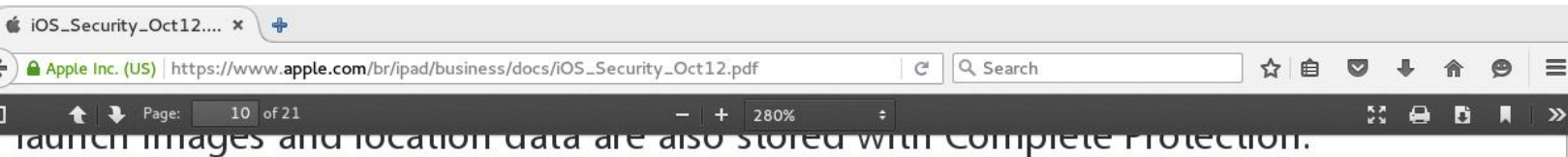


moxie0 committed on Nov 10, 2013



Showing 57 changed files with 2,194 additions and 495 deletions

## 2012: Apple deploys Curve25519



### 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


**GitHub** This repository Search

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 agreement.
- 3) Initiate v2 key exchange messages.
- 4) Accept v1 key exchange messages.
- 5) T0FU Curve25519 identities.

 **moxie0** committed on Nov 10, 2013

Showing **57 changed files** with **2,194 additions** and **495 deletions**.



# Apple deploys Curve25519

39

ation data are also stored with Complete Protection.

en

**CompleteUnlessOpen):** Some files may need to be written while a good example of this is a mail attachment downloading in the behavior is achieved by using asymmetric elliptic curve cryptography (9). Along with the usual per-file key, Data Protection generates a key pair. A shared secret is computed using the file's private key and the device's Open class public key, whose corresponding private key is derived from the user's passcode and the device UID. The per-file key is wrapped with the shared secret and stored in the file's metadata along with the file's corresponding private key is then wiped from memory. As soon as the file is opened, the per-file key is also wiped from memory. To open the file again, the file is decrypted using the Protected Unless Open class's private key and the device's public key; its hash is used to unwrap the per-file key, which is then used to decrypt the file.

**User Authentication**

**CompleteUntilFirstUserAuthentication):** This class behaves in a similar way to Complete Protection, except that the decrypted class key is not wiped from memory when the device is locked. The protection in this class has been used for desktop full-disk encryption, and protects data from attacks.

**Protected):** This class key is protected only with the UID, and is kept in memory. This is the default class for all files not otherwise assigned to a specific protection class. Since all the keys needed to decrypt files in this class are stored in memory, encryption only affords the benefit of fast remote wipe. If a file is encrypted using the Protected class, it is still stored in encrypted form (as is all data

# 2013: Signal deploys Curve25519

40

The screenshot shows a GitHub commit page for the repository `WhisperSystems / Signal-Android`. The commit message is `Migrate to Curve25519.` The commit was made by `moxie0` on Nov 10, 2013. The commit details show 57 changed files with 2,194 additions and 495 deletions. The commit message includes a list of changes:

- 1) Generate a Curve25519 identity key.
- 2) Use Curve25519 ephemerals and identities for v2 3DHE agreement.
- 3) Initiate v2 key exchange messages.
- 4) Accept v1 key exchange messages.
- 5) TOFU Curve25519 identities.

# 2014: OpenSSH

Changes since OpenSSH 6.5

This is a feature release.

New features:

- \* `ssh(1), sshd(8)`: Diffie-Hellman key exchange method is the default.
- \* `ssh(1), sshd(8)`: Ed25519 is a new elliptic curve key exchange method used for both client and server.
- \* Add a new private key protection method to protect keys at rest. Ed25519 keys, but also existing keys of other algorithms. We intend to make this the default. Details of the new method are in the man pages.
- \* `ssh(1), sshd(8)`: "chacha20-poly1305" ChaCha20 stream cipher and Poly1305 authentication encryption mode.
- \* `ssh(1), sshd(8)`: Servers that use DSA keys will be able to use a weaker key exchange method.
- \* `ssh(1), sshd(8)`: use a weaker key exchange method.

...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

**Authentication):** This class behaves in  
 ...ot that the decrypted class key is not  
 ...locked. The protection in this class has  
 ...option, and protects data from attacks

...ected 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

Migrate to Curve25519... x

GitHub, Inc. (US) | https://github.com/WhisperSystems/Signal-Android/commit/c3c6fd2d4fc62c8a369


GitHub This repository Search

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 agreement.
- 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.

Changes since OpenSSH 6.4  
 =====

This is a feature-focused release.

New features:

- \* ssh(1), sshd(8): Add support for KexDiffieHellman in Daniel Bernstein method is the default when both th
- \* ssh(1), sshd(8): Add support for Ed25519 is a elliptic curve signature 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 transport "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

Migrate to Curve25519


GitHub This repository Search

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 agreement.
- 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**.

Changes since OpenSSH 6.4  
=====

This is a feature-focused release.

New features:

- \* ssh(1), sshd(8): Add support for key exchange using Daniel Bernstein's Curve25519. This method is the default when both the client and server support it.
- \* ssh(1), sshd(8): Add support for Ed25519 as a public key signature scheme. Ed25519 is an elliptic curve signature scheme that offers better security than ECDSA and DSA and good performance. It is used for both user and host keys.
- \* Add a new private key format that uses a bcrypt KDF to protect keys at rest. This format is used unconditionally for Ed25519 keys, but may be requested when generating other existing keys of other types via the `-o ssh-keygen(1)` option. We intend to make the new format the default in the next release. 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.chacha20` file.
- \* ssh(1), sshd(8): Refuse RSA keys from old proprietary servers that use the obsolete RSA+MD5 signature scheme. It will still be possible to connect with these clients/servers, but DSA keys will be accepted, and OpenSSH will refuse connections entirely in a future release.
- \* ssh(1), sshd(8): Refuse old proprietary clients and servers that use a weaker key exchange hash calculation.

# 2013: Signal deploys Curve25519

Migrate to Curve25519... x

GitHub, Inc. (US) | <https://github.com/WhisperSystems/Signal-Android/commit/c3c6fd2d4fc62c8a369> Search


**GitHub** This repository Search

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 agreement.
- 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 deploys Curve25519

http://www...lease-6.5 x

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.

# Signal deploys Curve25519

40

Systems/Signal-Android/commit/c3c6fd2d4fc62c8a365

Search

Issues / **Signal-Android**

Issues **613** Pull requests **28** Wiki Pulse

**Curve25519.**

Curve25519 identity key.

25519 ephemerals and identities for v2 3DHE agreement

key exchange messages.

key exchange messages.

25519 identities.

mitted on Nov 10, 2013

ged files with 2,194 additions and 495 deletions.

# 2014: OpenSSH deploys Curve25519

41

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: EdDSA- for signa X25519

2015.10: ECC sta paving w

2015.11: X25519

These ar

Many m

/curve2

and /ed

62c8a369 Search

Explo

android

requests 28 Wiki Pulse

ity key.

nd identities for v2 3DHE agreeme

ssages.

ages.

3

ditions and 495 deletions.

http://www...lease-6.5 x

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 CF  
EdDSA—Ed25519  
for signatures. Alr  
X25519 and X448

2015.10: NIST rec  
ECC standards for  
paving way for new

2015.11: BoringSS  
X25519 and Ed25519

These are just som  
Many more: [iani](#)  
[/curve25519-dep](#)  
and [/ed25519-de](#)

## 2014: OpenSSH deploys Curve25519

http://www...lease-6.5 x

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 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/publications/curve25519-deployment](http://ianix.com/publications/curve25519-deployment) and [/ed25519-deployment](http://ianix.com/publications/ed25519-deployment)

## 2014: OpenSSH deploys Curve25519

```

http://www...lease-6.5 x
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](http://ianix.com/pub/curve25519-deployment.html) and [/ed25519-deployment.html](http://ianix.com/pub/ed25519-deployment.html).