On the Security of Edwards Curves

Watson Ladd

April 28, 2014

The Problem

- Short Weierstrass curves are secure

The Problem

- Short Weierstrass curves are secure
- Slow

The Problem

- Short Weierstrass curves are secure
- Slow
- Implementations difficult

The Problem

- Short Weierstrass curves are secure
- Slow
- Implementations difficult
- Standardized curves make these worse

The Problem

- Short Weierstrass curves are secure
- Slow
- Implementations difficult
- Standardized curves make these worse
- Security-performance tradeoff

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step
- Best addition 14M, doubling 9M

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step
- Best addition 14M, doubling 9M
- RFC 6090 got the addition law wrong

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step
- Best addition 14M, doubling 9M
- RFC 6090 got the addition law wrong
- Probably not only case

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step
- Best addition 14M, doubling 9M
- RFC 6090 got the addition law wrong
- Probably not only case
- Identity is not in the plane: cannot be represented

Problem Redux

- $y^2 = x^3 + ax + b$ has addition law
- Addition law cannot be extended to all points
- Can define ladder: costs 19M per step
- Best addition 14M, doubling 9M
- RFC 6090 got the addition law wrong
- Probably not only case
- Identity is not in the plane: cannot be represented
- Branches driven by secret data: deep analysis required

Solution

- Use curve shapes with complete addition law

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero
- Identity is $(0, 1)$.

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero
- Identity is $(0, 1)$.
- 9M addition, 7M doubling

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero
- Identity is $(0, 1)$.
- 9M addition, 7M doubling
- Addition has no exceptional cases

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero
- Identity is $(0, 1)$.
- 9M addition, 7M doubling
- Addition has no exceptional cases
- Code can get correct answer every time without data-dependent branches

Solution

- Use curve shapes with complete addition law
- Twisted Edwards: $ax^2 + y^2 = 1 + dx^2y^2$
- Introduced in 2007, generalized 2010
- When d is not a square no division by zero
- Identity is $(0, 1)$.
- 9M addition, 7M doubling
- Addition has no exceptional cases
- Code can get correct answer every time without data-dependent branches
- Easier to analyze for correctness

Security

- Every curve is a twisted Edwards curve over a small extension

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve
- Cofactor not 1: protocols need slight adjustment in some cases

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve
- Cofactor not 1: protocols need slight adjustment in some cases
- To avoid wrong-curve attacks compress points: twist secure

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve
- Cofactor not 1: protocols need slight adjustment in some cases
- To avoid wrong-curve attacks compress points: twist secure
- Can also use Montgomery form for all Edwards curves: fast ladder

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve
- Cofactor not 1: protocols need slight adjustment in some cases
- To avoid wrong-curve attacks compress points: twist secure
- Can also use Montgomery form for all Edwards curves: fast ladder
- Montgomery form is $y^2 = x^3 + Ax^2 + x$
- Twist security removes checks from Montgomery form

Security

- Every curve is a twisted Edwards curve over a small extension
- Solving DLP on Edwards curve equivalent on short Weierstrass curve
- Cofactor not 1: protocols need slight adjustment in some cases
- To avoid wrong-curve attacks compress points: twist secure
- Can also use Montgomery form for all Edwards curves: fast ladder
- Montgomery form is $y^2 = x^3 + Ax^2 + x$
- Twist security removes checks from Montgomery form

Life with cofactors

- Solution 1: Write Diffie-Hellman as ahG, bhG deriving $abhhG$

Life with cofactors

- Solution 1: Write Diffie-Hellman as ahG, bhG deriving $abhhG$
- Solution 2: Order checks

Life with cofactors

- Solution 1: Write Diffie-Hellman as ahG, bhG deriving $abhhG$
- Solution 2: Order checks
- Solution 1 is faster

Life with cofactors

- Solution 1: Write Diffie-Hellman as ahG, bhG deriving $abhhG$
- Solution 2: Order checks
- Solution 1 is faster
- Because identity is representable, no issues

Life with cofactors

- Solution 1: Write Diffie-Hellman as ahG, bhG deriving $abhhG$
- Solution 2: Order checks
- Solution 1 is faster
- Because identity is representable, no issues
- Naive implementations work again

Picking the right curves

- Same security considerations as Weierstrass curves

Picking the right curves

- Same security considerations as Weierstrass curves
- Need to permit a cofactor of 4 over base field: standards allow it

Picking the right curves

- Same security considerations as Weierstrass curves
- Need to permit a cofactor of 4 over base field: standards allow it
- High embedding degree, prime order twist, not supersingular

Picking the right curves

- Same security considerations as Weierstrass curves
- Need to permit a cofactor of 4 over base field: standards allow it
- High embedding degree, prime order twist, not supersingular
- Best known attack is Pollard rho method or variations

Picking the right curves

- Same security considerations as Weierstrass curves
- Need to permit a cofactor of 4 over base field: standards allow it
- High embedding degree, prime order twist, not supersingular
- Best known attack is Pollard rho method or variations
- Generally use small constants to speed up calculations

Picking the right curves

- Same security considerations as Weierstrass curves
- Need to permit a cofactor of 4 over base field: standards allow it
- High embedding degree, prime order twist, not supersingular
- Best known attack is Pollard rho method or variations
- Generally use small constants to speed up calculations
- Security picture very well understood on $\mathbb{F}_p$

Which curves?

- Pick security levels: fastest curve at each security level

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level
- Use Montgomery form alongside: already proposed and implemented for TLS

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level
- Use Montgomery form alongside: already proposed and implemented for TLS
- All other curves only twisted Edwards form

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level
- Use Montgomery form alongside: already proposed and implemented for TLS
- All other curves only twisted Edwards form
- Primes not congruent to 1 modulo 8

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level
- Use Montgomery form alongside: already proposed and implemented for TLS
- All other curves only twisted Edwards form
- Primes not congruent to 1 modulo 8
- Point Compression: patent expires in July

Which curves?

- Pick security levels: fastest curve at each security level
- Curve25519 clear winner at 128 bit level
- Use Montgomery form alongside: already proposed and implemented for TLS
- All other curves only twisted Edwards form
- Primes not congruent to 1 modulo 8
- Point Compression: patent expires in July
- Send only y coordinate, use in ECDH: Montgomery ladder interop

How big?

- Hash your points!

How big?

- Hash your points!
- All protocols do this

How big?

- Hash your points!
- All protocols do this
- Reduction to DDH (not SDH)

How big?

- Hash your points!
- All protocols do this
- Reduction to DDH (not SDH)
- Cheon paper completely irrelevant

How big?

- Hash your points!
- All protocols do this
- Reduction to DDH (not SDH)
- Cheon paper completely irrelevant
- If you are happy with P256, Curve25519 will be same security

How big?

- Hash your points!
- All protocols do this
- Reduction to DDH (not SDH)
- Cheon paper completely irrelevant
- If you are happy with P256, Curve25519 will be same security
- But much faster: makes deployment easier

How big?

- Hash your points!
- All protocols do this
- Reduction to DDH (not SDH)
- Cheon paper completely irrelevant
- If you are happy with P256, Curve25519 will be same security
- But much faster: makes deployment easier
- Same story at larger security levels