Elliptic Curve Cryptography in Practice (Bos et al.)

a.k.a. ECC: What? How? Why? Where?

Brendan Cordy

Cryptography Fundamentals

- Authentication: Who is on the other end?

Cryptography Fundamentals

- Authentication: Who is on the other end?
- Key Generation: I'll tell you a secret... in public... but it's still our secret.

Cryptography Fundamentals

- Authentication: Who is on the other end?
- Key Generation: I'll tell you a secret... in public... but it's still our secret.
- Encryption: ebgguvegrravfgurorfg

Elliptic Curves are Great!

- Elliptic Curve methods are the current state of the art for the first two problems. (7.2% ECDH support figure not so meaningful)

Elliptic Curves are Great!

- Elliptic Curve methods are the current state of the art for the first two problems. (7.2% ECDH support figure not so meaningful)
- DSA Keys: 2048 bits, ECDSA Keys: 256 bits.

Elliptic Curves are Great!

- Elliptic Curve methods are the current state of the art for the first two problems. (7.2% ECDH support figure not so meaningful)
- DSA Keys: 2048 bits, ECDSA Keys: 256 bits.
- If math stops now, ECDSA keys will stay under 1024 bits for as long as computers are made out of classical logic gates.

Elliptic Curves are Bad!

- ECDSA verification takes more time than RSA, but not enough to be impractical.

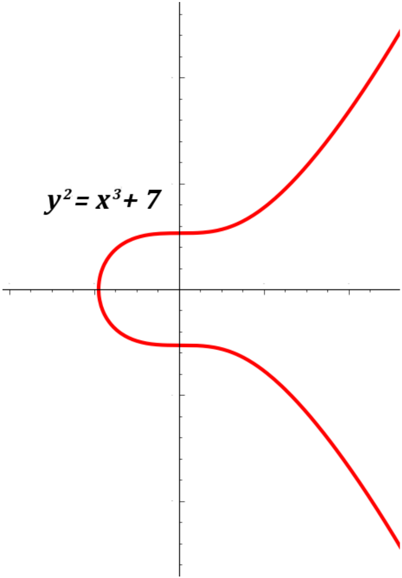
Elliptic Curves are Bad!

- ECDSA verification takes more time than RSA, but not enough to be impractical.
- ‘Certicom holds U.S. Patent 6,782,100 on calculating the x-coordinate of the double of a point in binary curves via a Montgomery ladder in projective coordinates’.

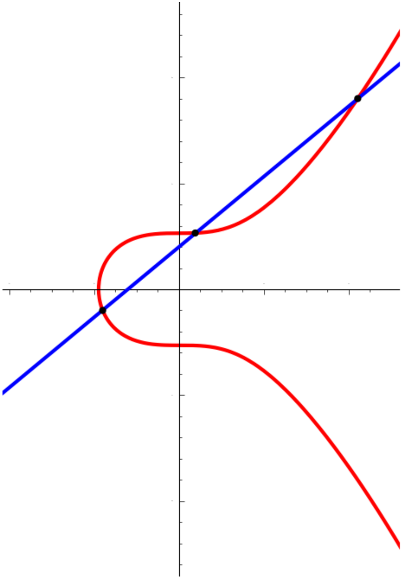
Elliptic Curves are Bad!

- ECDSA verification takes more time than RSA, but not enough to be impractical.
- ‘Certicom holds U.S. Patent 6,782,100 on calculating the x-coordinate of the double of a point in binary curves via a Montgomery ladder in projective coordinates’.
- Dual_EC_DRGB scared everybody.

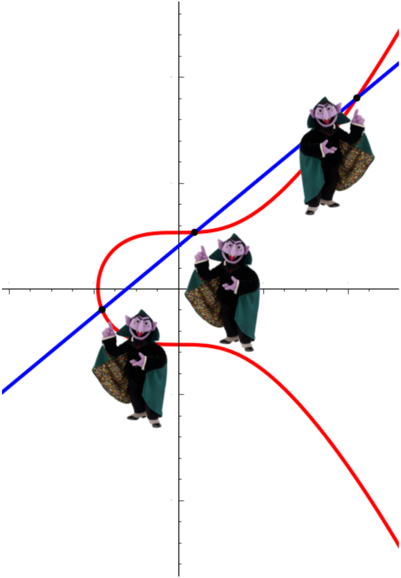
Elliptic Curves



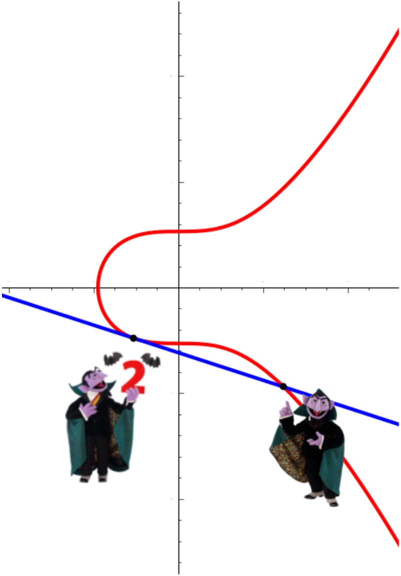
Elliptic Curves



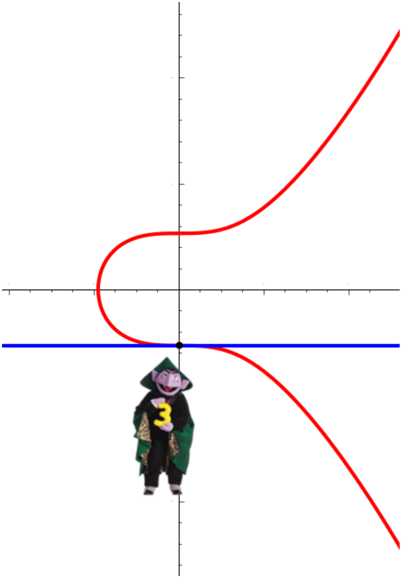
Elliptic Curves



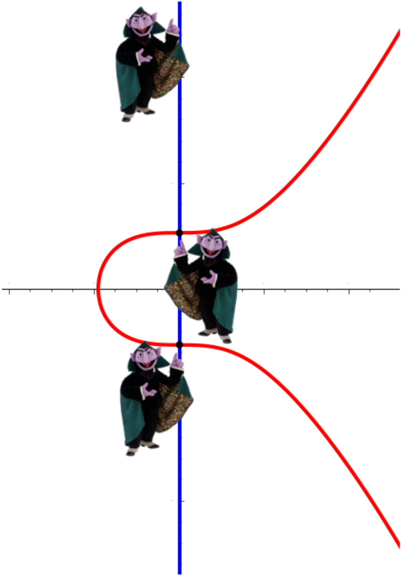
Elliptic Curves



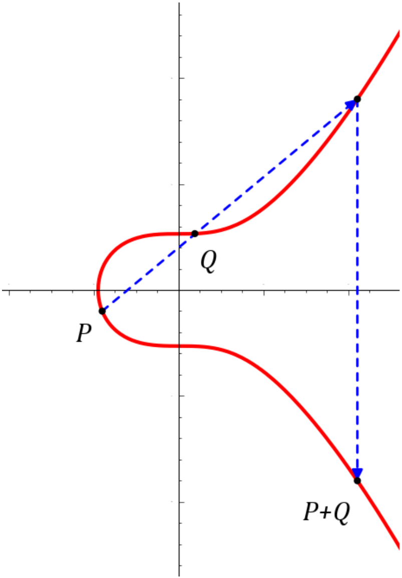
Elliptic Curves



Elliptic Curves



Adding Points



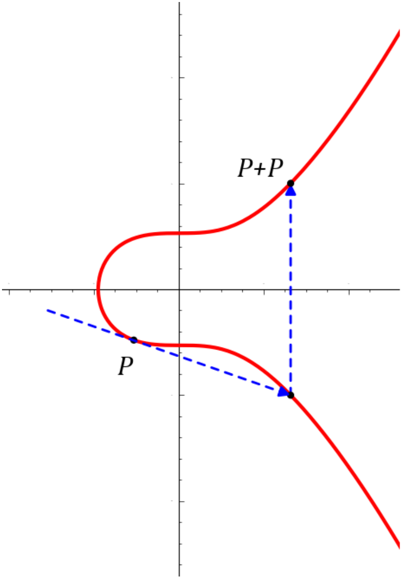
Adding Points

- Strange, but has all the nice properties that addition should. Explicitly computing the sum takes a handful of operations.

$$\text{Let } \lambda = \frac{y_Q - y_P}{x_Q - x_P}, \text{ then}$$

$$\begin{aligned} x_{P+Q} &= \lambda^2 - x_P - x_Q \\ y_{P+Q} &= \lambda(x_Q - x_{P+Q}) - y_P \end{aligned}$$

Point Doubling



Point Doubling

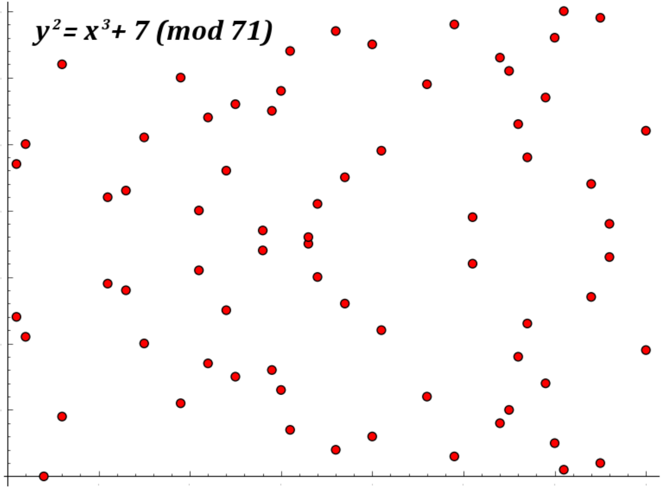
- Again, explicitly computing the coordinates takes a handful of operations.

Point Doubling

- Again, explicitly computing the coordinates takes a handful of operations.

- In fact, we can use the same formulas, but with $\lambda = \frac{3{x_P}^2}{2y_P}$, so doubling takes about the same amount of time as adding distinct points.

Everything Works in $\mathbb{F}_p$



Elliptic Curve Terminology

- In practice, a 'curve' means an explicit equation as well as a finite field and base point P on the curve.

Elliptic Curve Terminology

- In practice, a ‘curve’ means an explicit equation as well as a finite field and base point P on the curve.

- Standards: NIST, Certicom, DJB.

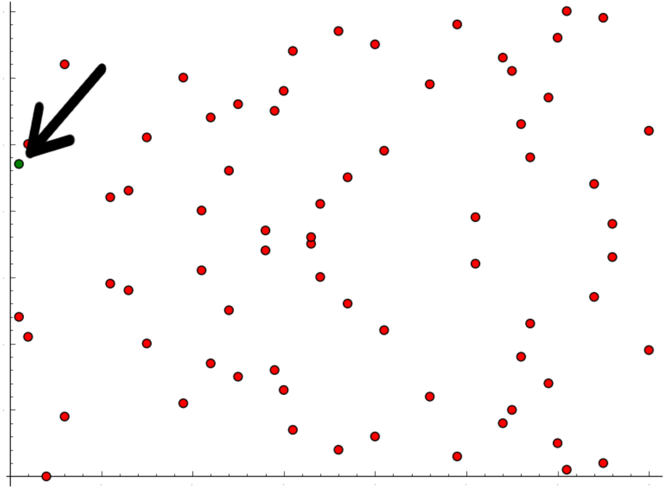
Elliptic Curve Terminology

- In practice, a ‘curve’ means an explicit equation as well as a finite field and base point P on the curve.

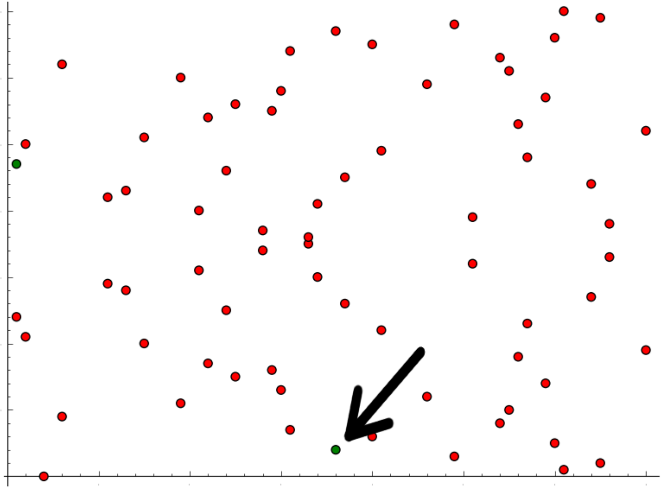
- Standards: NIST, Certicom, DJB.

- In the secp256k1 standard, the field is $\mathbb{F}_p$ with $p = 2^{256} - 4294966319$ and the curve has about that many points on it.

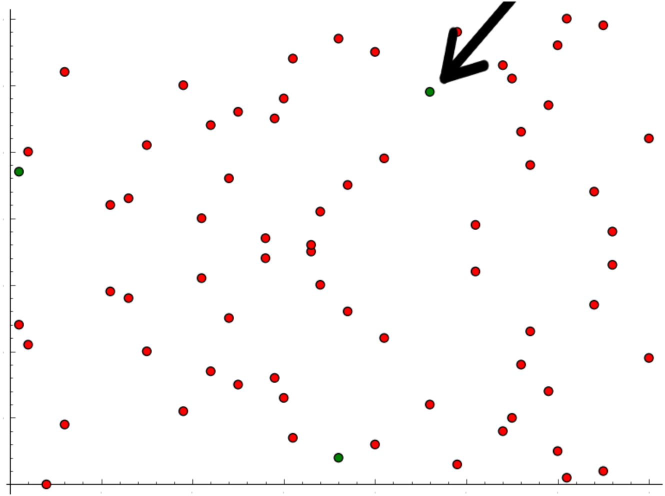
Consider $P(1, 47)$



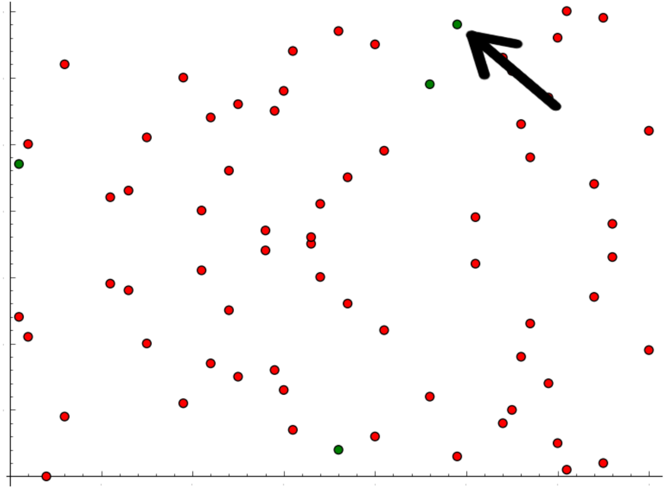
Compute P+P



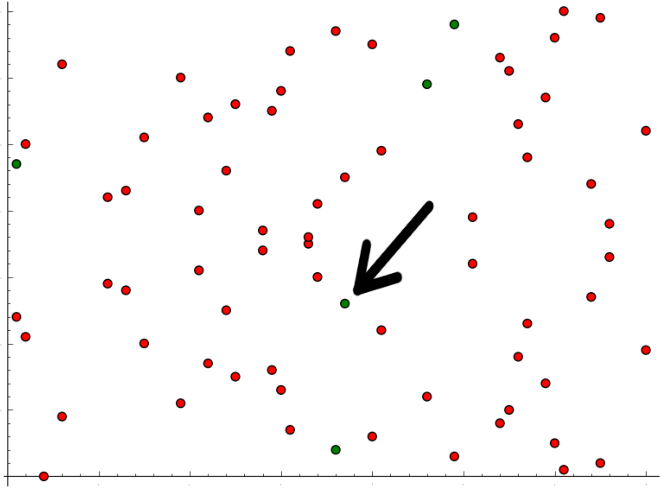
Compute P+P+P



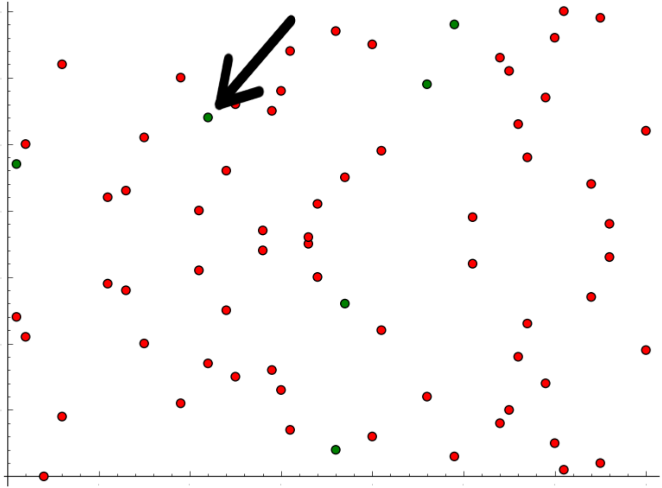
Compute P+P+P+...



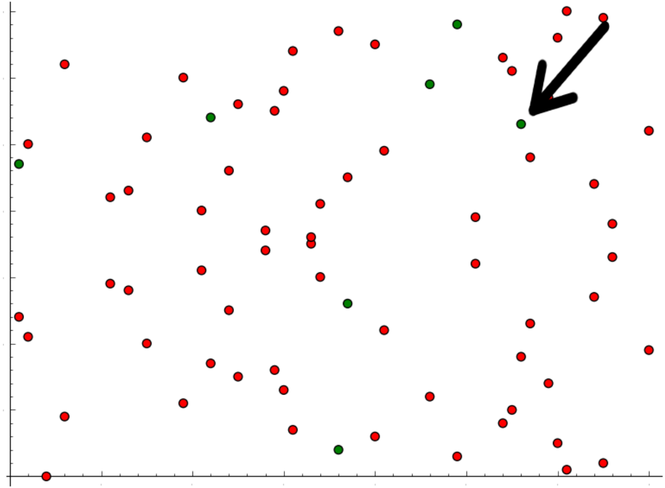
Compute P+P+P+...



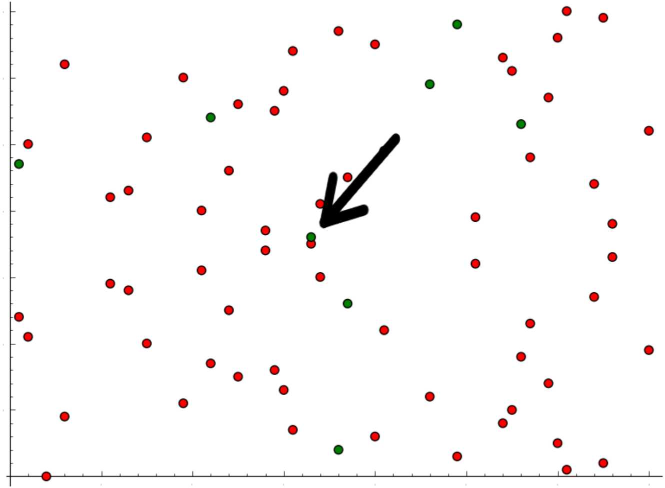
Compute P+P+P+...



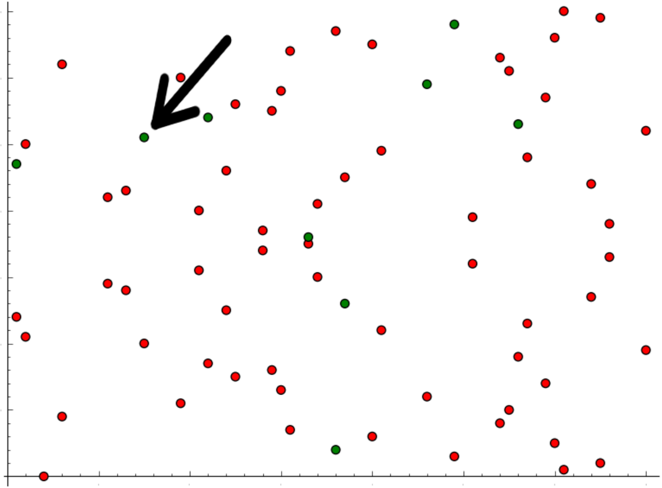
Compute P+P+P+...



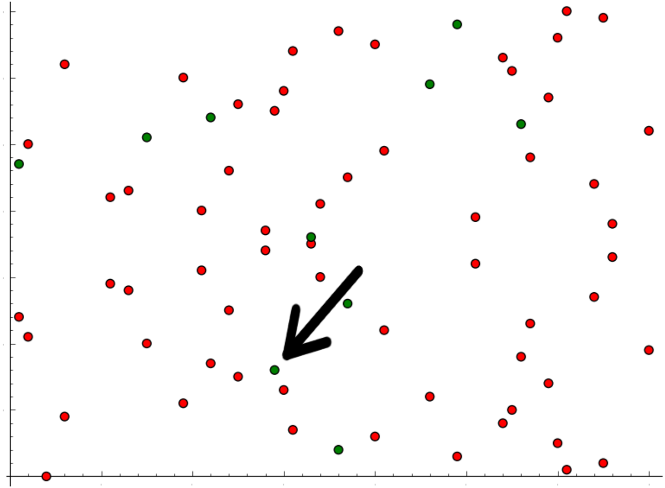
Compute P+P+P+...



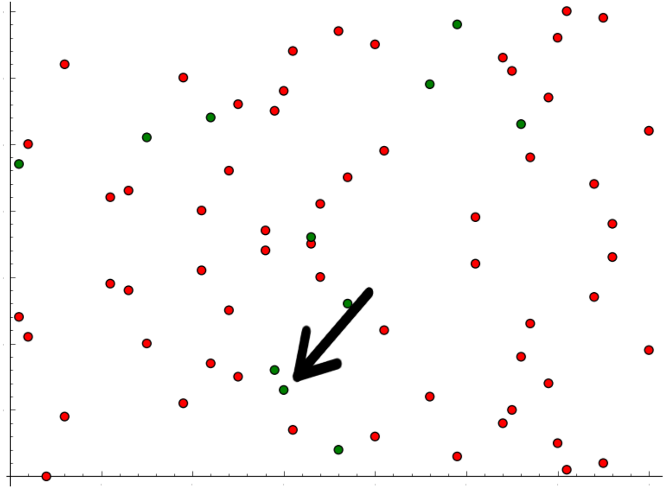
Compute P+P+P+...



Compute P+P+P+...



Compute P+P+P+...



Point Multiplication

- Let $nP = \overbrace{P + P + \ldots + P}^{n \text{ times}}$.

Point Multiplication

- Let $nP = \overbrace{P + P + \ldots + P}^{n \text{ times}}$.
- We want a point P whose multiples cycle through all points on the curve (or at least a non-negligible proportion of them).

Point Multiplication

- Let $nP = \overbrace{P + P + \ldots + P}^{n \text{ times}}$.

- We want a point P whose multiples cycle through all points on the curve (or at least a non-negligible proportion of them).

- Finding nP appears to require n additions. However, there is a clever way to do it.

Peasant Multiplication

$$179P = 128P + 32P + 16P + 2P + P$$

Peasant Multiplication

$$179P = 128P + 32P + 16P + 2P + P$$

- How many doublings? $\lfloor \log_2(n) \rfloor$

Peasant Multiplication

$$179P = 128P + 32P + 16P + 2P + P$$

- How many doublings? $\lfloor \log_2(n) \rfloor$
- How many additions? $\leq \lfloor \log_2(n) \rfloor$

Peasant Multiplication

$$179P = 128P + 32P + 16P + 2P + P$$

- How many doublings? $\lfloor \log_2(n) \rfloor$
- How many additions? $\leq \lfloor \log_2(n) \rfloor$
- $n \to 2\log_2(n)$: Exponential speedup!

Generating Key-Pairs

- Take a random 256-bit integer d, and use peasant multiplication to compute $Q = dP$.

Generating Key-Pairs

- Take a random 256-bit integer d, and use peasant multiplication to compute $Q = dP$.
- What if instead we knew P and Q, but wanted to compute d? ¯_(ツ)_/¯

Generating Key-Pairs

- Take a random 256-bit integer d, and use peasant multiplication to compute $Q = dP$.
- What if instead we knew P and Q, but wanted to compute d? ¯_(ツ)_/¯
- The point Q is the public key, while the number d is kept secret.

Digital Signatures (ECDSA)

- If you know that $Q = dP$, you can solve...

$$\begin{aligned} aP + bQ &= xP \\ aP + bdP &= xP \\ (a + bd)P &= xP \\ x &= a + bd \end{aligned}$$

Digital Signatures (ECDSA)

- If you know that $Q = dP$, you can solve...

$$\begin{aligned} aP + bQ &= xP \\ aP + bdP &= xP \\ (a + bd)P &= xP \\ x &= a + bd \end{aligned}$$

- Anyone can check whether solution works, without knowledge of d.

Digital Signatures (ECDSA)

- Let h be a hash of a message to be signed, k be a nonce, and r be the x-coordinate of kP. Solve $hP + rQ = zkP$ for z.

Digital Signatures (ECDSA)

- Let h be a hash of a message to be signed, k be a nonce, and r be the x-coordinate of kP. Solve $hP + rQ = zkP$ for z.
- The message is sent with the triple (Q, r, z). Verifiers hash the message to obtain h, and check the equation holds. (They don't know k, but they're given r, so they know kP).

Don't Reuse k (No Sony No!)

- Suppose that messages with hashes h and h' are both signed using nonce k.

$$
\begin{aligned}
hP + rQ &= zkP & \qquad hP + rdP &= zkP \\
h'P + rQ &= z'kP & \qquad h'P + rdP &= z'kP
\end{aligned}
$$

Don't Reuse k (No Sony No!)

- Suppose that messages with hashes h and h' are both signed using nonce k.

$$\begin{aligned} hP + rQ &= zkP & \qquad hP + rdP &= zkP \\ h'P + rQ &= z'kP & \qquad h'P + rdP &= z'kP \end{aligned}$$

- Two equations, two unknowns (the x-coord of kP and the x-coord of dP).

Bitcoin

- Bitcoin is a list of anonymous account numbers (addresses) with balances that's maintained by an anonymous peer to peer network.

1v6X...qT74	$\rightarrow$	54335
1Ag6...yz93	$\rightarrow$	916
1ccV...8kLE	$\rightarrow$	0
14rT...u3d5	$\rightarrow$	1665

Authenticating Transactions

- If the accounts are anonymous, why can't any user spend money in any account?

Authenticating Transactions

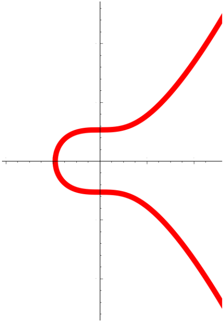
- If the accounts are anonymous, why can't any user spend money in any account?
- From $y^2 = x^3 + 7 \pmod{2^{256} - 4294966319}$ and base point P we can generate a key-pair (d, Q), and then an address from Q.

Authenticating Transactions

- If the accounts are anonymous, why can't any user spend money in any account?
- From $y^2 = x^3 + 7 \pmod{2^{256} - 4294966319}$ and base point P we can generate a key-pair (d, Q), and then an address from Q.
- With ECDSA, we can verify that a message must have originated from the individual who generated the address.

What is a Bitcoin Address?

- `1ELwdsETv4pv1SGvwZ4n2uzXT7bLnR8iVo`



Base 58

0 1 2 3 4 5 6 7 8 9

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

Base 58

0 1 2 3 4 5 6 7 8 9

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

- There are 62 alphanumeric characters, but four of them are easy to confuse.

Base 58

0 1 2 3 4 5 6 7 8 9

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

- There are 62 alphanumeric characters, but four of them are easy to confuse.
- The characters left are ordered like hex.

Base 58

0 1 2 3 4 5 6 7 8 9

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

- There are 62 alphanumeric characters, but four of them are easy to confuse.
- The characters left are ordered like hex.

Base 58

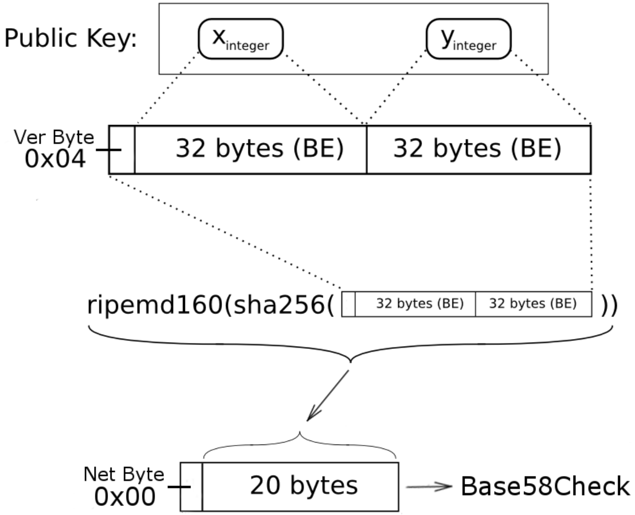
1 2 3 4 5 6 7 8 9

A B C D E F G H J K L M N P Q R S T U V W X Y Z

a b c d e f g h i j k m n o p q r s t u v w x y z

- There are 62 alphanumeric characters, but four of them are easy to confuse.
- The characters left are ordered like hex.

Building Addresses



Cryptographic Hashes

- What are sha256 and ripemd160?

Cryptographic Hashes

- What are sha256 and ripemd160?
- Compression: 64 Bytes to 20 Bytes.

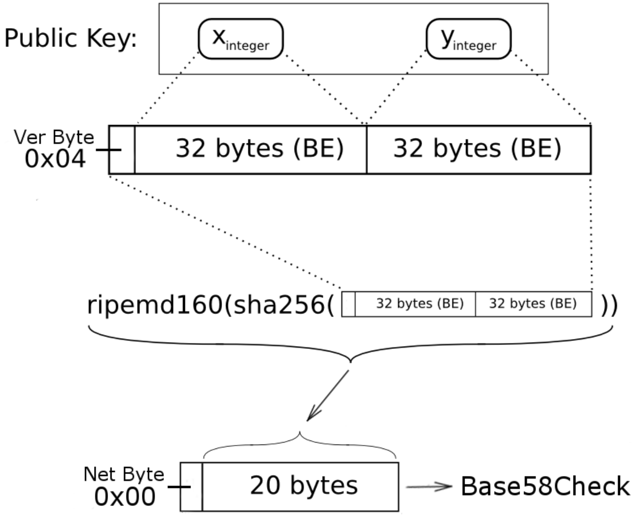
Cryptographic Hashes

- What are sha256 and ripemd160?
- Compression: 64 Bytes to 20 Bytes.
- Cryptographic hash functions are *one-way*.

$$sha256(1729) = ? \quad \leftarrow \quad \text{Super Fast!}$$

$$sha256(?) = 1729 \quad \leftarrow \quad \text{Super Slow!}$$

Building Addresses



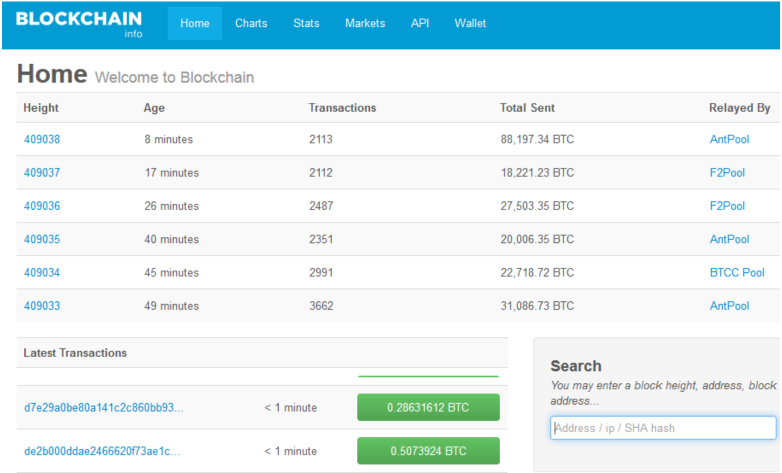
Consequences of Hashing

- What happens if you start with a point Q which is *not* on the curve, and use it to build an address?

Consequences of Hashing

- What happens if you start with a point Q which is *not* on the curve, and use it to build an address?
- You are putting money into limbo. Any amount sent to such an address becomes inaccessible forever!

Counting Money in the Void!



BLOCKCHAIN info

Home | Charts | Stats | Markets | API | Wallet

Home Welcome to Blockchain

Height	Age	Transactions	Total Sent	Relayed By
409038	8 minutes	2113	88,197.34 BTC	AntPool
409037	17 minutes	2112	18,221.23 BTC	F2Pool
409036	26 minutes	2487	27,503.35 BTC	F2Pool
409035	40 minutes	2351	20,006.35 BTC	AntPool
409034	45 minutes	2991	22,718.72 BTC	BTCC Pool
409033	49 minutes	3662	31,086.73 BTC	AntPool

Latest Transactions		
d7e29a0be80a141c2c860bb93...	< 1 minute	0.28631612 BTC
de2b000ddae2466620f73ae1c...	< 1 minute	0.5073924 BTC

Search

You may enter a block height, address, block address...

Address / ip / SHA hash

Silly Addresses I

- Build an address from the empty string!

Address	Balance
`1HT7x...K8d4E`	\$31 500

Silly Addresses I

- Build an address from the empty string!

Address	Balance
`1HT7x...K8d4E`	\$31 500

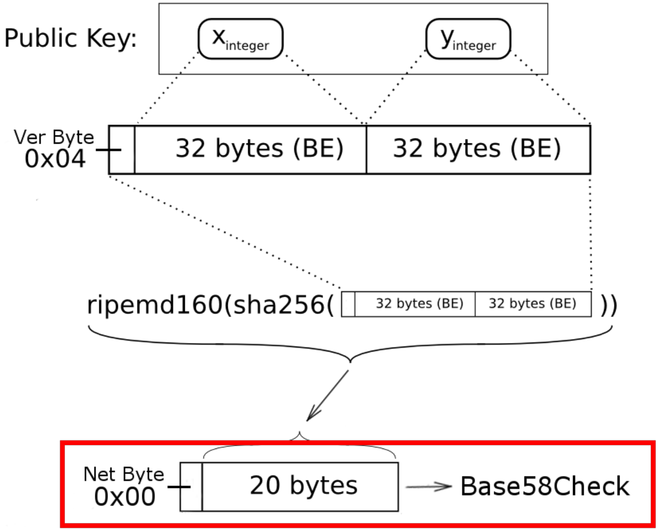
- Build an address from the point (0,0)!

Address	Balance
`1FYMZ...YKQxh`	\$1 650

Silly Addresses II

- Convert simple hex values to Base 58, and add correct checksums!

Building Addresses



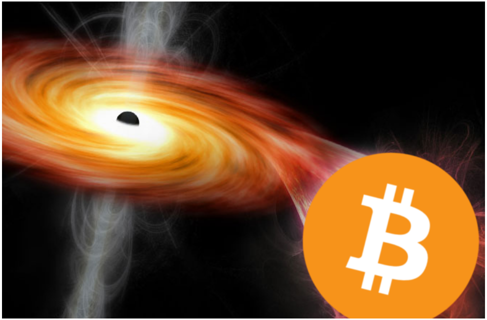
Silly Addresses II

- Convert simple hex values to Base 58, and add correct checksums!

Hex String	Address	Balance
`000...000`	`11111...oLvT2`	\$22 900
`000...001`	`11111...Zbvjr`	\$5
`AAA...AAA`	`1GZQK...R1zmr`	\$10
`FFF...FFF`	`1QLbz...5j6Qr`	\$5

Adding It Up

- After many queries, I was able to verify at least 128 BTC ($\sim$ \$230 000) is currently, and forevermore will be, in limbo.



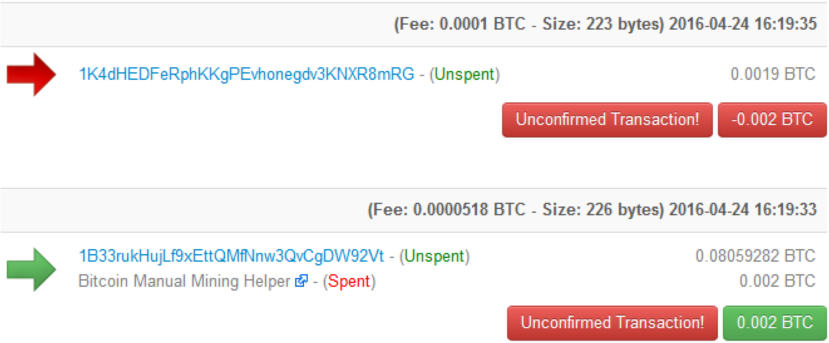
Weak Private Keys

- What happens if you send money to an address that was generated from a really weak private key?

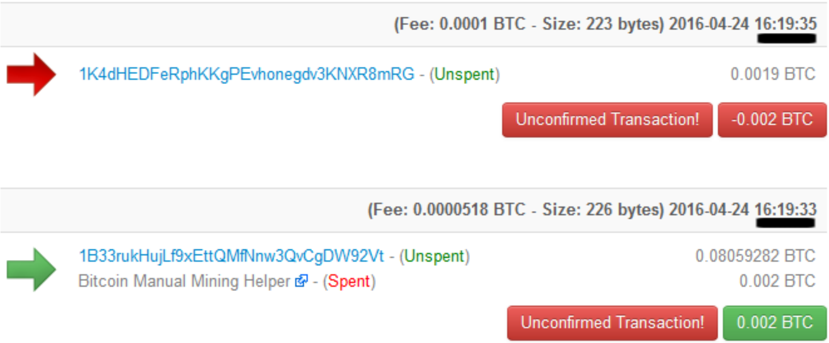
Weak Private Keys

- What happens if you send money to an address that was generated from a really weak private key?
- Anyone who knows how addresses are built can claim it!
- I sent \$1 to `1EHNa...F6kZm` ($d = 1$).

Sending $1 to 1EHNa...F6kZm



Sending \$1 to 1EHNa...F6kZm



Sending $1 to 1EHNa...F6kZm

(Fee: 0.0001 BTC - Size: 233 bytes) 2016-04-24 16:23:53

1aa5cmqmvQq8YQTEqcTmW7dfBNuFwgdCD - (Unspent) 0.0019 BTC

Unconfirmed Transaction! -0.002 BTC

(Fee: 0.0001 BTC - Size: 223 bytes) 2016-04-24 16:19:35

1K4dHEDFeRphKKgPEvhonegdv3KNXR8mRG - (Unspent) 0.0019 BTC

Unconfirmed Transaction! -0.002 BTC

(Fee: 0.0000518 BTC - Size: 226 bytes) 2016-04-24 16:19:33

1B33rukHujLf9xEttQMfNnw3QvCgDW92Vt - (Unspent) 0.08059282 BTC

Bitcoin Manual Mining Helper - (Spent, Spent) 0.002 BTC

Unconfirmed Transaction! 0.002 BTC

Sending \$1 to 1EHNa...F6kZm

(Fee: 0.0001 BTC - Size: 233 bytes) 2016-04-24 16:23:53

1aa5cmqmvQq8YQTEqcTmW7dfBNuFwgdCD - (Unspent) 0.0019 BTC

-0.002 BTC

(Fee: 0.0001 BTC - Size: 223 bytes) 2016-04-24 16:19:35

1K4dHEDFeRphKKgPEvhonegdv3KNXR8mRG - (Unspent) 0.0019 BTC

Unconfirmed Transaction! -0.002 BTC

(Fee: 0.0000518 BTC - Size: 226 bytes) 2016-04-24 16:19:33

1B33rukHujLf9xEttQMfNnw3QvCgDW92Vt - (Unspent) 0.08059282 BTC

Bitcoin Manual Mining Helper - (Spent, Spent) 0.002 BTC

0.002 BTC

Weak Private Key Harvesting

Address	Private Key	Received
`16QaF...AsBFM`	0	\$5
`1EHNa...F6kZm`	1	\$1990
`1JPbz...uha5m`	`#Points - 1`	\$11
`12M4Q...CpST7`	$2^{256}-1$	\$2

- At least \$14 000 USD has been harvested from addresses with weak private keys.

Weak Private Key Harvesting

Address	Private Key	Received
`16QaF...AsBFM`	0	\$5
`1EHNa...F6kZm`	1	\$1990
`1JPbz...uha5m`	`#Points - 1`	\$11
`12M4Q...CpST7`	$2^{256}-1$	\$2

- At least \$14 000 USD has been harvested from addresses with weak private keys.
- Vast majority are from RNG problems!

References

Bitcoin: A Peer-to-Peer Electronic Cash System, *Satoshi Nakamoto*.
https://bitcoin.org/bitcoin.pdf, 2008.

Elliptic Curve Cryptography in Practice, *Bos, Halderman, et al*.
https://eprint.iacr.org/2013/734.pdf, 2013.

SEC2: Recommended Elliptic Curve Parameters, *Certicom Research*.
http://www.secg.org/SEC2-Ver-1.0.pdf, 2000

Bitcoin Wiki: Elliptic Curve Public Key to BTC Address Conversion.
https://en.bitcoin.it/wiki/File:PubKeyToAddr.png

Blockchain.info API Library (Python, v1)
https://github.com/blockchain/api-v1-client-python

Blockchain.info
https://blockchain.info