Flexible Models for Secure Systems

Sarah Meiklejohn

A creation story

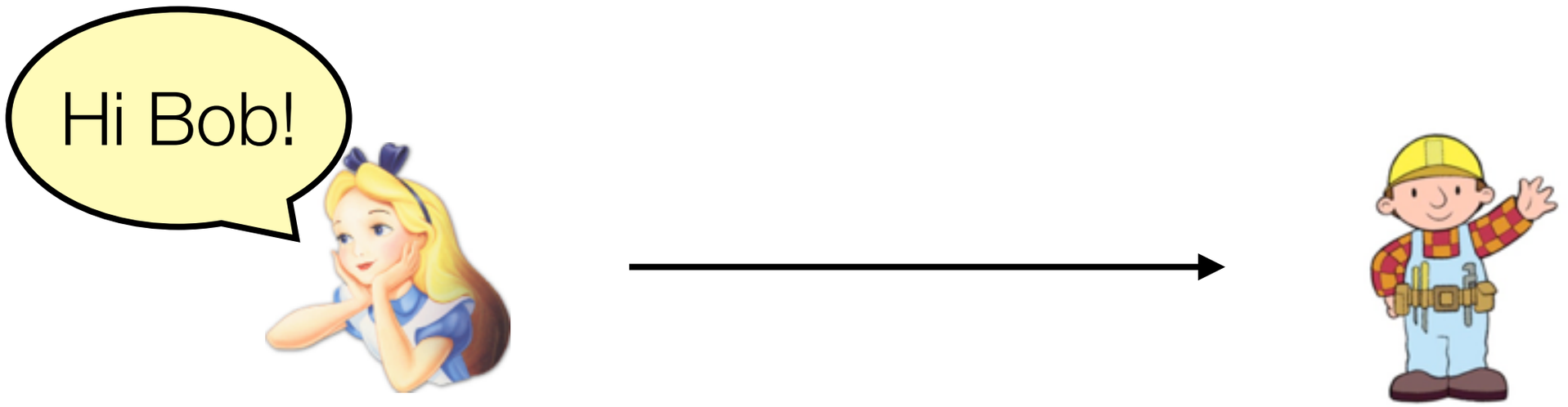
Motivating scenario that founded modern cryptography:

A creation story

Motivating scenario that founded modern cryptography:

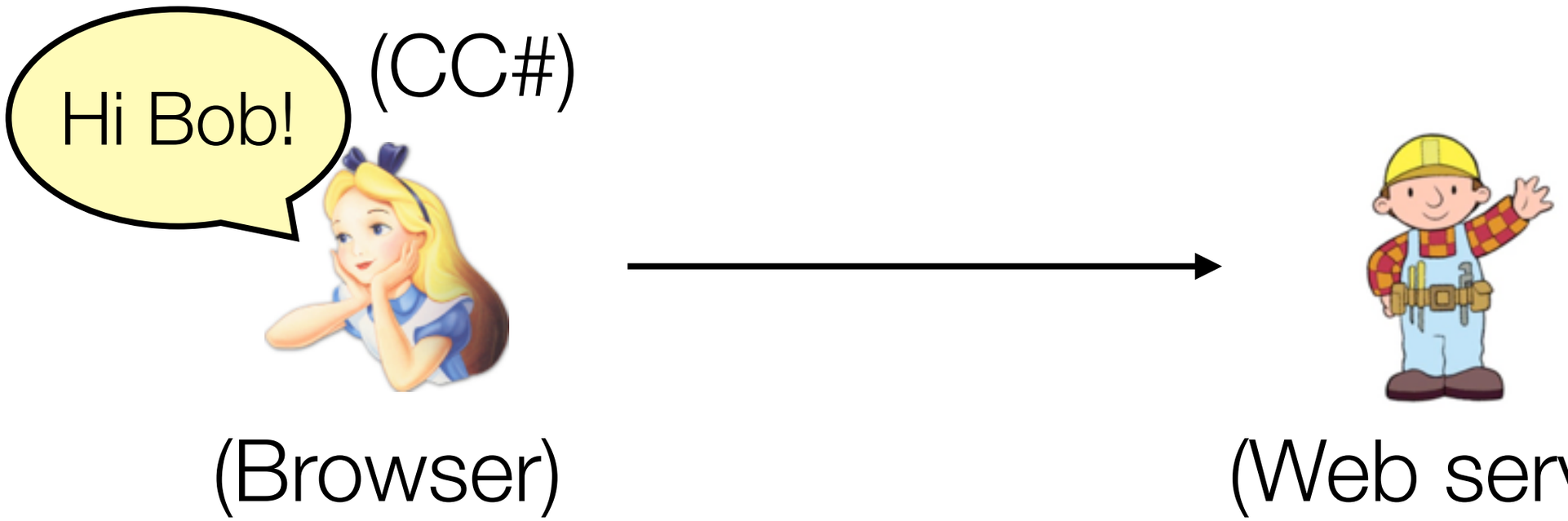
A creation story

Motivating scenario that founded modern cryptography:



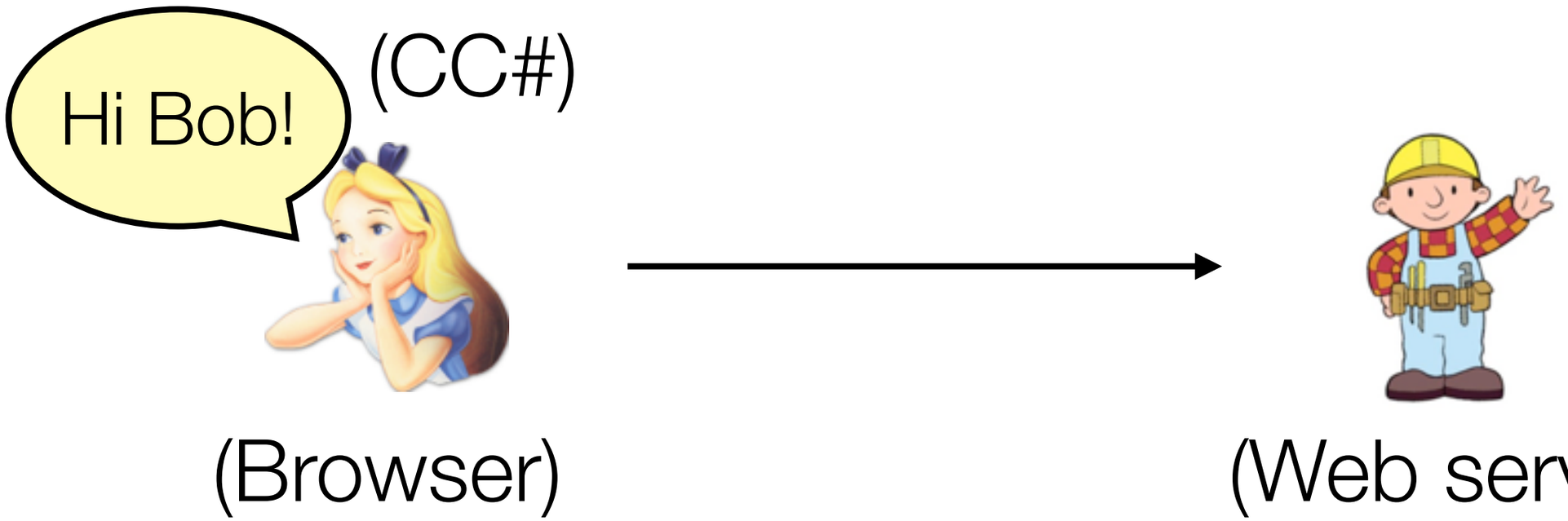
A creation story

Motivating scenario that founded modern cryptography:



A creation story

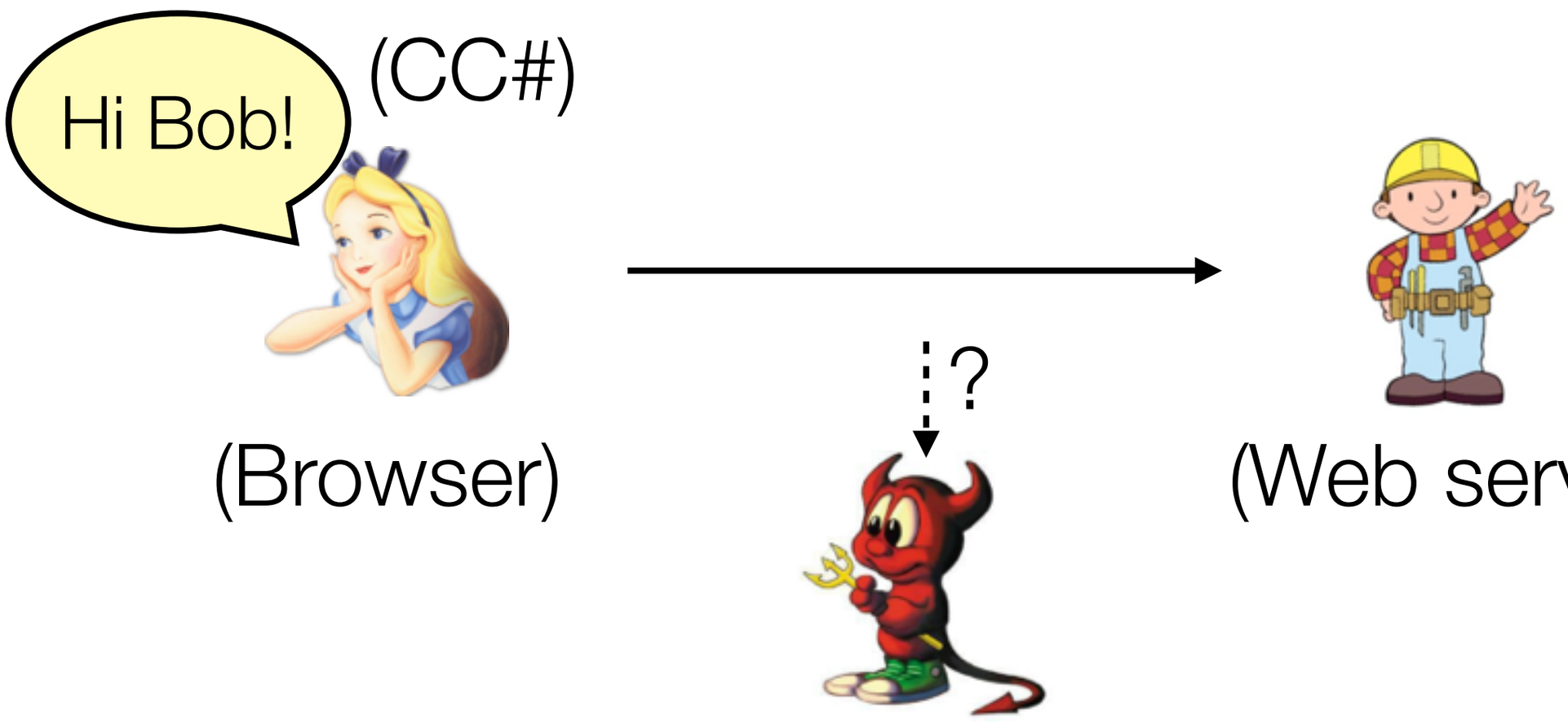
Motivating scenario that founded modern cryptography:



Security model for this interaction is well established

A creation story

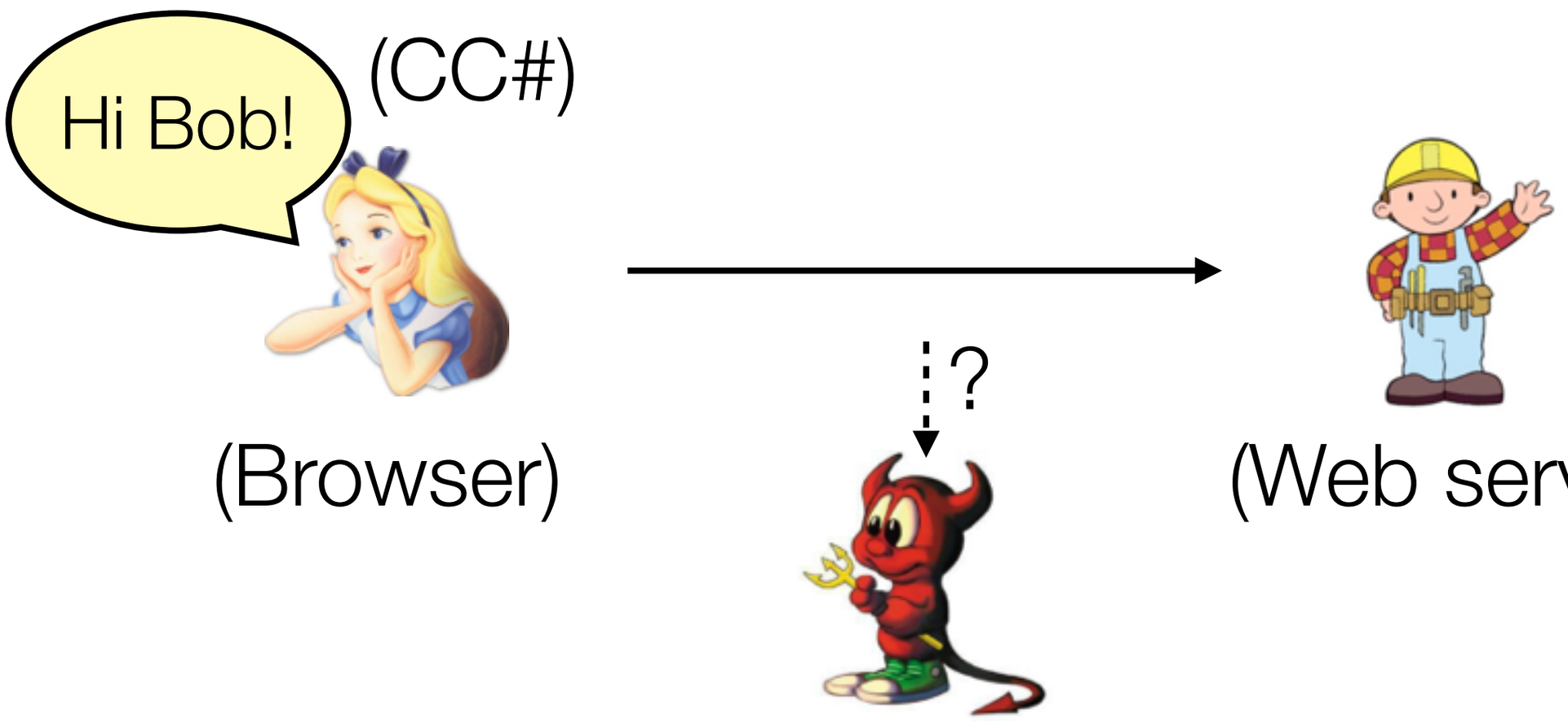
Motivating scenario that founded modern cryptography:



Security model for this interaction is well established

A creation story

Motivating scenario that founded modern cryptography:

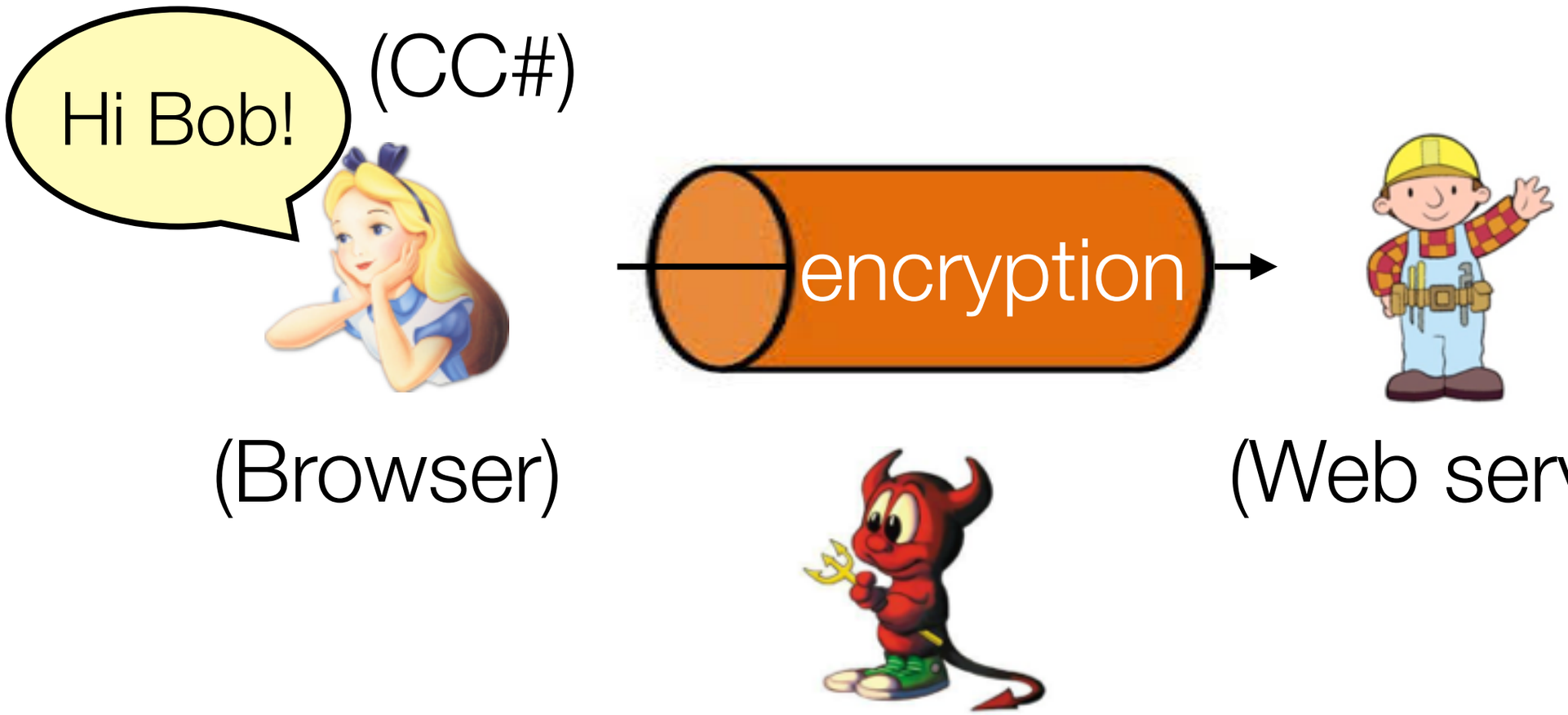


Security model for this interaction is well established

Encryption works for secure online communication: SSL/TLS [1996]

A creation story

Motivating scenario that founded modern cryptography:

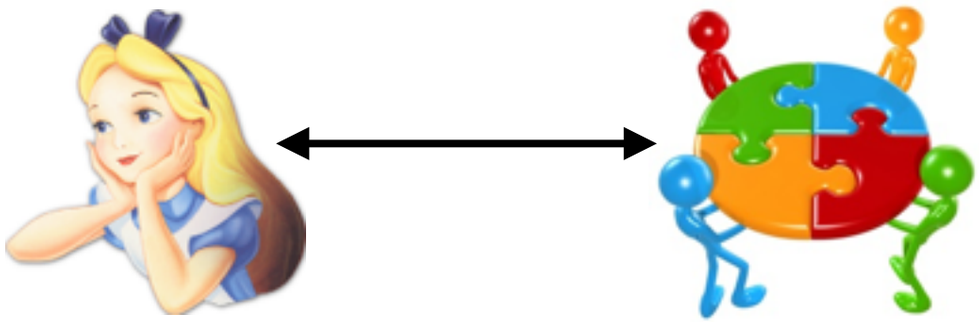


Security model for this interaction is well established

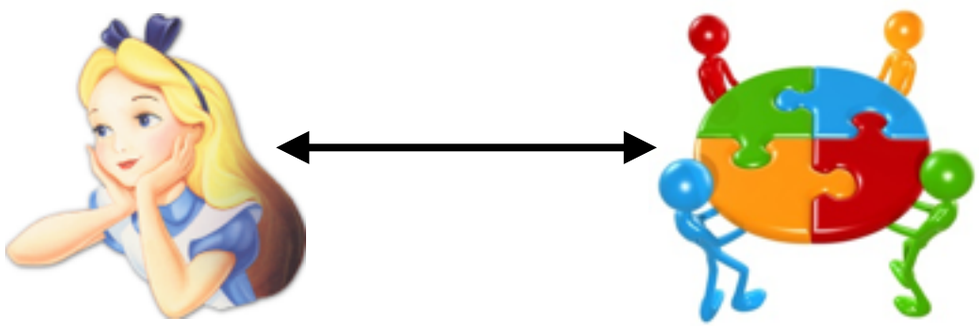
Encryption works for secure online communication: SSL/TLS [1996]

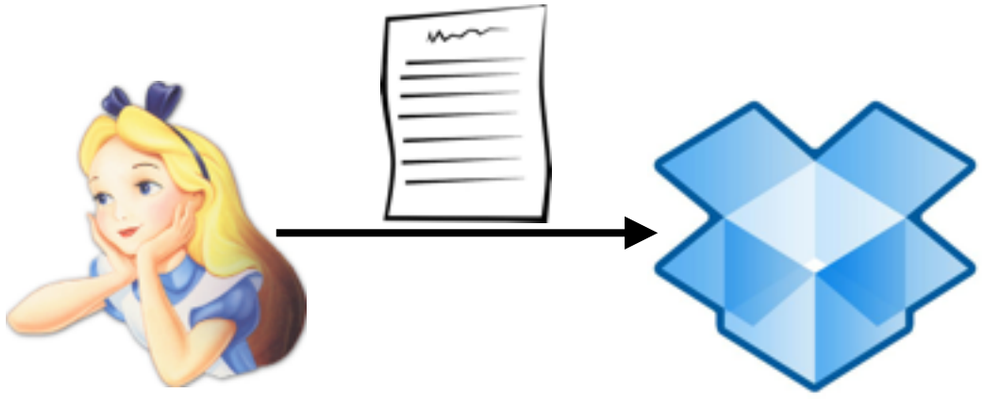
Fast-forward 15 years…

Fast-forward 15 years…

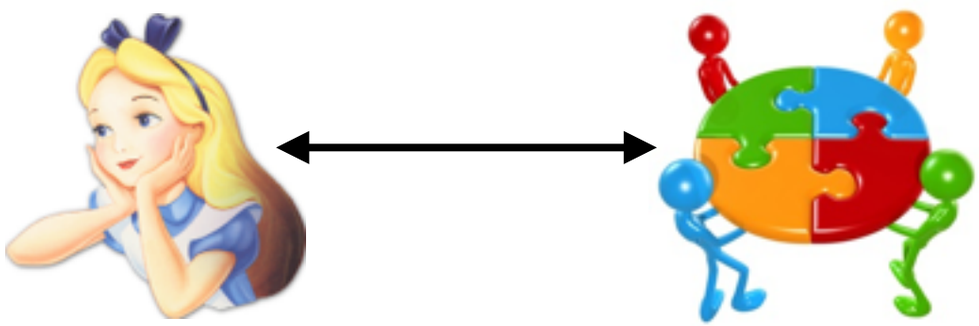


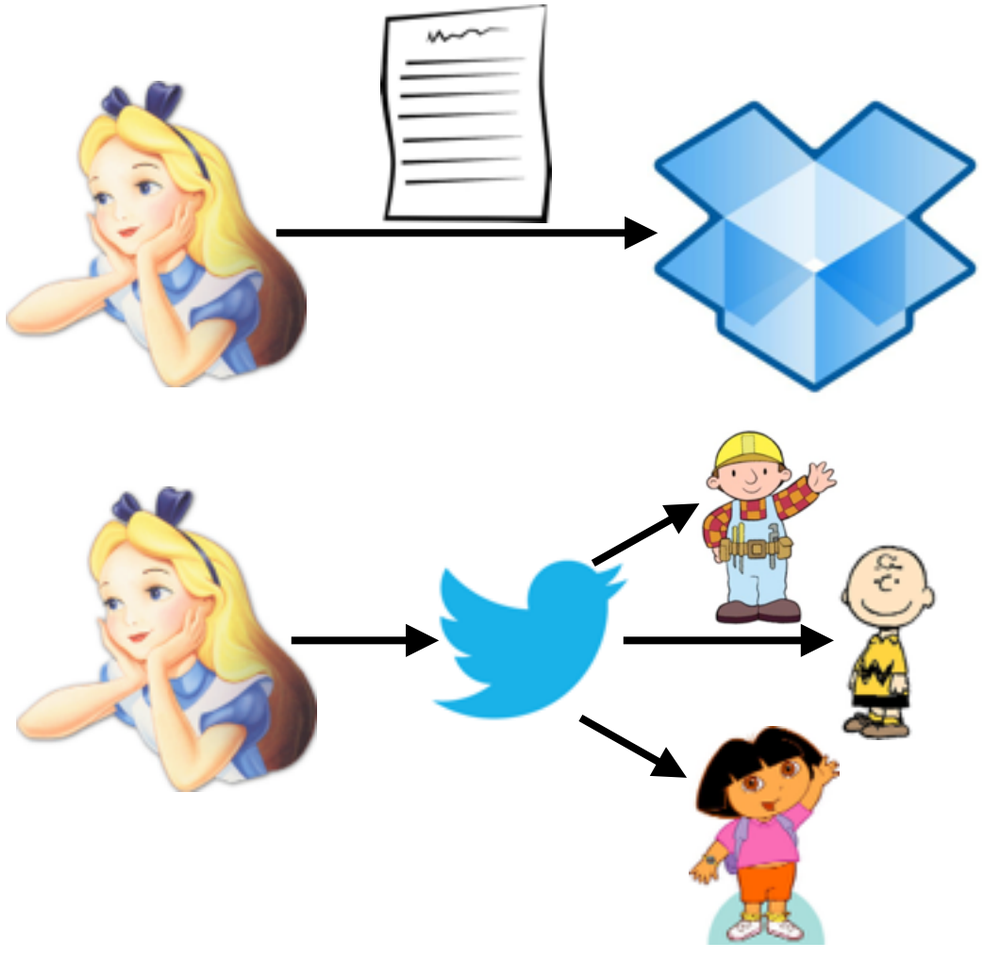
Fast-forward 15 years…



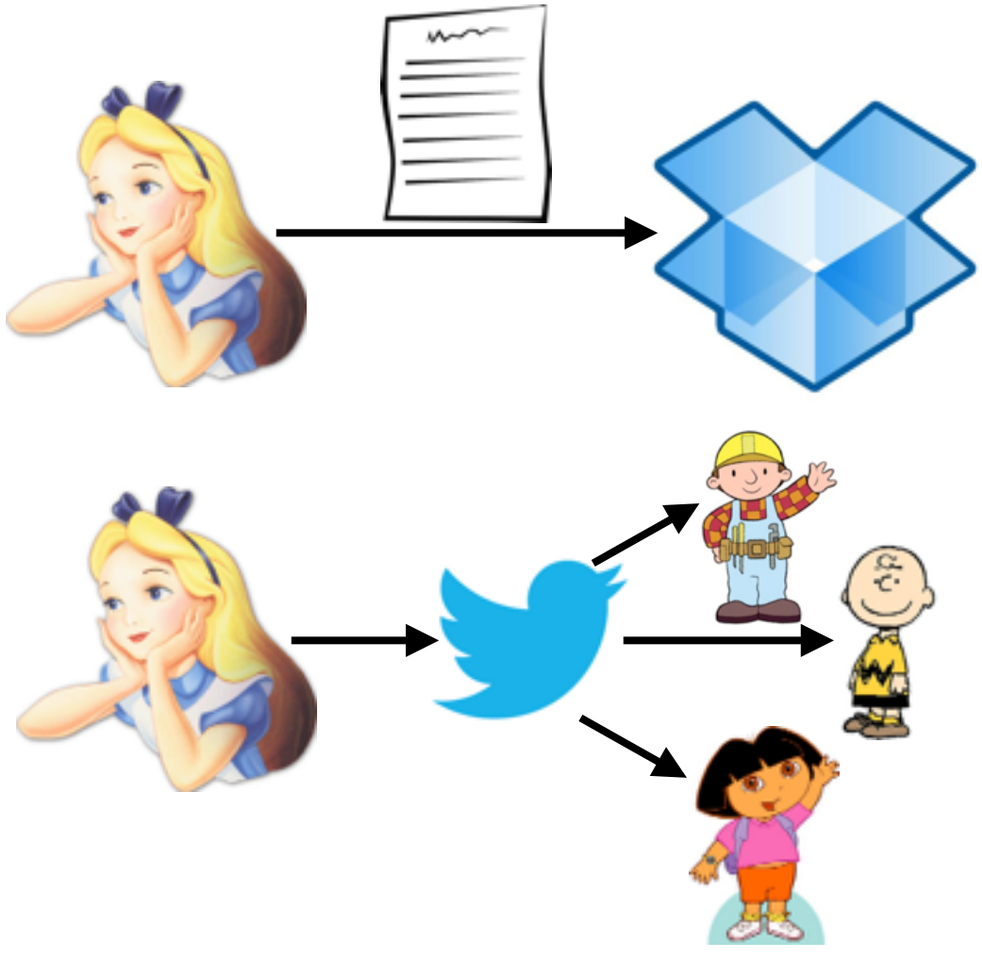


Fast-forward 15 years…

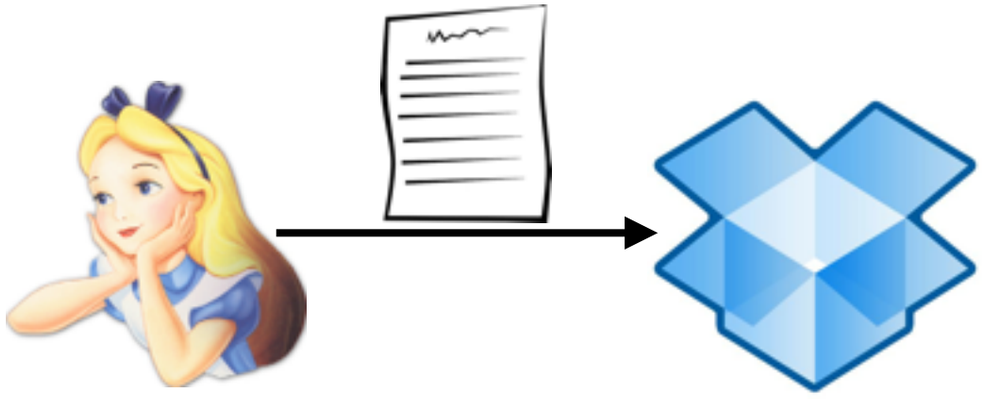


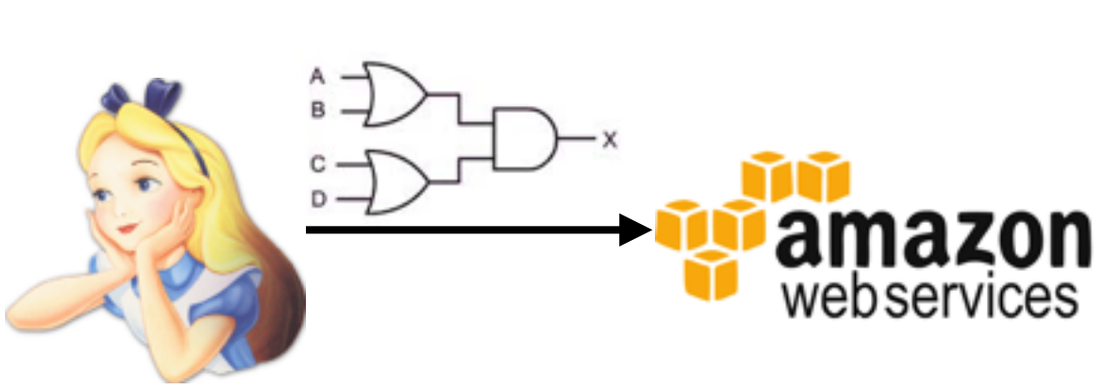


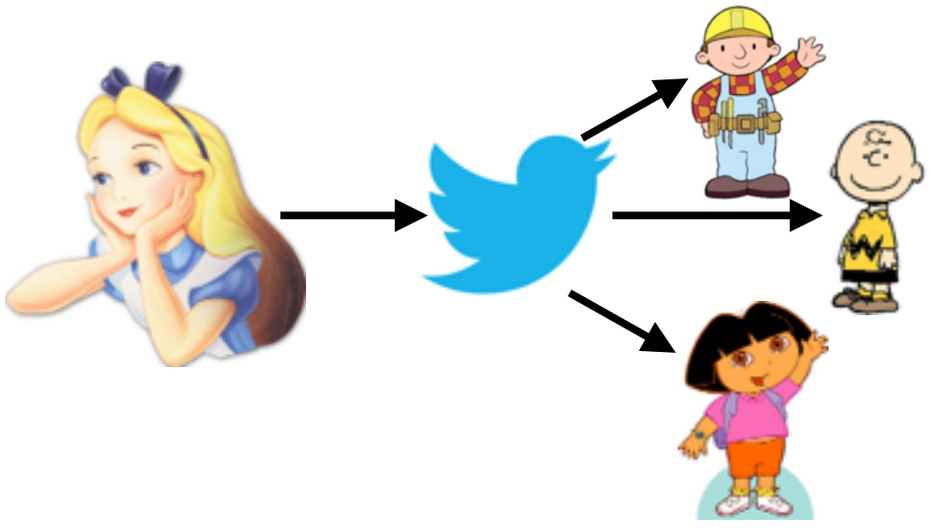
Fast-forward 15 years…



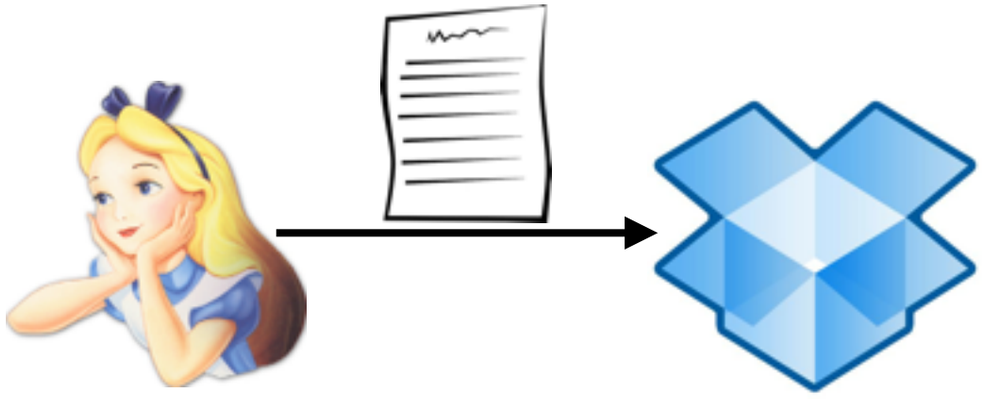
Fast-forward 15 years…

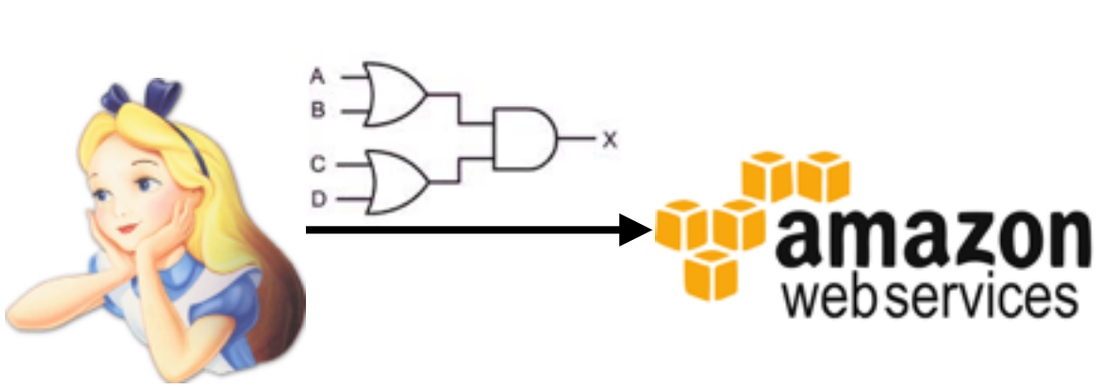


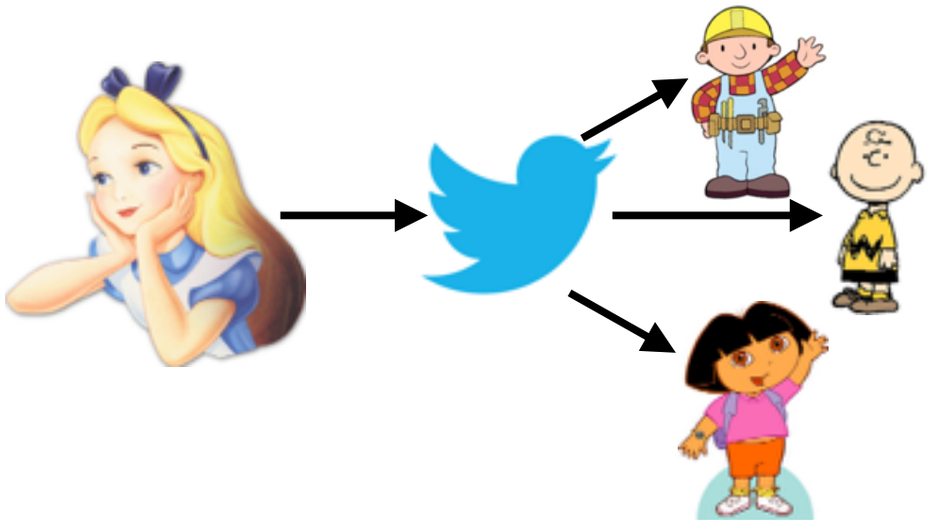


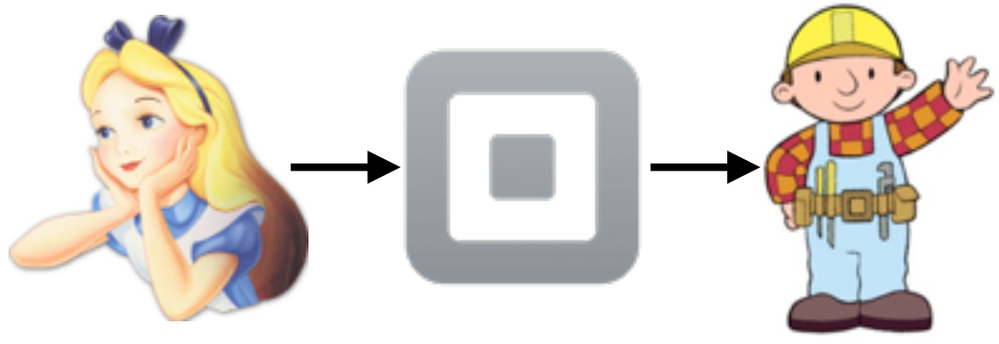


Fast-forward 15 years…



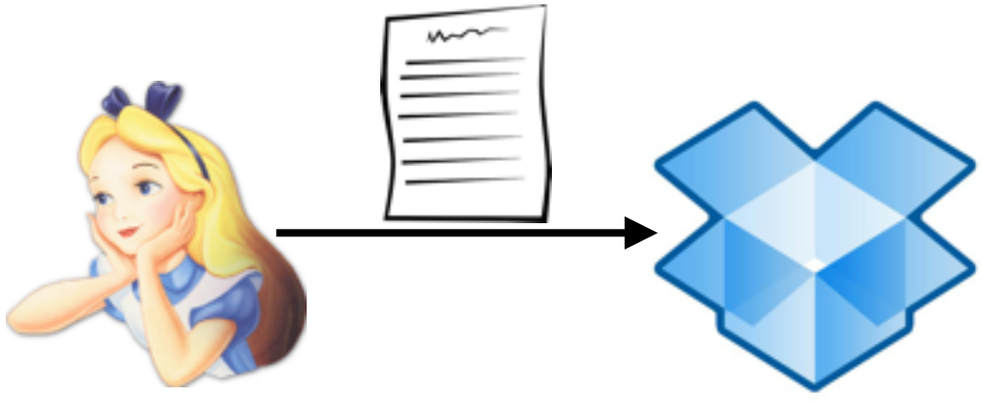


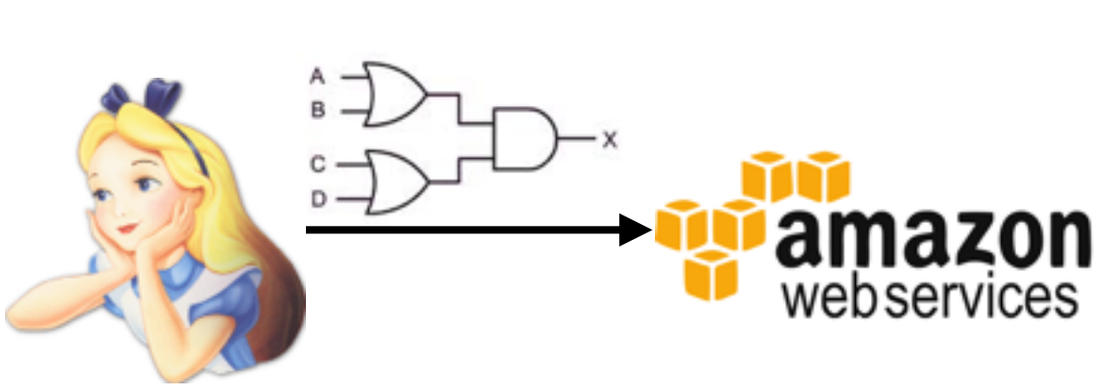


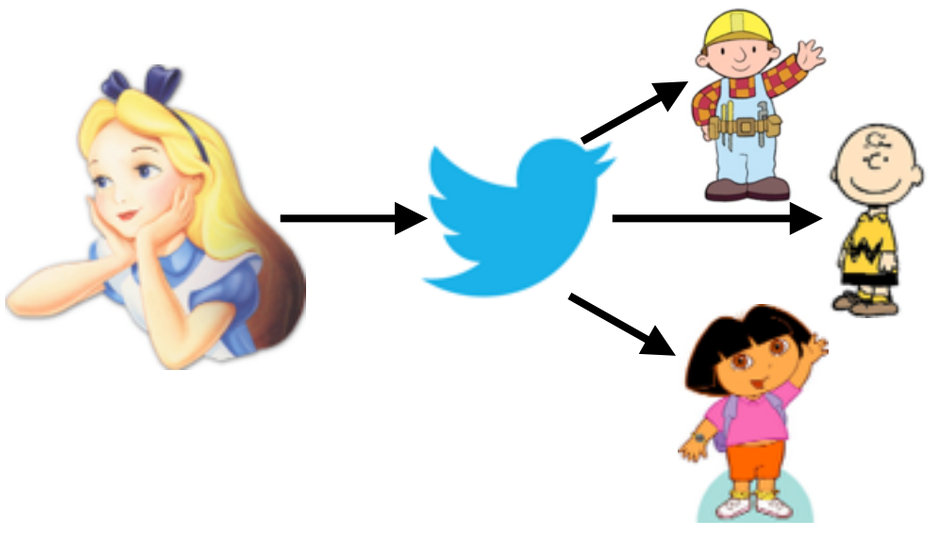


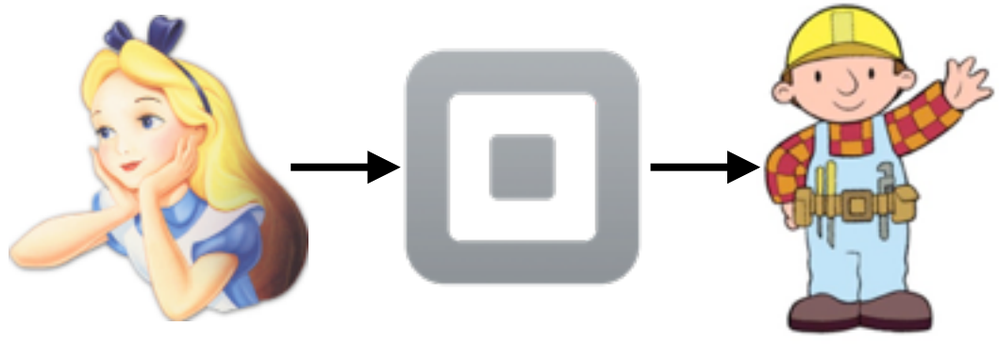
Fast-forward 15 years…

Problem #1: everyone is an adversary!



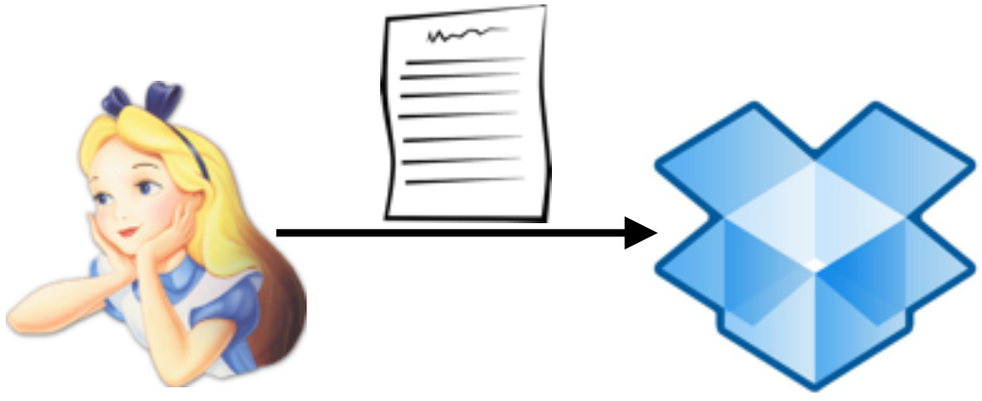


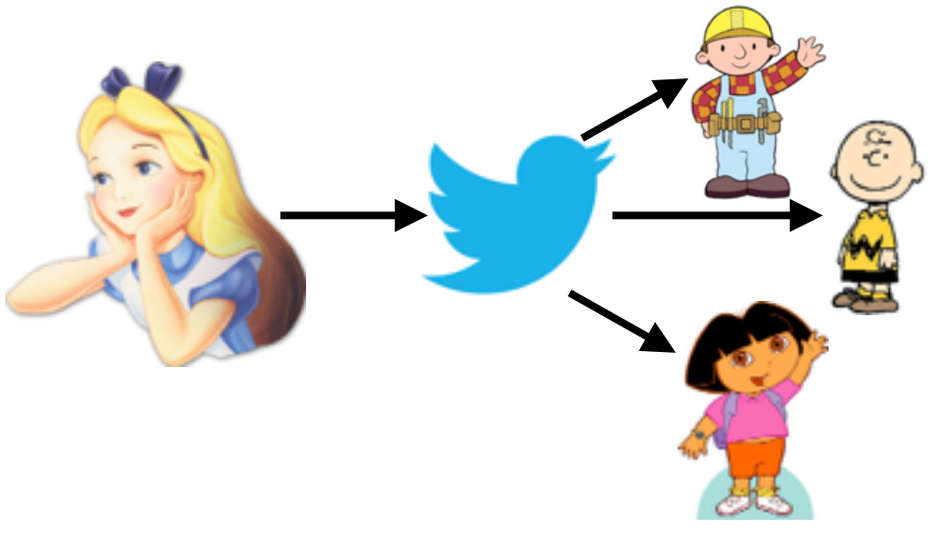


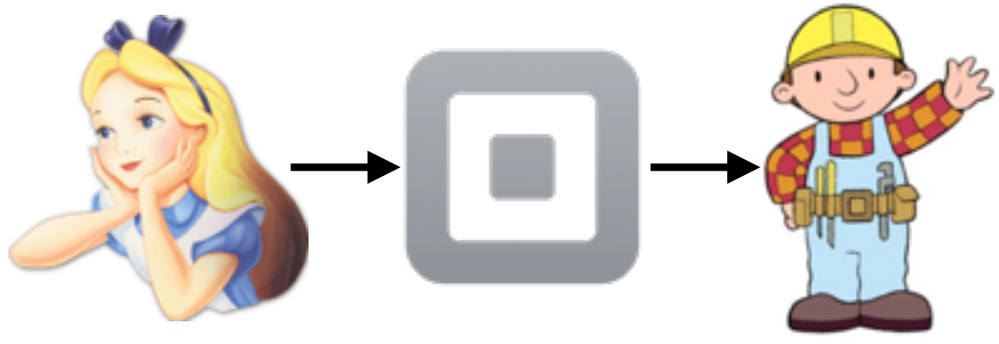


Fast-forward 15 years…

Problem #1: **everyone is an adversary!**

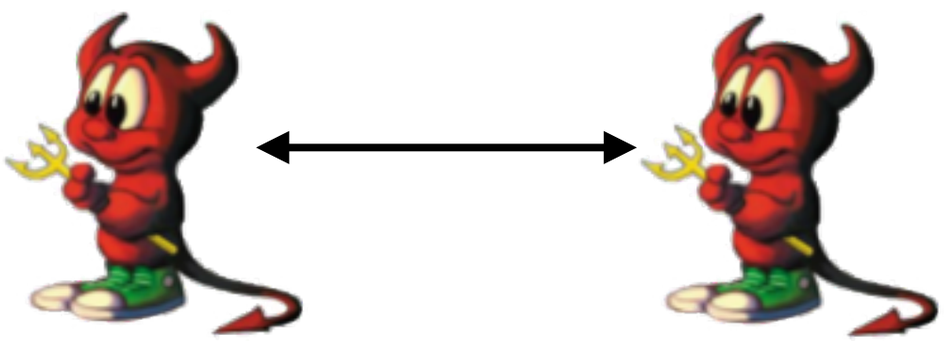


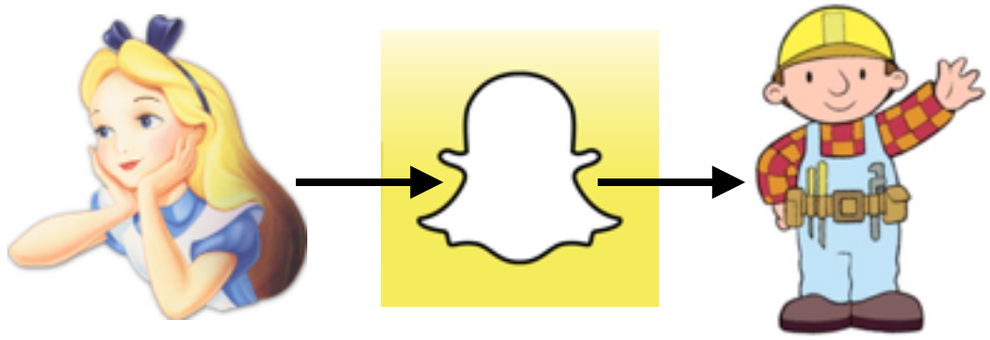


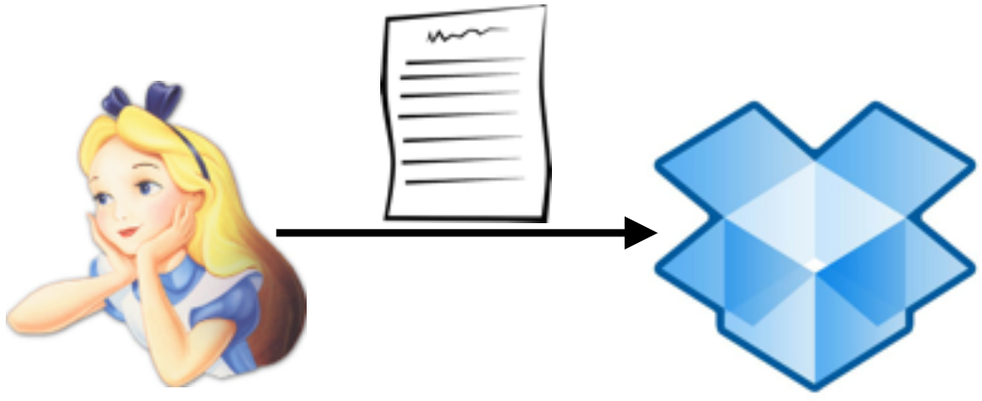


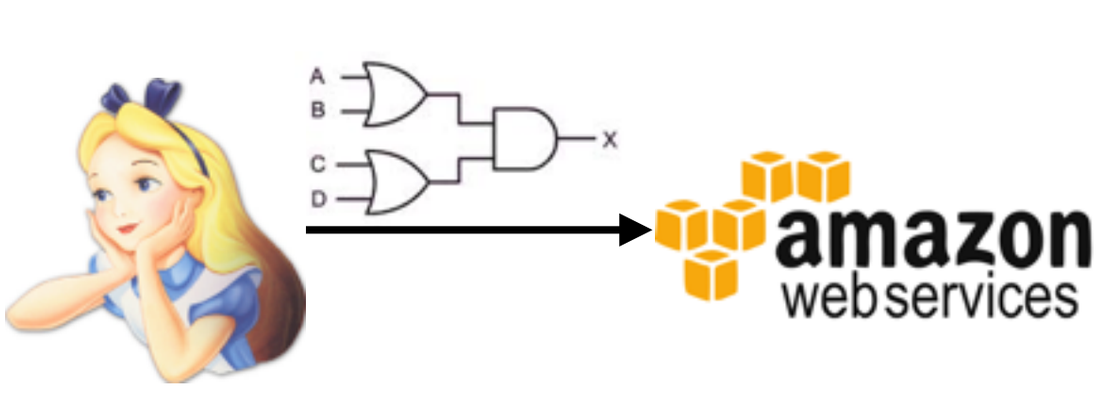
Fast-forward 15 years…

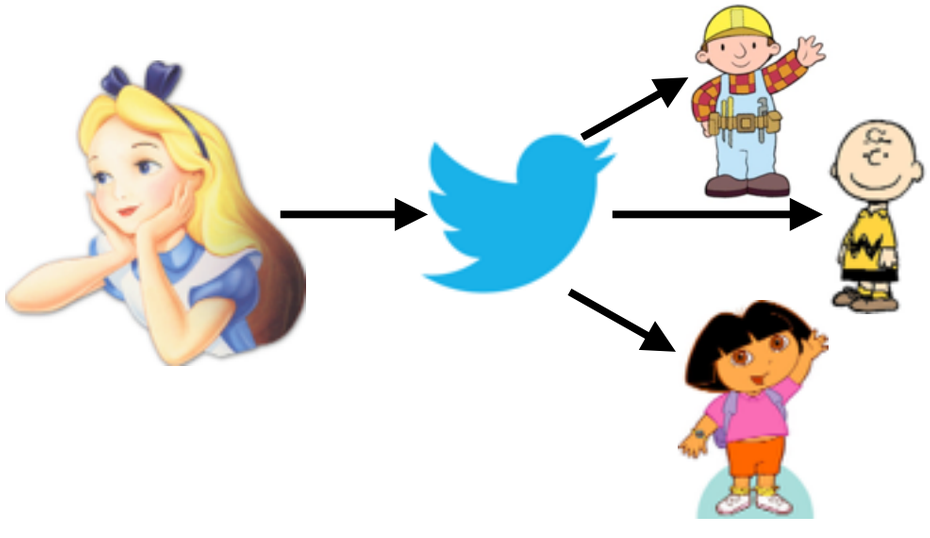
Problem #1: everyone is an adversary!

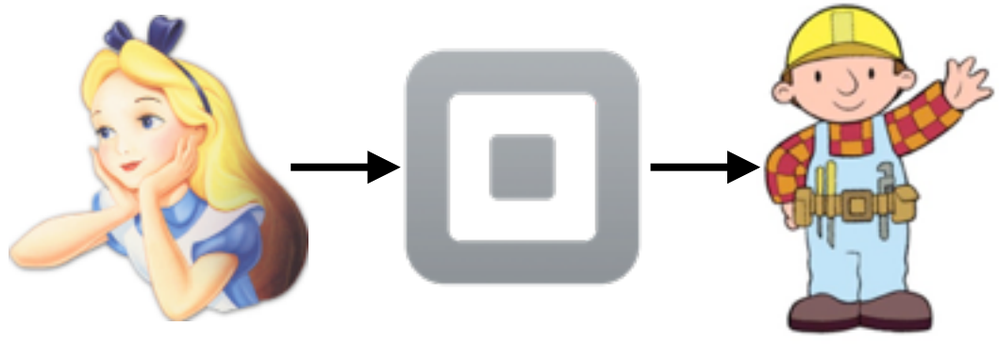






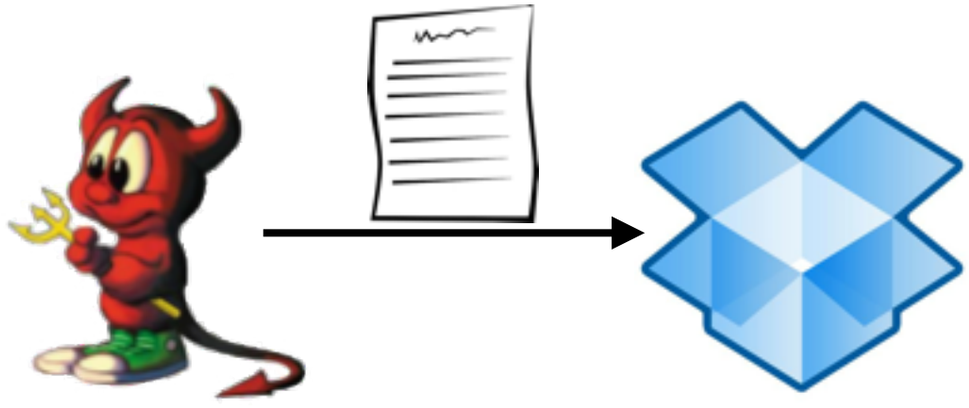


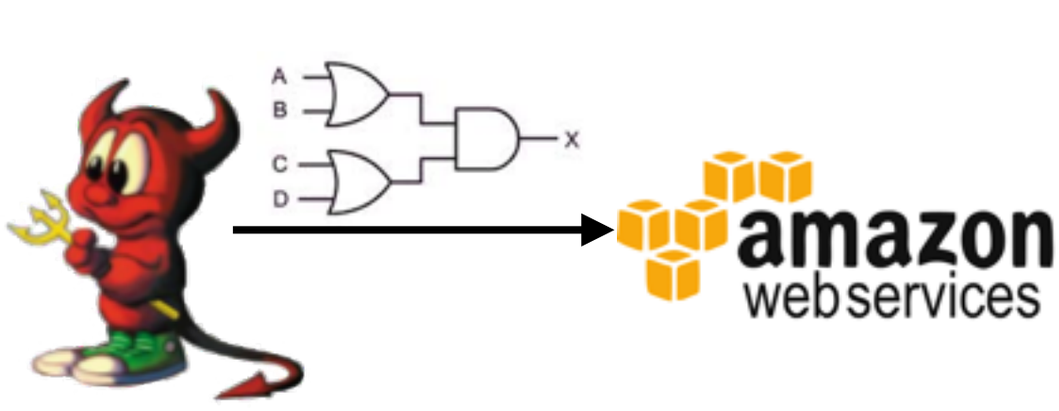


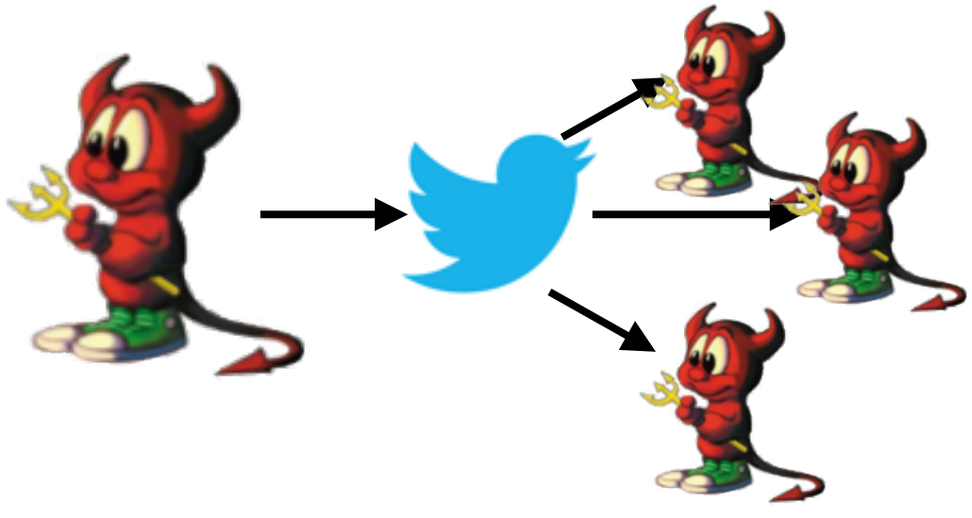


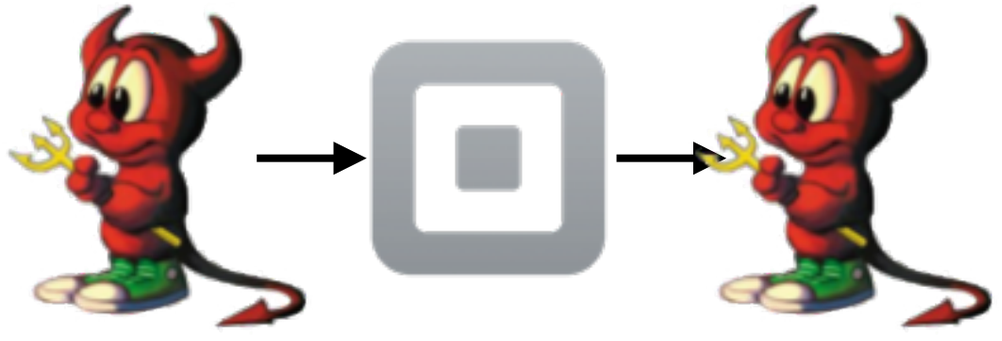
Fast-forward 15 years…

Problem #1: everyone is an adversary!



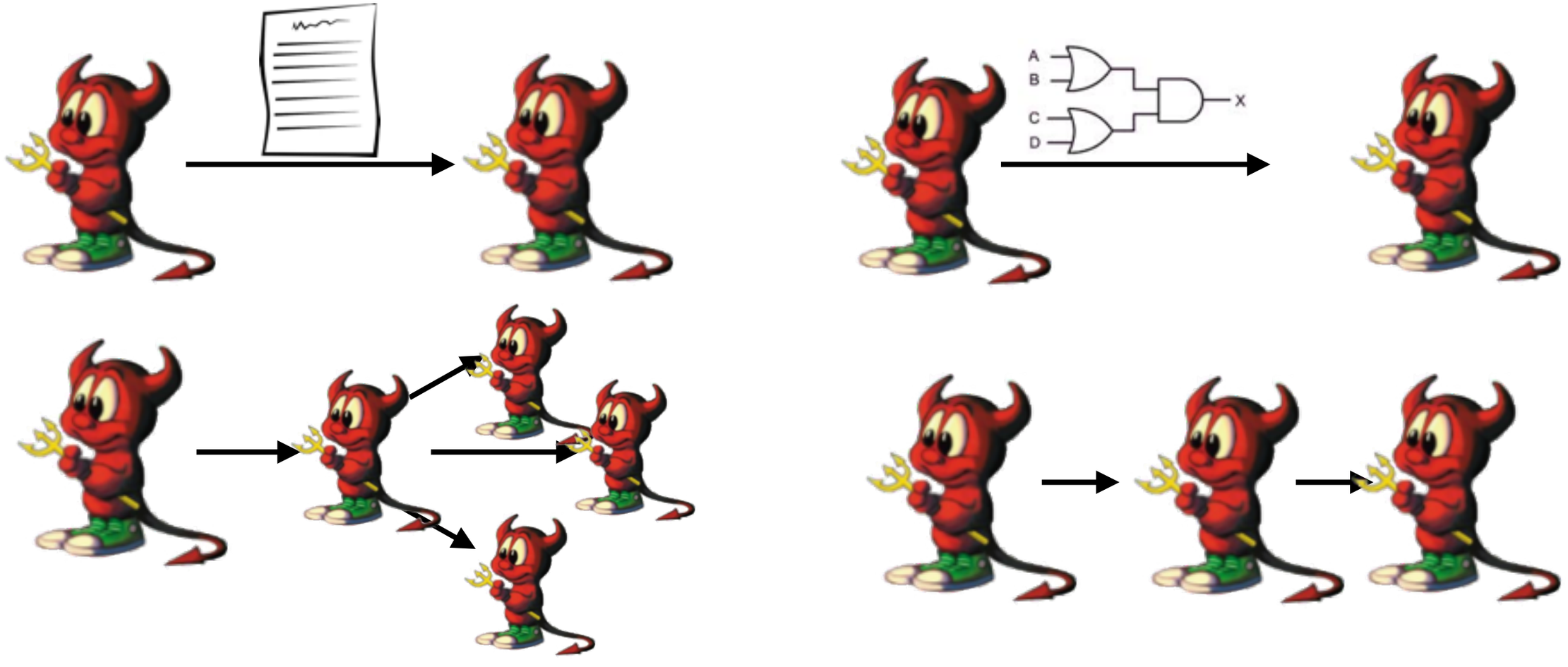






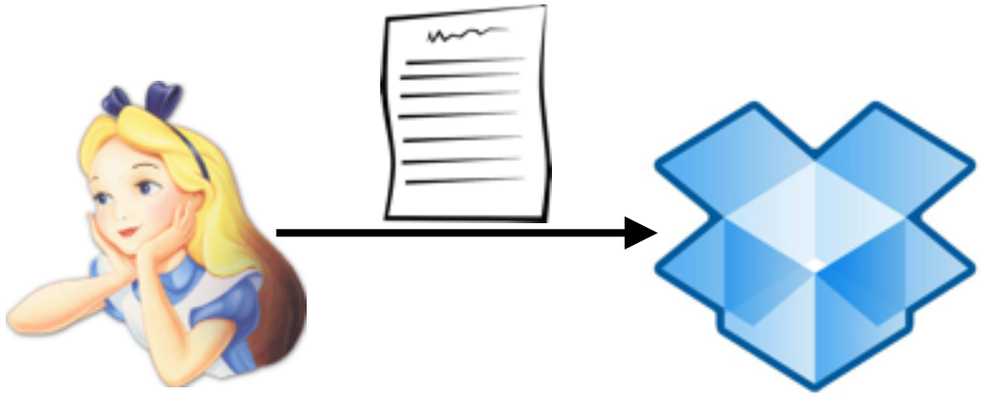
Fast-forward 15 years…

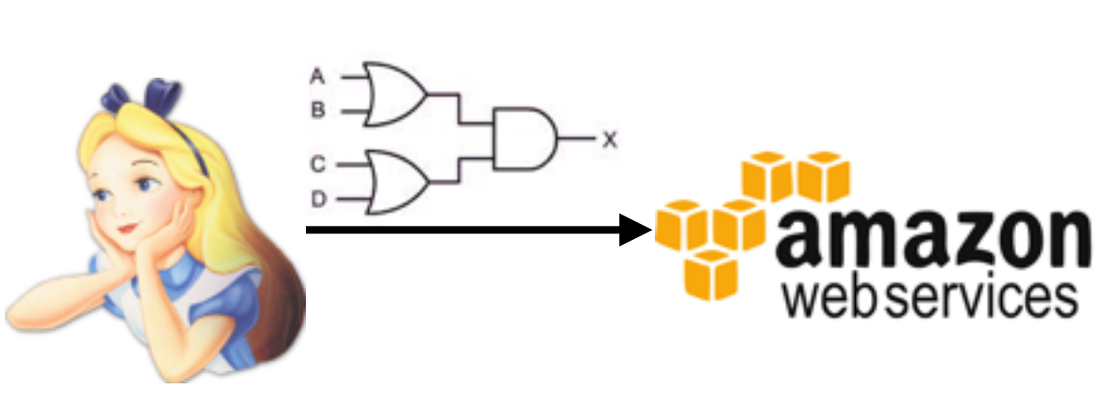
Problem #1: **everyone is an adversary!**

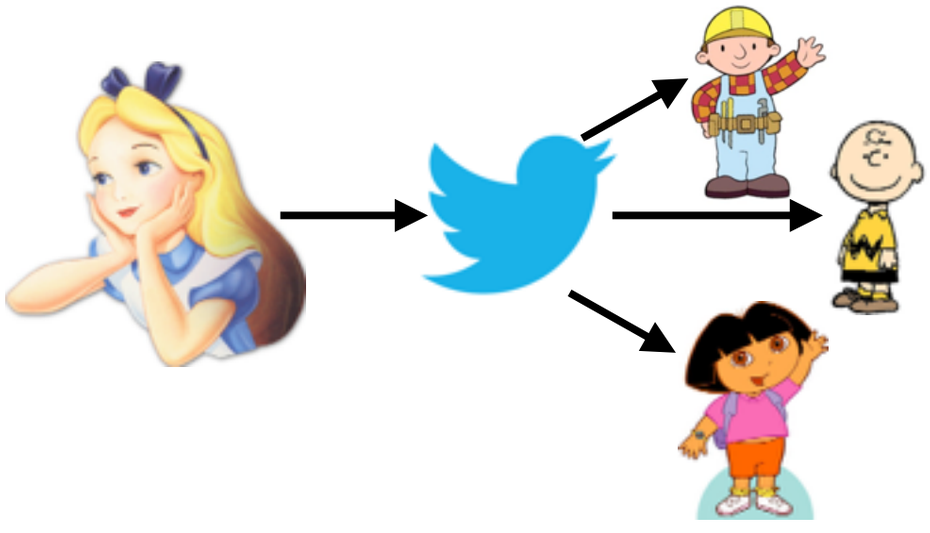


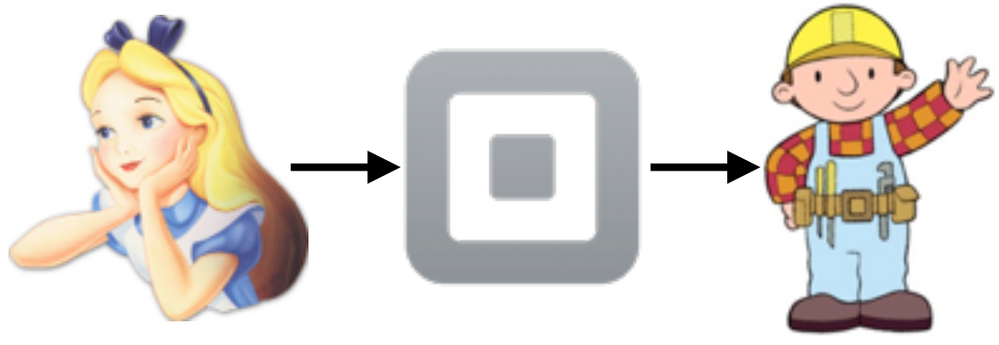
Fast-forward 15 years…

Problem #2: what are the security goals?



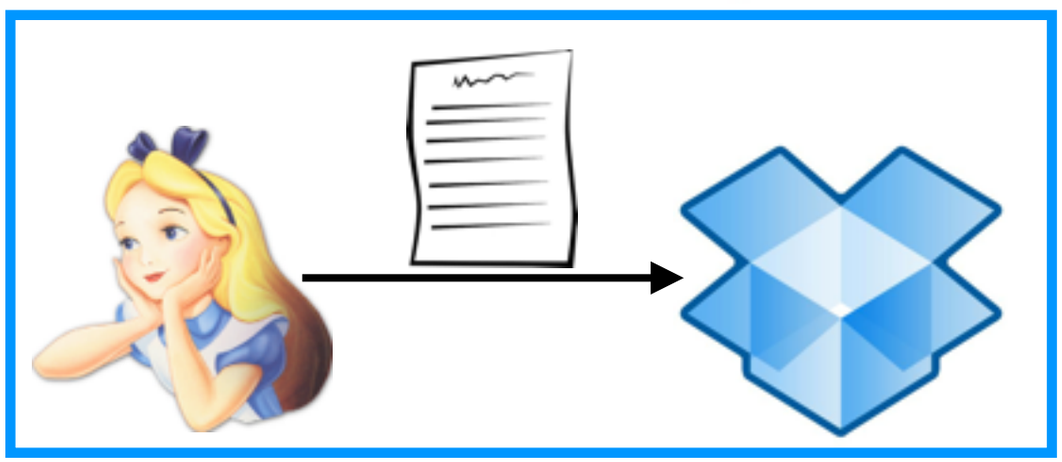


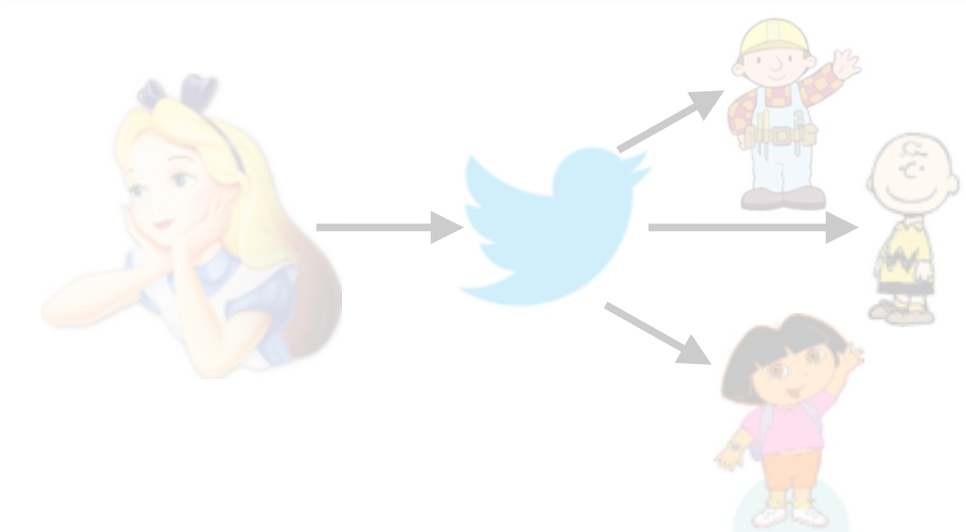


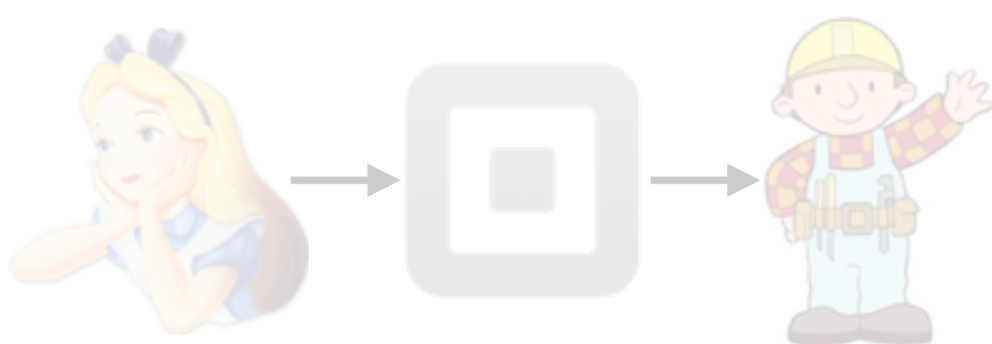


Fast-forward 15 years…

Problem #2: what are the security goals?

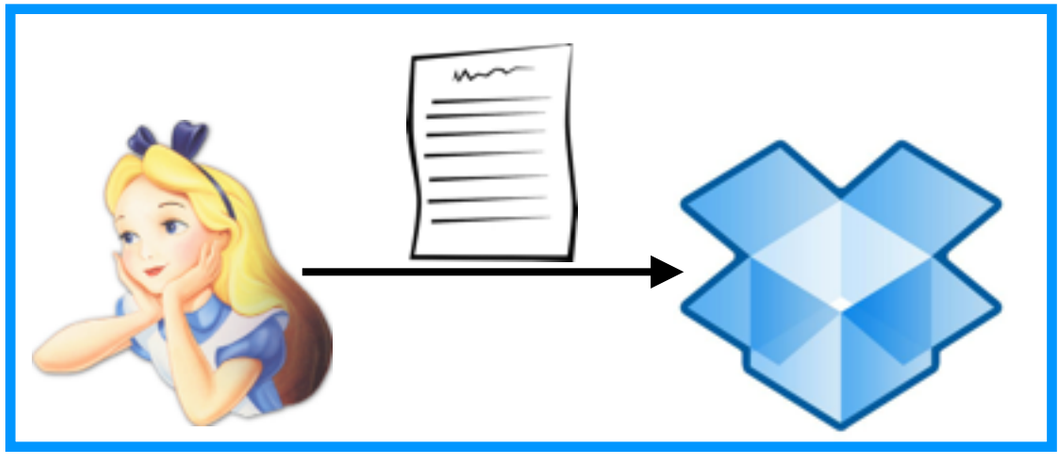




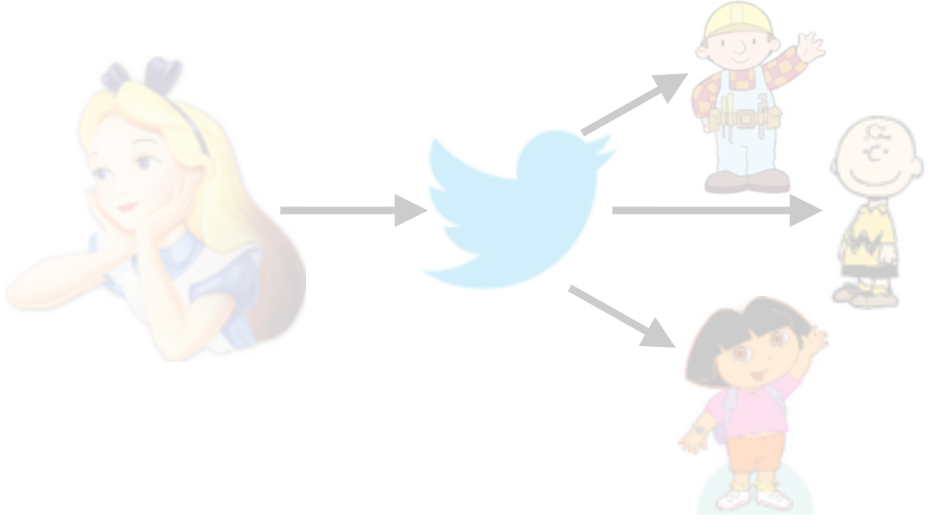


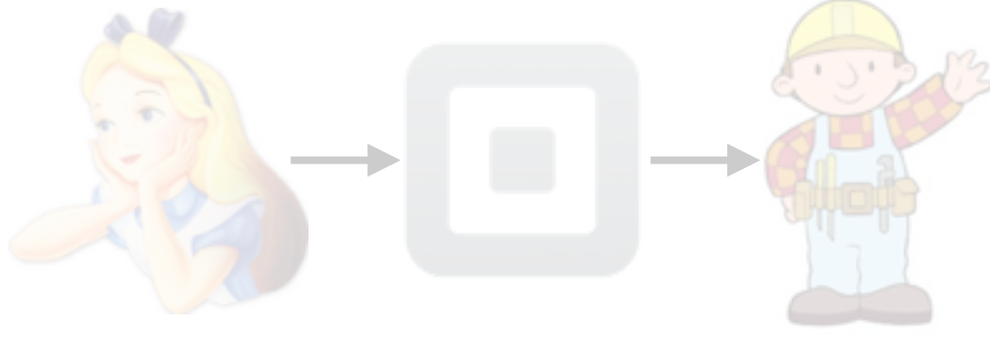
Fast-forward 15 years…

Problem #2: what are the security goals?



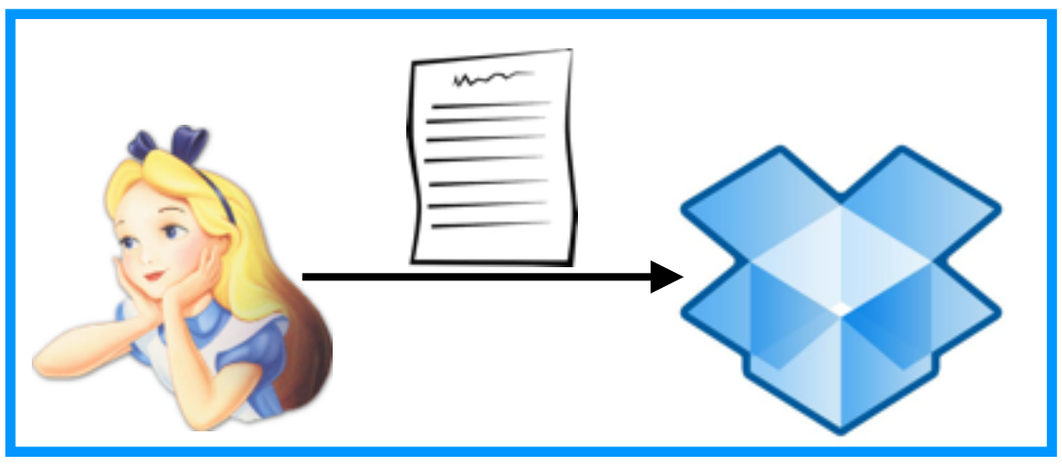
- can Bob retrieve Alice's files?



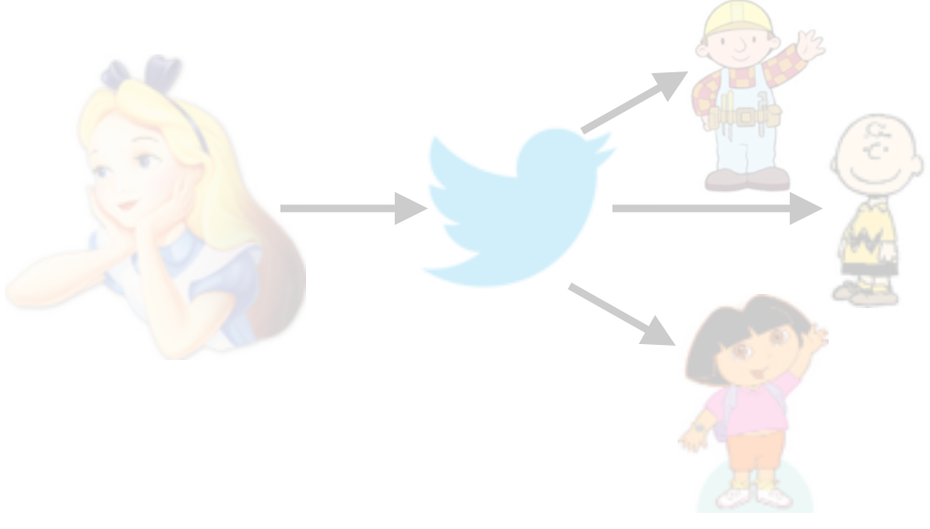


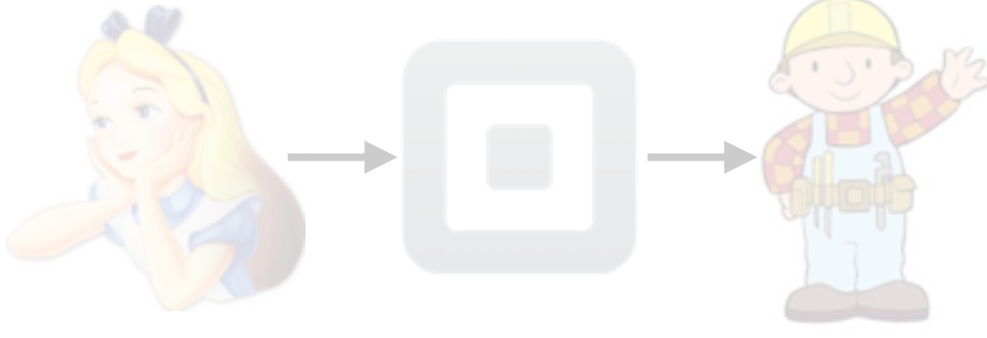
Fast-forward 15 years…

Problem #2: what are the security goals?



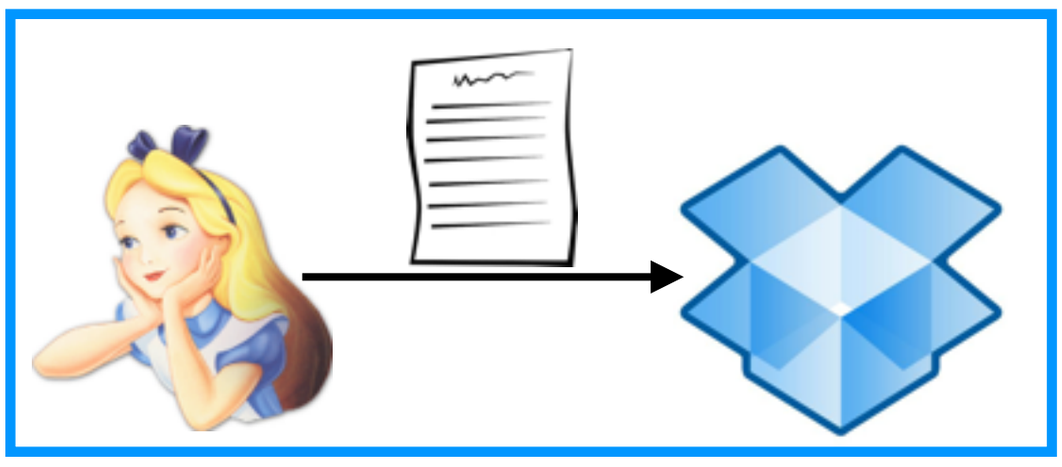
- can Bob retrieve Alice's files?
- can Dropbox read Alice's files?



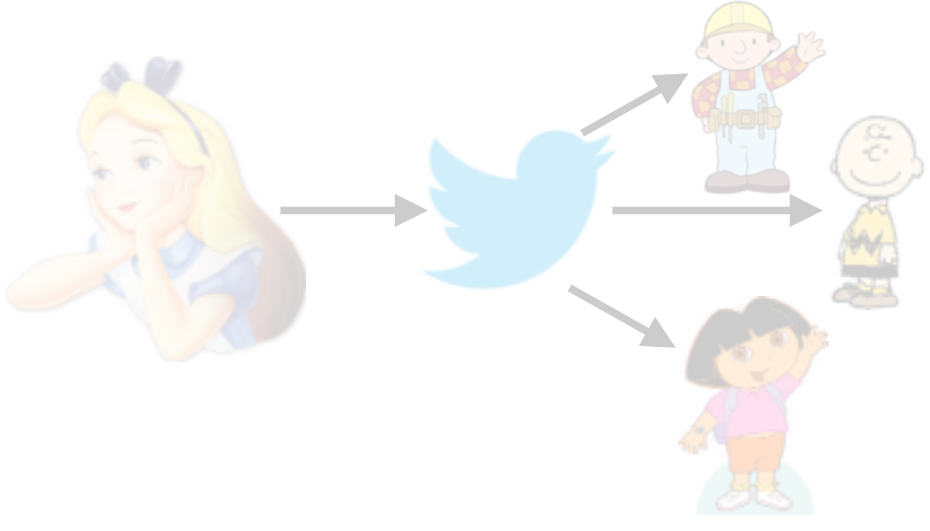


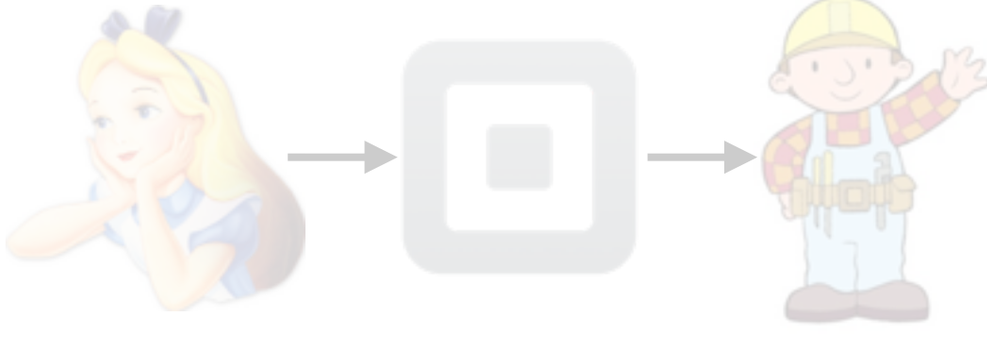
Fast-forward 15 years…

Problem #2: what are the security goals?



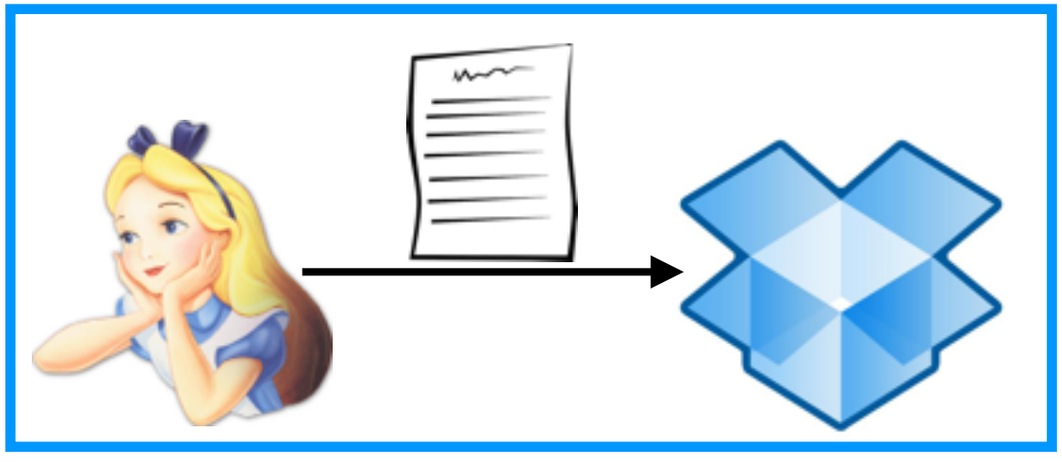
- can Bob retrieve Alice's files?
- can Dropbox read Alice's files?
- did Dropbox delete Alice's files?



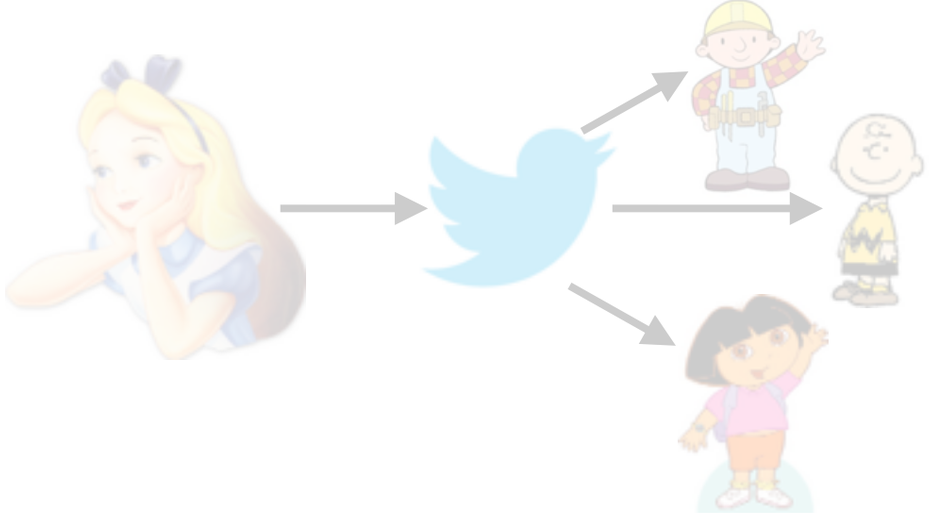


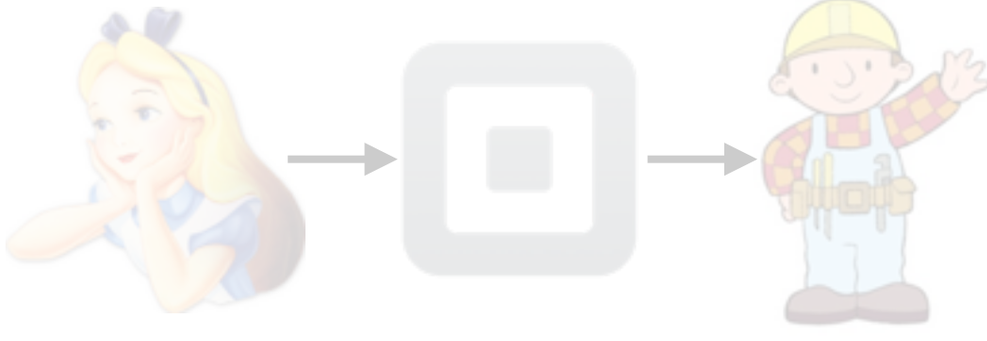
Fast-forward 15 years…

Problem #2: what are the security goals?



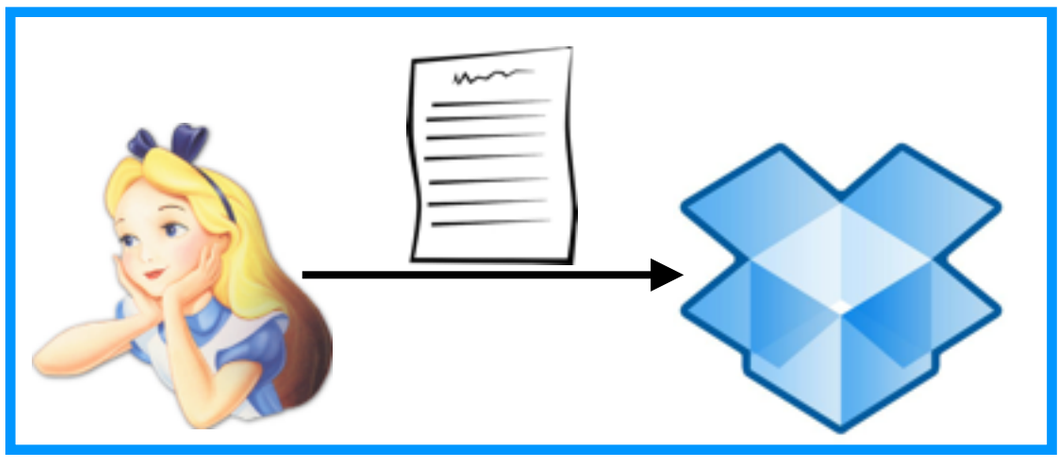
- can Bob retrieve Alice's files?
- can Dropbox read Alice's files?
- did Dropbox delete Alice's files?
- can Dropbox efficiently store files?



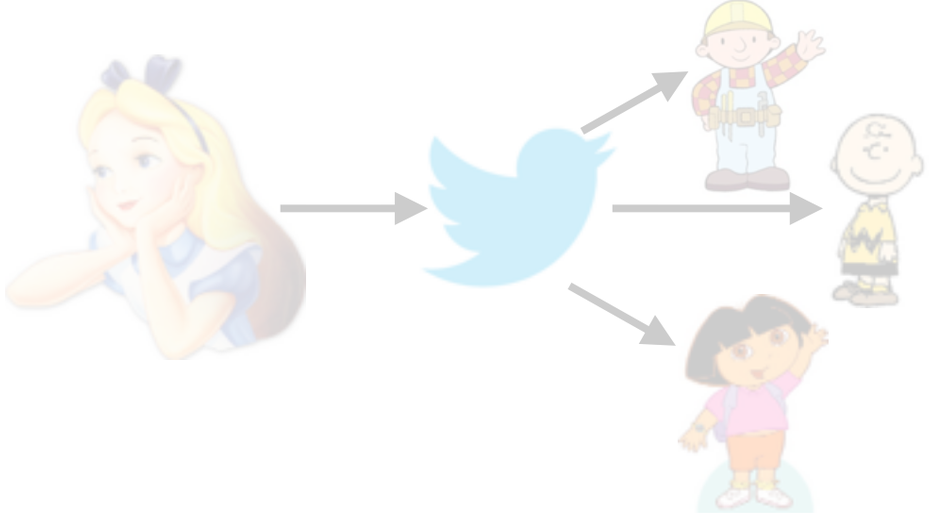


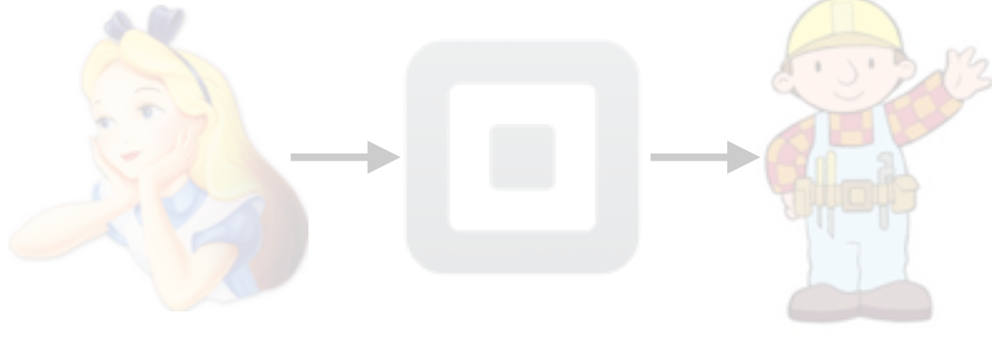
Fast-forward 15 years…

Problem #2: what are the security goals?



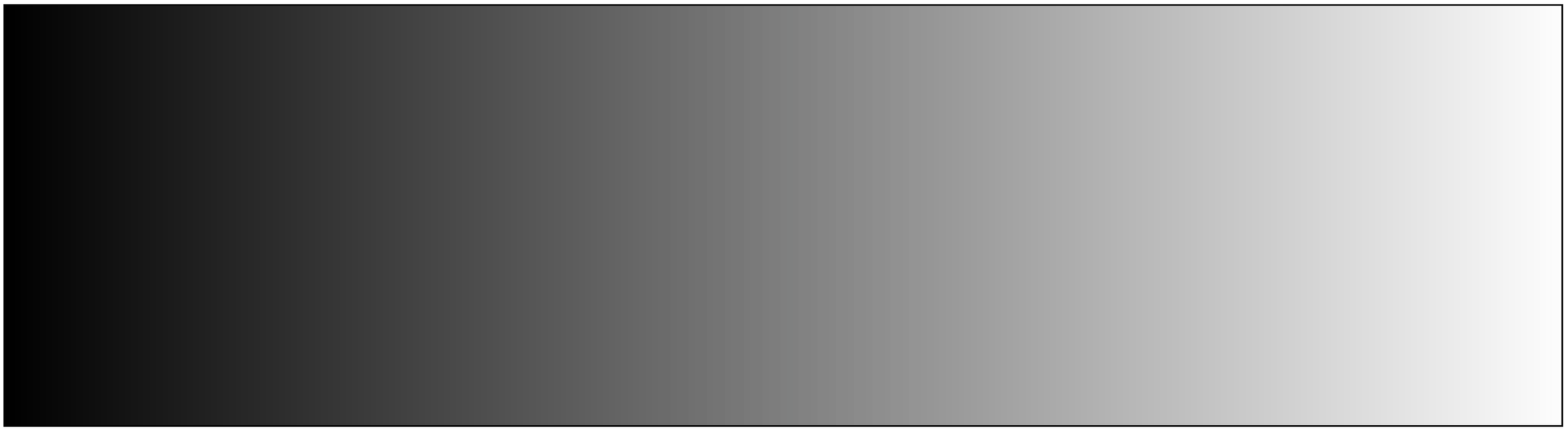
- can Bob retrieve Alice's files?
- can Dropbox read Alice's files?
- did Dropbox delete Alice's files?
- can Dropbox efficiently store files?
- ?





A spectrum of solutions

A spectrum of solutions

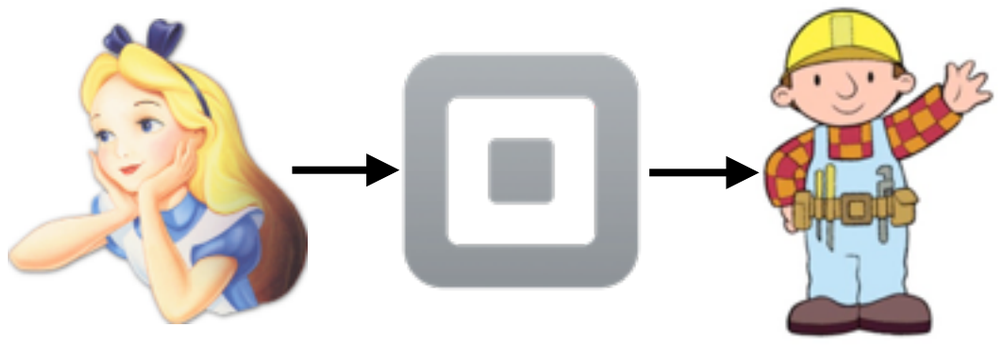


theory practice

A spectrum of solutions

theory

practice

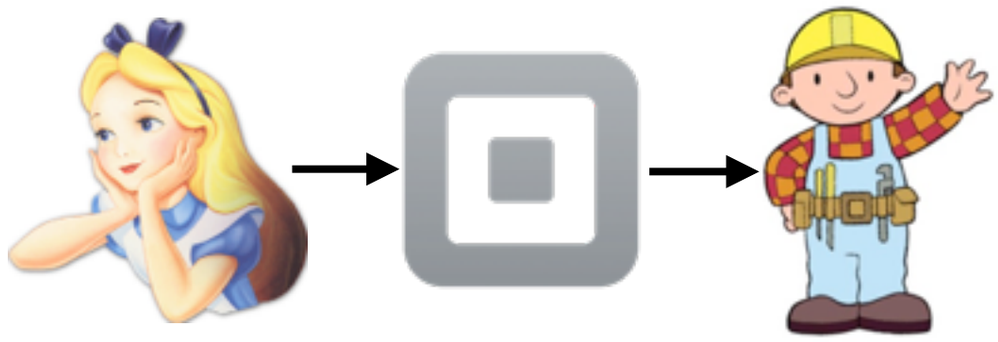


A spectrum of solutions

e-cash [C82]

theory

practice

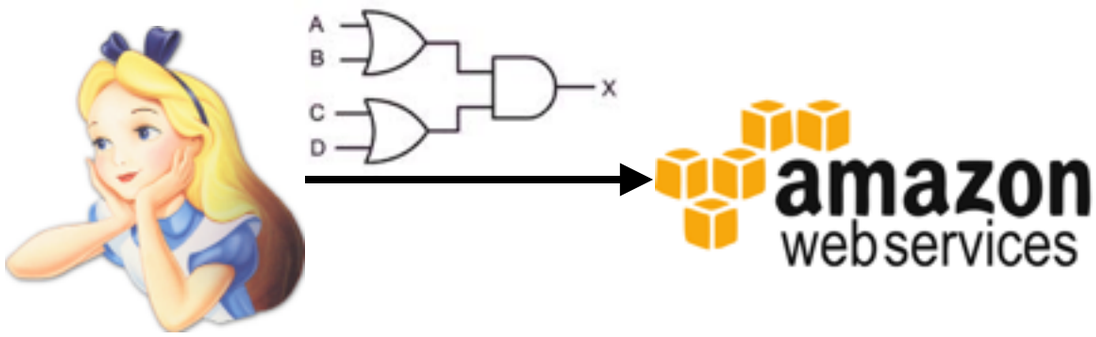


A spectrum of solutions

e-cash [C82]

theory

practice



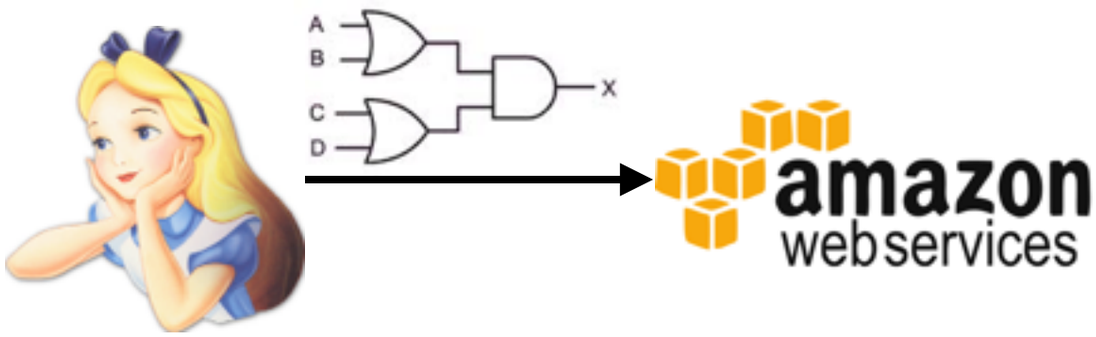
A spectrum of solutions

outsourced comp. [GGP10]

e-cash [C82]

theory

practice



A spectrum of solutions

theory practice

A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

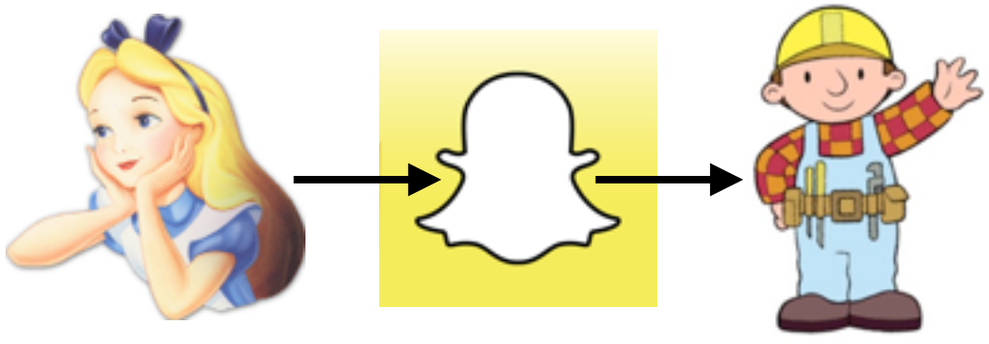
outsourced comp. [GGP10]

ORAM [GO96]

e-cash [C82]

theory

practice



A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

outsourced comp. [GGP10]

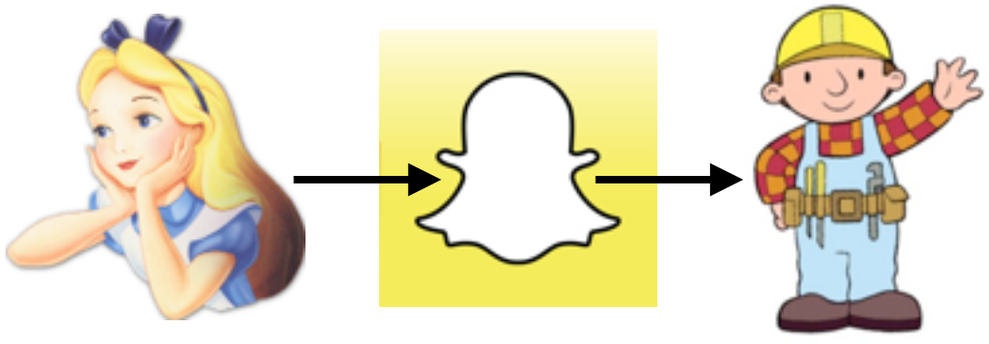
ORAM [GO96]

e-cash [C82]

Wickr

theory

practice



A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

outsourced comp. [GGP10]

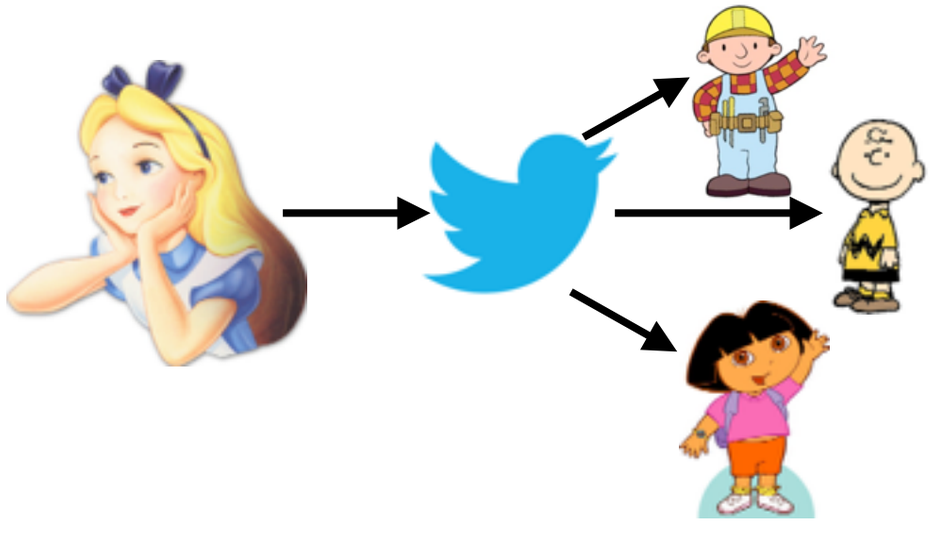
ORAM [GO96]

e-cash [C82]

Wickr

theory

practice



A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

outsourced comp. [GGP10]

ORAM [GO96]

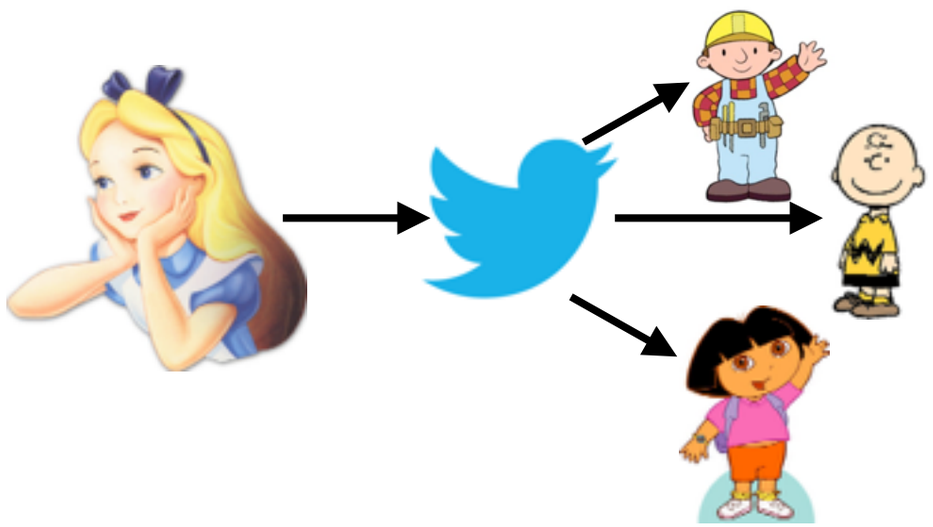
e-cash [C82]

Wickr

diaspora*

theory

practice



A spectrum of solutions

theory practice

A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

outsourced comp. [GGP10]

ORAM [GO96]

e-cash [C82]

Bitcoin

SSL/TLS

TrueCrypt

Wickr

Tor

diaspora*

PGP

theory

practice

A spectrum of solutions

RKA [B93,K93]

MPC [Y82]

FHE [G09]

outsourced comp. [GGP10]

ORAM [GO96]

e-cash [C82]

Bitcoin

SSL/TLS

TrueCrypt

Wickr

Tor

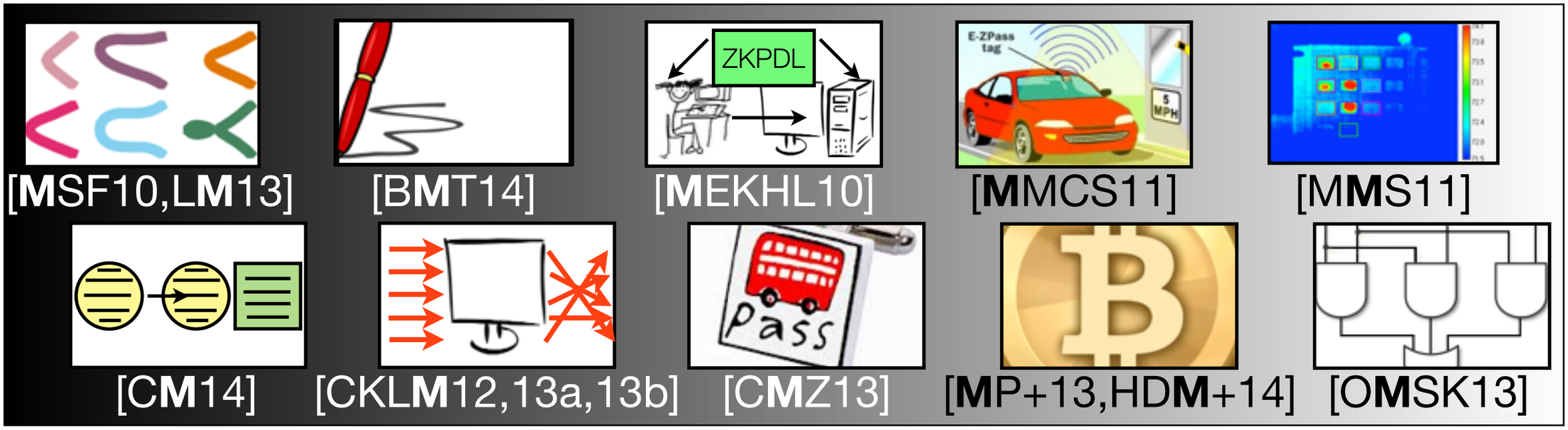
diaspora*

PGP

theory

practice

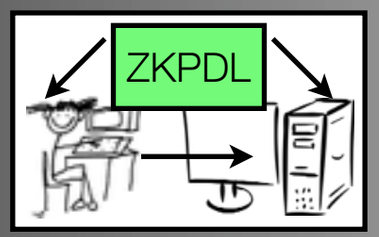
My research


[MSF10,LM13]
[BMT14]
ZKPDL
[MEKHL10]
E-ZPass tag
5 MPH
[MMCS11]
[MMS11]
[CM14]
[CKLM12,13a,13b]
pass
[CMZ13]
[MP+13,HDM+14]
[OMSK13]

My research

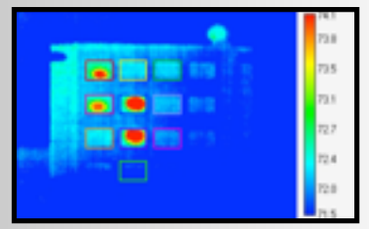

[**M**SF10,L**M**13]

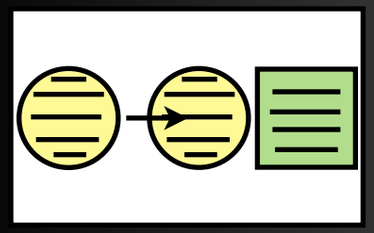

[B**M**T14]

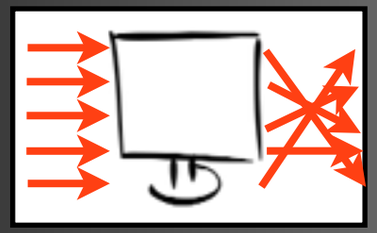


[**M**EKHL10]

[**M**MCS11]

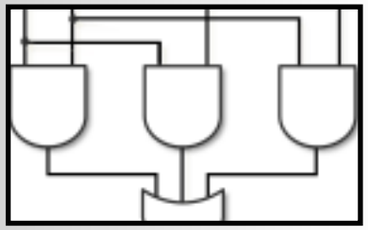

[M**M**S11]


[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]

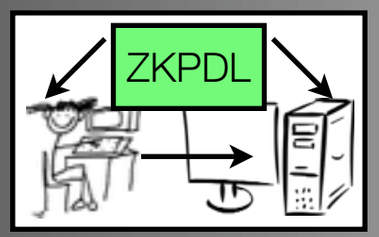

RKA


Bitcoin

My research

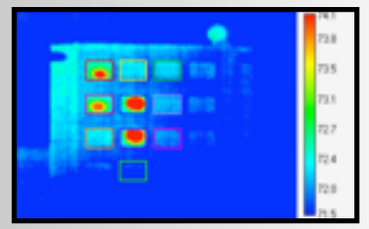

[**M**SF10,L**M**13]

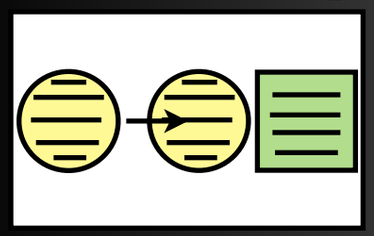

[B**M**T14]

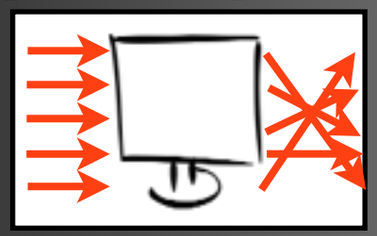


[**M**EKHL10]

[**M**MCS11]

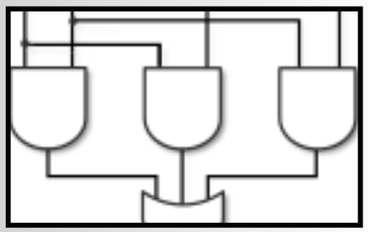

[M**M**S11]

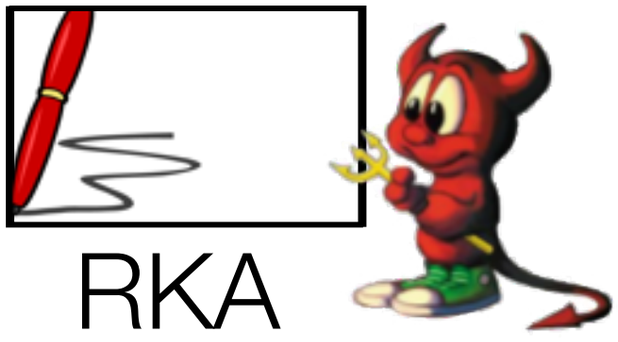

[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]

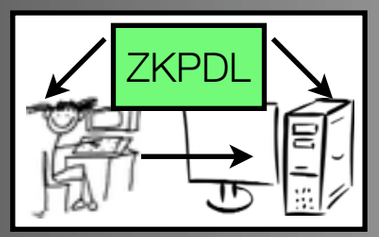

RKA


Bitcoin

My research

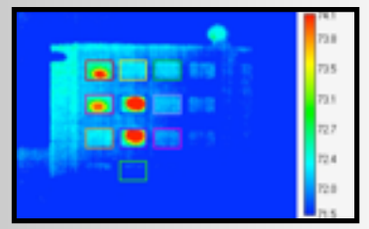

[**M**SF10,L**M**13]

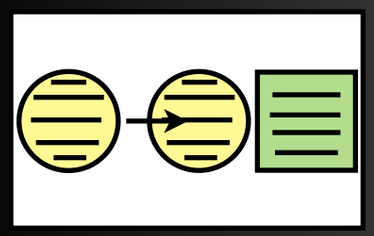

[B**M**T14]

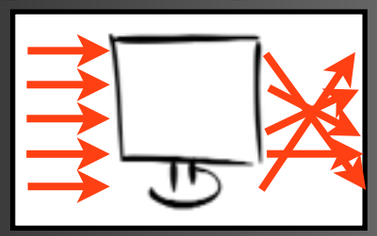


[**M**EKHL10]

[**M**MCS11]

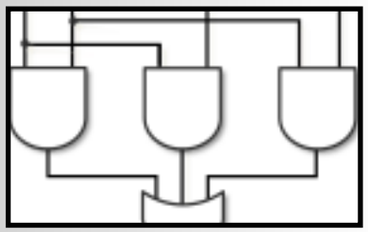

[M**M**S11]

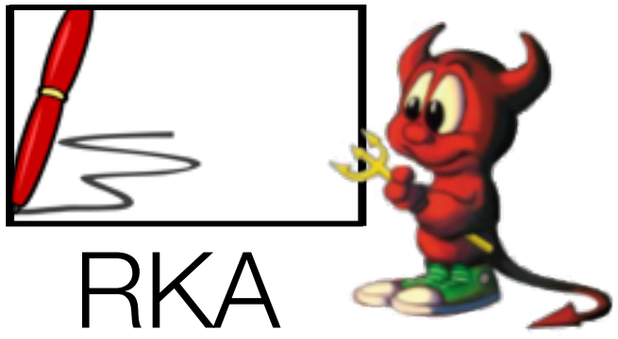

[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]

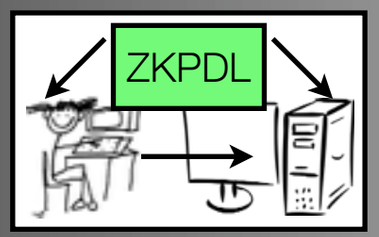

RKA


Bitcoin

My research

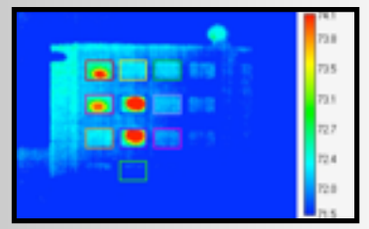

[**M**SF10,L**M**13]

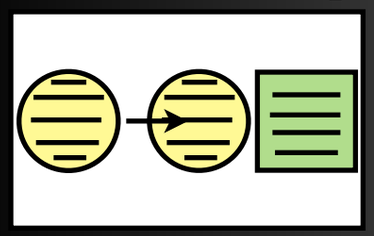

[B**M**T14]

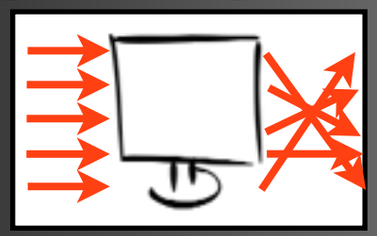


[**M**EKHL10]

[**M**MCS11]

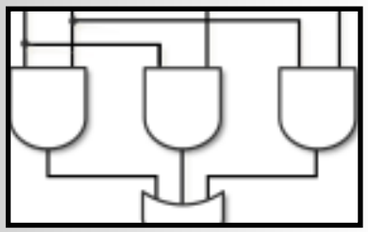

[M**M**S11]


[C**M**14]


[CKL**M**12,13a,13b]

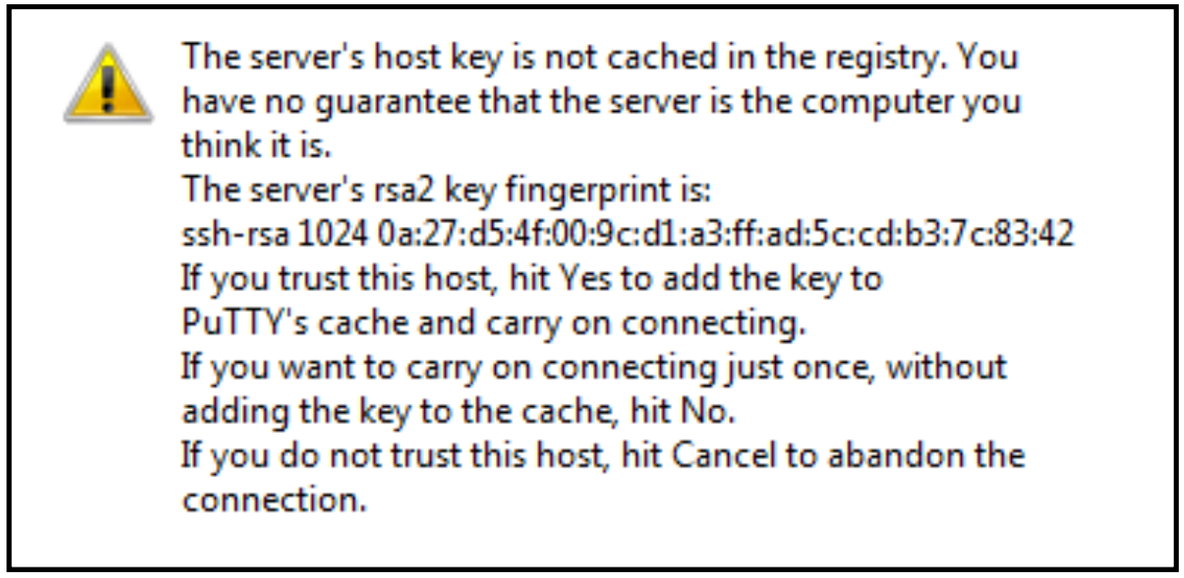
[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]

Digital signatures are everywhere

Digital signatures are everywhere

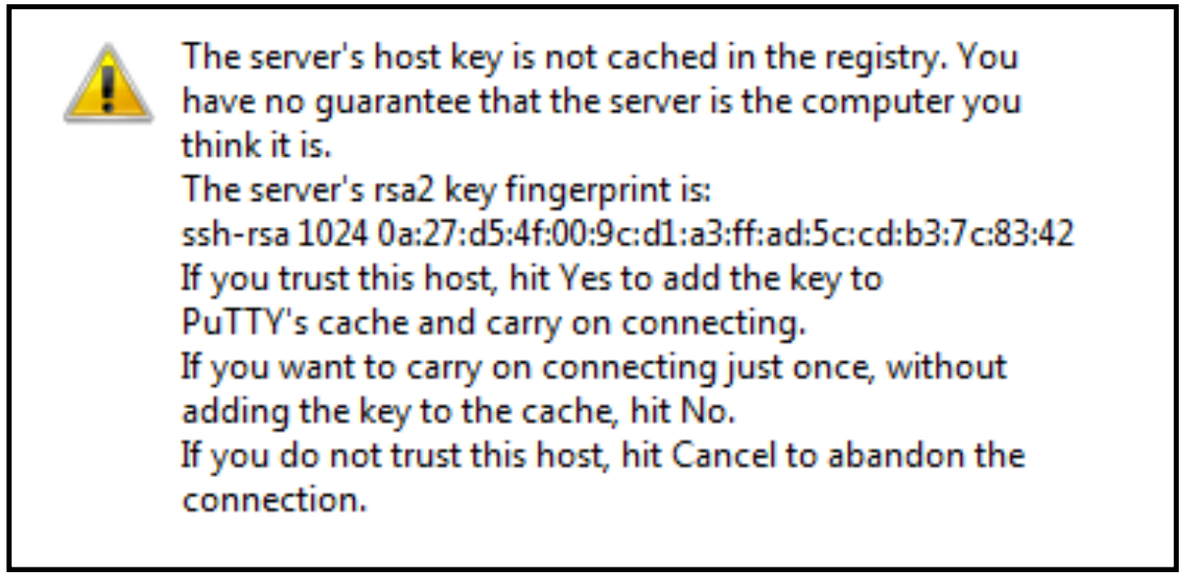


Digital signatures are everywhere

17ijmE4nZstNnevbHLawnhCW9nZ	0.37 BTC
1cTHK2B4D3qg4kj44wxfeQCTE6N	3.11 BTC
1ZhKkzRLLfpNGNHf8qDDH7uRZta	0.20 BTC
1QJ3FK8grGXKEPqVUC1tx6G5Hw	5.16 BTC

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 1024 0a:27:d5:4f:00:9c:d1:a3:ff:ad:5c:cd:b3:7c:83:42
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

Digital signatures are everywhere



Digital signatures are everywhere

Signature Algorithm	SHA-1 with RSA Encryption (1 2 840 113549 1 1 5)
Parameters	none
Not Valid Before	Wednesday, July 20, 2011 5:00:00 PM PT
Not Valid After	Thursday, July 18, 2013 4:59:59 PM PT
Public Key Info	
Algorithm	RSA Encryption (1 2 840 113549 1 1 1)
Parameters	none
Public Key	128 bytes : D5 14 57 A0 96 40 9F 84 08 C6 66 8D EE EC E3 03 B2 66 85 AC 5D BB 1C EF 15 93 FD 1F 20 A7 10 49 24 5B 39 D2 60 C8 9A DC C0 CE 40 34 E6 59 95 B6 52 50 FE 08 25 45 57 73 5F 3A ED 5F DF B6 5C 9D 8A 9C 62 FD 0F 61 BE DC F6 87 1D 80 9D 7F 7C 17 13 77 64 3C 47 F3 87 24 1F 60 61 E0 81 11 46 E4 DC 50 5C 39 53 E6 68 3D 86 3C 55 87 C8 BE FC 87 13 D9 5A AA 5D CC 3F 07 C1 74 CD C2 5E 27 16 11
Exponent	65537
Key Size	1024 bits
Key Usage	Encrypt, Verify, Derive
Signature	128 bytes : 84 FB EF B0 38 C7 B4 E6 BB 74 41 34 6C 52 6F 0E F7 7B 67 D4 D7 AD A7 EF 03 36 EF C0 40 DE 80 D3 0A A6 3C C9 C3 9E 31 52 07 EA 0F 44 63 EA 51 C2 D0 CE 9B A1 6D 99 F8 29 39 2C 40 98 67 12 8C A7 BF 43 30 D2 02 63 B3 F8 0C 36 85 50 8F 29 C2 87 CD E6 F0 EA 57 F2 A4 81 0E 94 B0 D8 46 67 2C 20 AE B6 89 64 A6 B0 DA 8E CF D4 09 7F 57 DC 9C 32 05 06 62 B2 08 14 FE 8B EA 11 13 BC F7 3B 7E F5

OK

```
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.11 (GNU/Linux)
Comment: Using GnuPG with Thunderbird - http://www.enigmail.net/

iQEcBAEBAgAGBQJSbpK5AAoJEKAl9HCTr8nwh8sIAIlViE5OmpHbzpcatKMdUL9q
0ehAanFyH9WEVxHuv/DAW9Mq35hsYRJL+2wDBUbyeCtCIhjgfW0jmgsYc0MevQva
w0MUOleYwBOVkDM6l6E5KN+fRC5qqjeLZvFO2yVrEdltCZeRvNzKm1MREbKnSBPd
vkMAW/G/U2dA9GmQjm3lw6Av8Sakh8FC+nGGOwtKHl3TpWK41RdqR503M7tBD1Ov
+d9zRchtD5Rs2X43CSTXq5g02zIBcCqarfLKGvPLHOc+4YW+9+9H9V/LT7vTDIDe
WWvIrYWM21p8eShdcoukW3aar3ovNqQ6B/8xzLeblAcruvIl3ejE2xAR20Btm50=
=jT/5
-----END PGP SIGNATURE-----
```

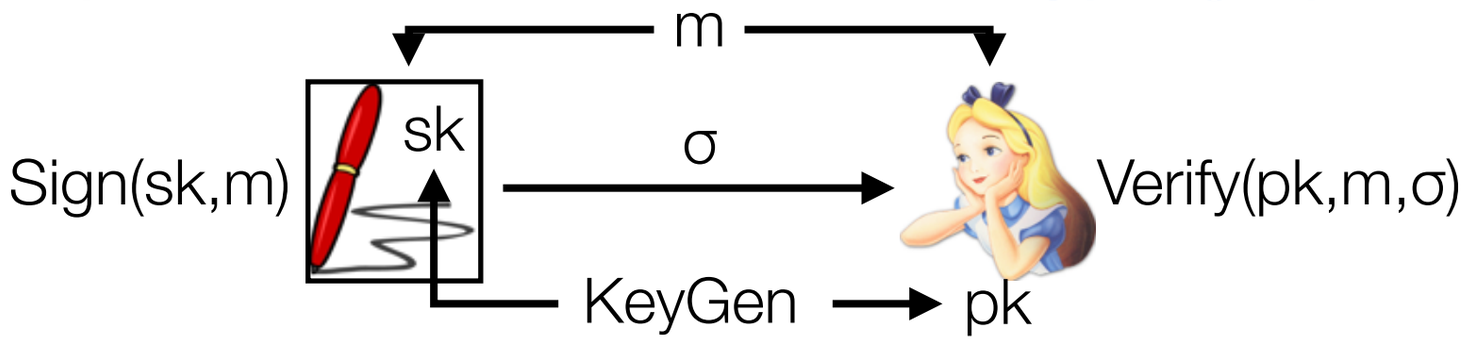
17ijmE4nZstNnevbHLawnhCW9nZ	0.37 BTC
1cTHK2B4D3qg4kj44wxfeQCTE6N	3.11 BTC
1ZhKkzRLLfpNGNHf8qDDH7uRZta	0.20 BTC
1QJ3FK8grGXKEPqVUC1tx6G5Hw	5.16 BTC

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 1024 0a:27:d5:4f:00:9c:d1:a3:ff:ad:5c:cd:b3:7c:83:42
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

Practical security of digital signatures

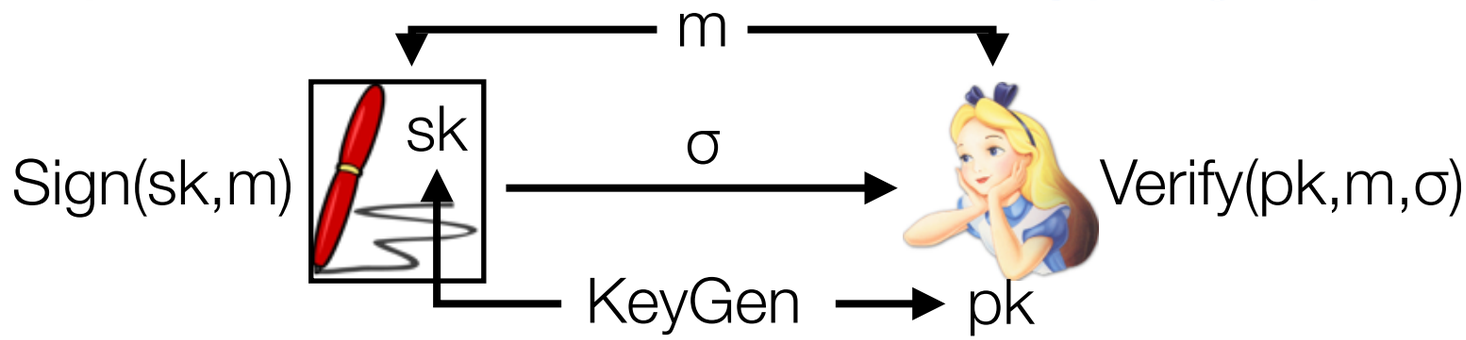
Practical security of digital signatures

Signatures are proved secure in standard cryptographic models



Practical security of digital signatures

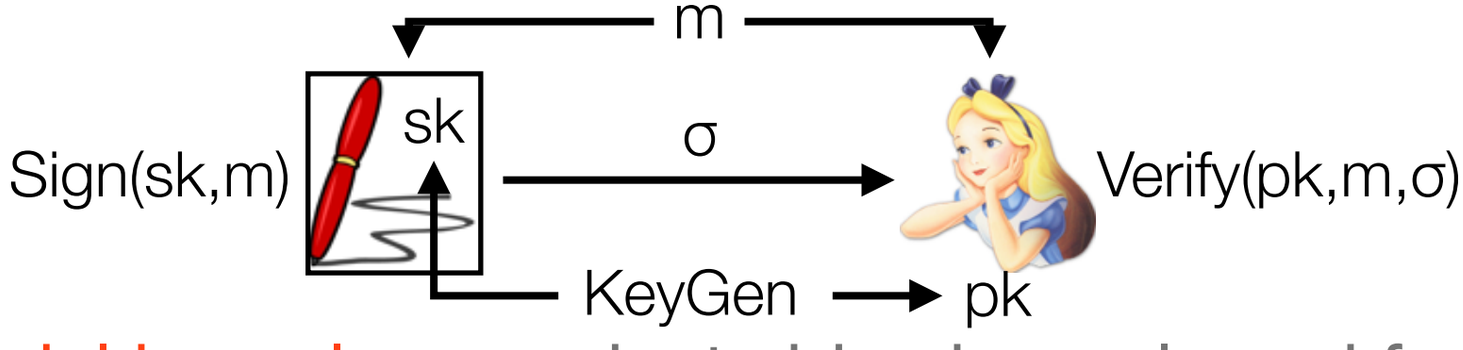
Signatures are proved secure in standard cryptographic models



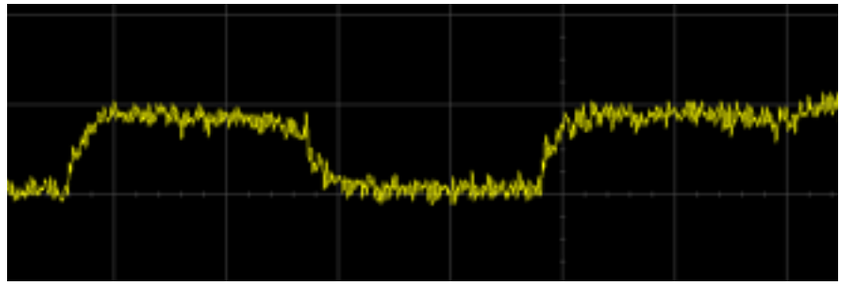
Standard model is useless against side channels and fault injection

Practical security of digital signatures

Signatures are proved secure in standard cryptographic models

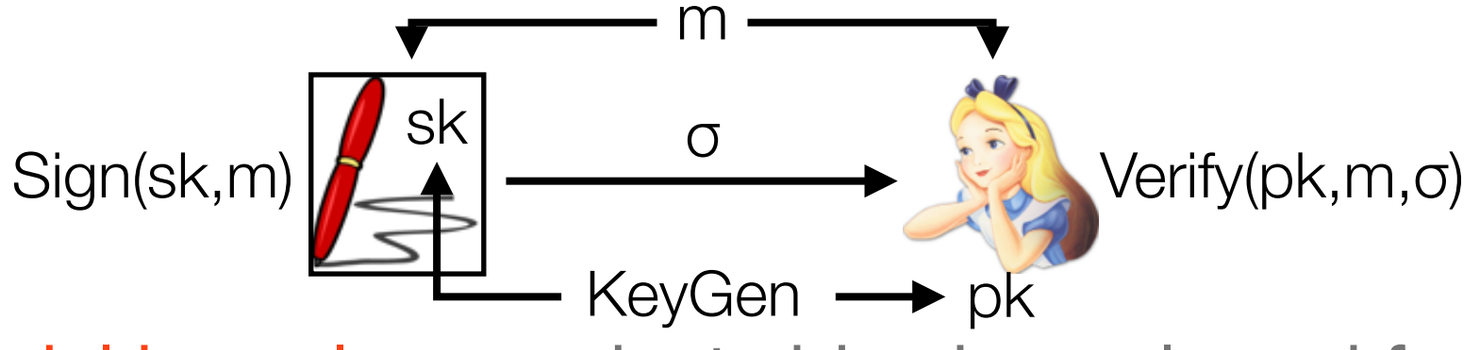


Standard model is useless against side channels and fault injection

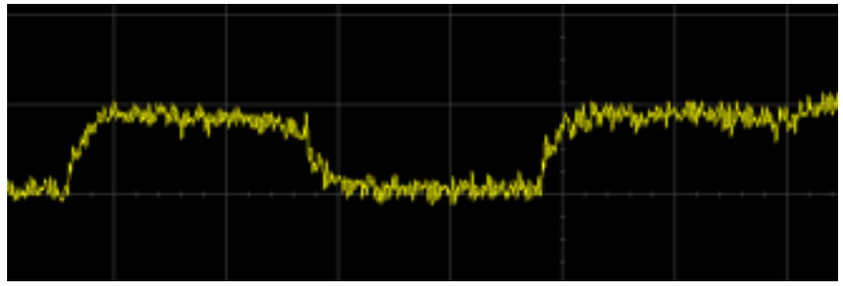


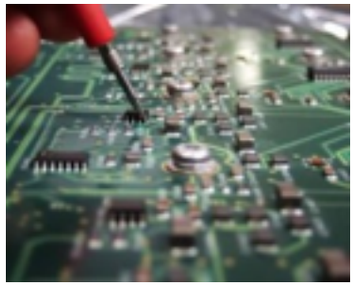
Practical security of digital signatures

Signatures are proved secure in standard cryptographic models



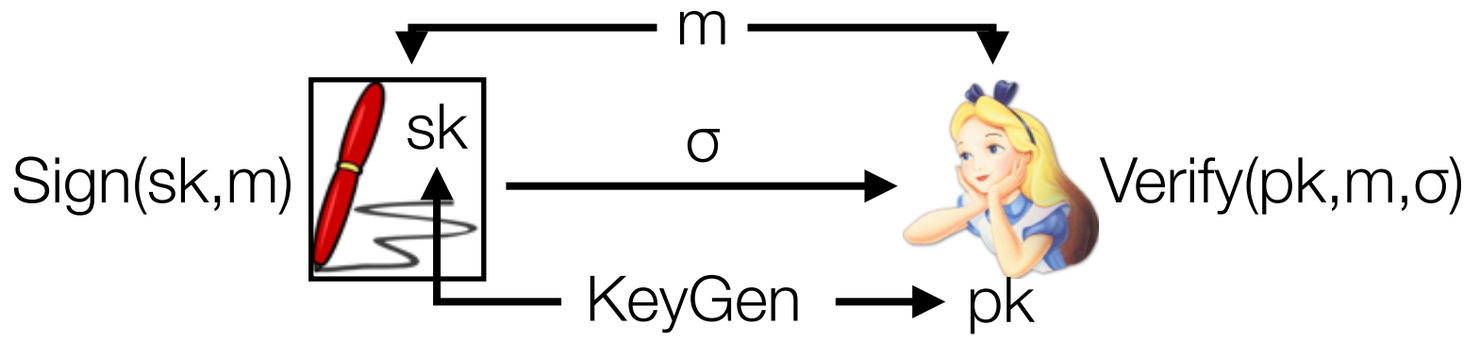
Standard model is useless against side channels and fault injection



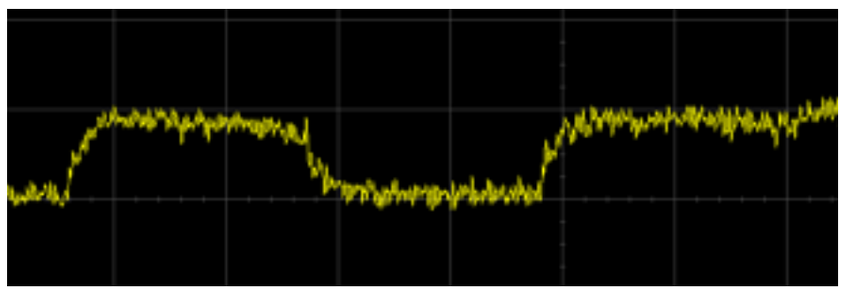


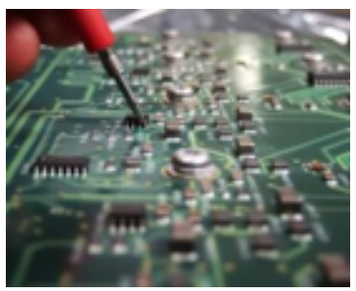
Practical security of digital signatures

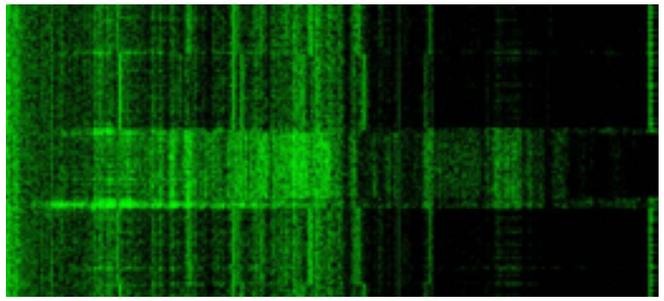
Signatures are proved secure in standard cryptographic models



Standard model is useless against side channels and fault injection

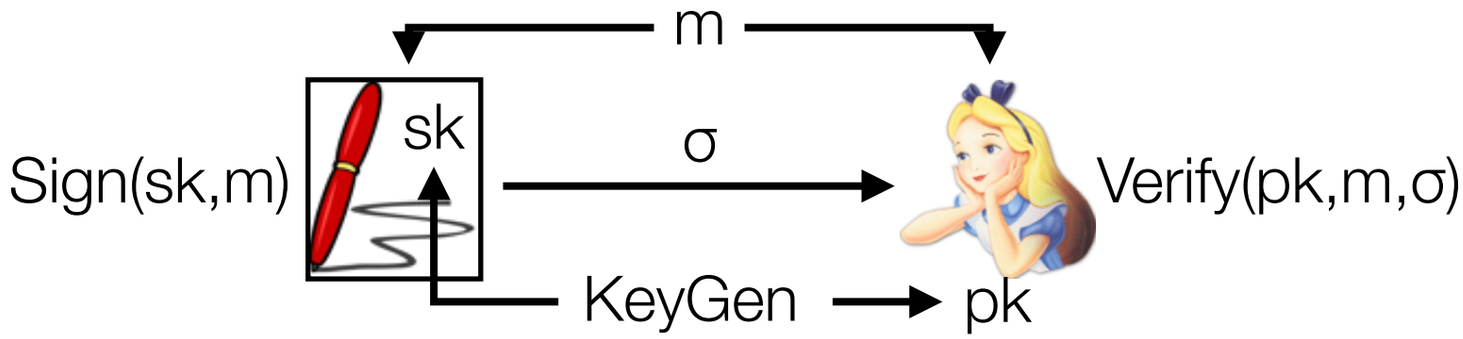




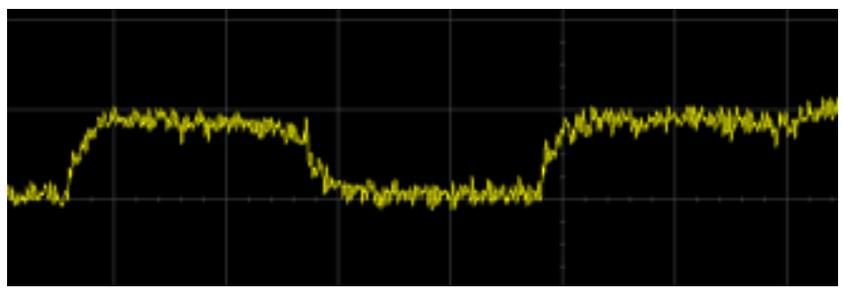
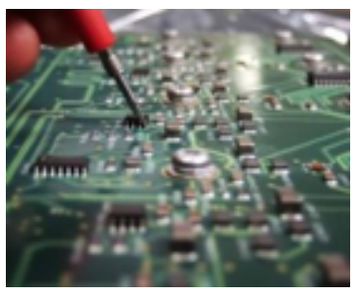
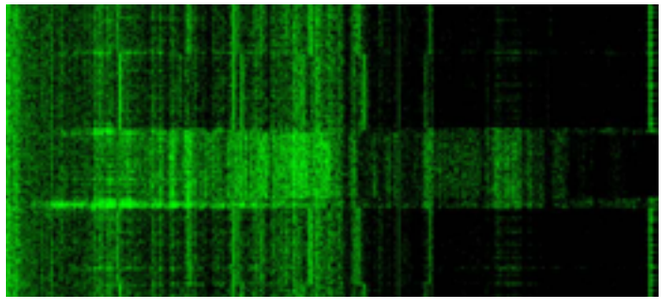


Practical security of digital signatures

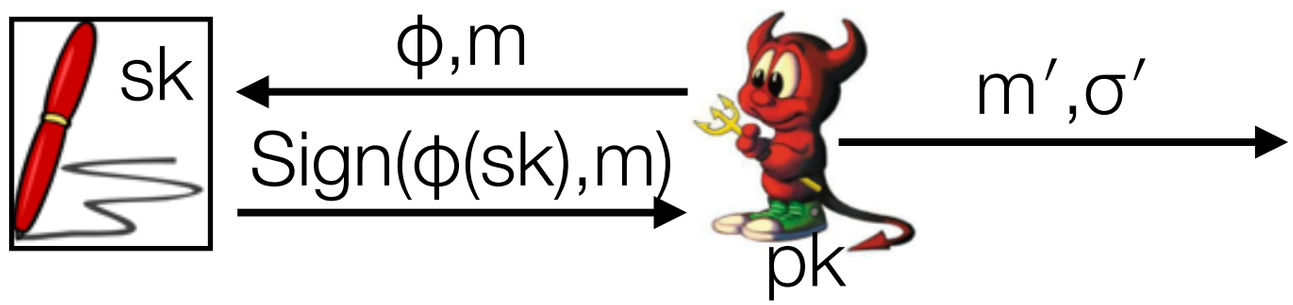
Signatures are proved secure in standard cryptographic models



Standard model is useless against side channels and fault injection

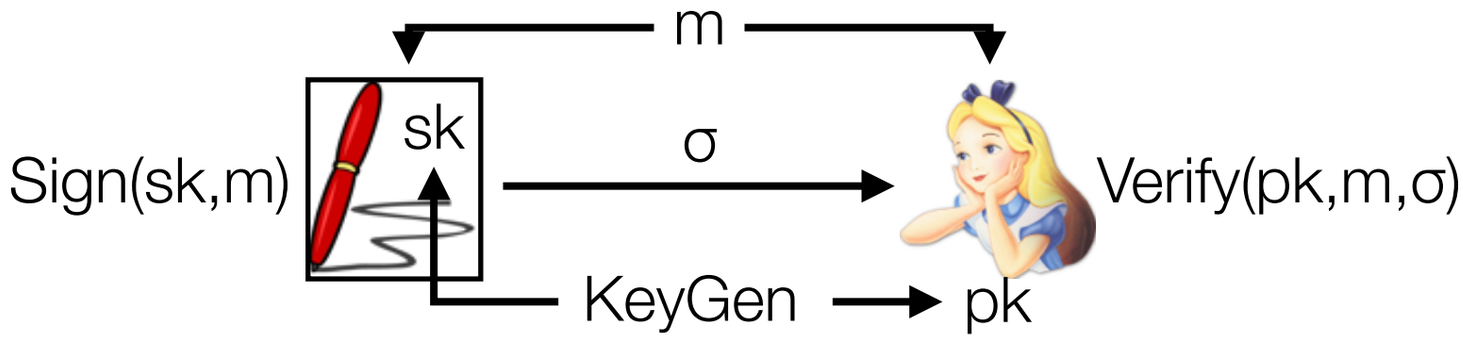




RKA (related key attack) security considers these attacks

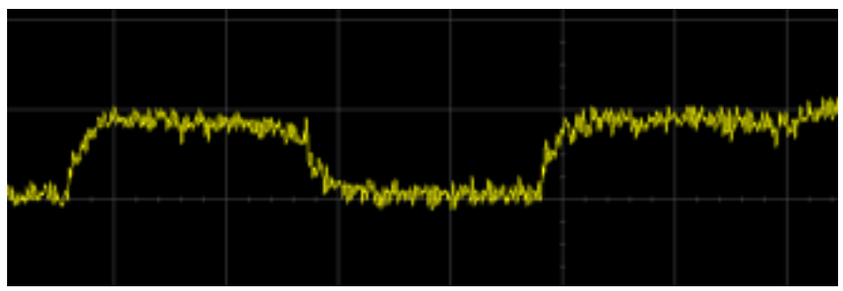
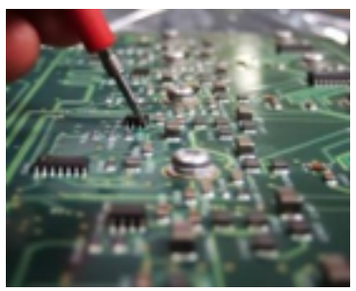
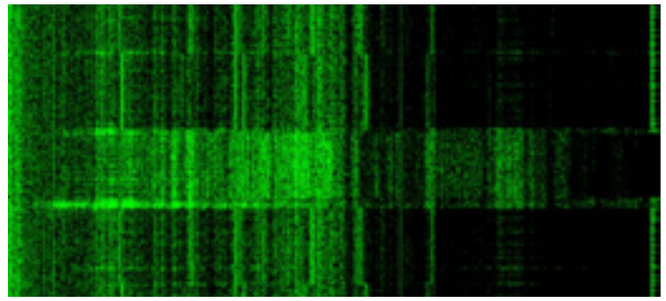


Practical security of digital signatures

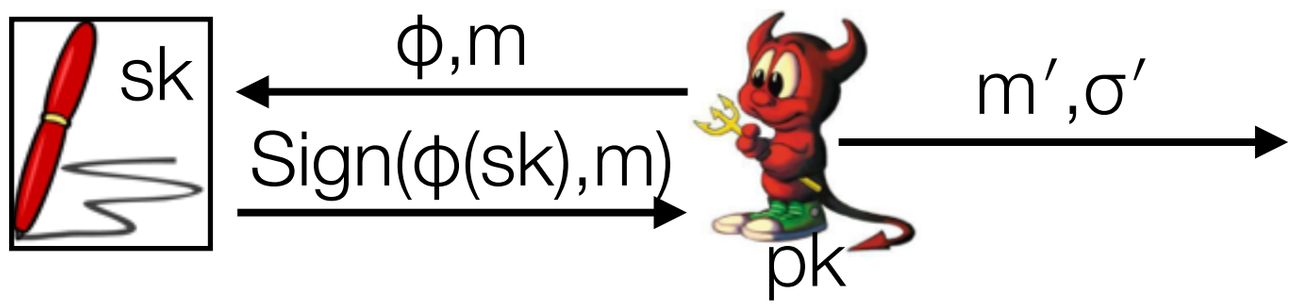
Signatures are proved secure in standard cryptographic models



Standard model is useless against side channels and fault injection

RKA (related key attack) security considers these attacks



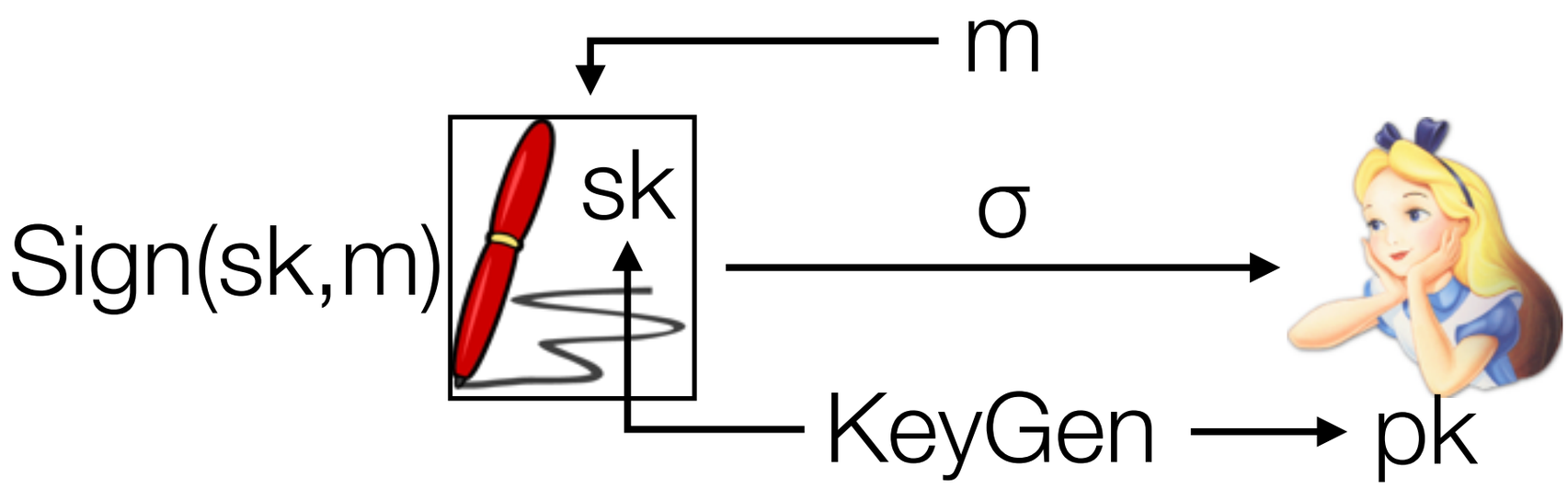
Our research can help create better RKA schemes

Standard definitions for signatures [GMR88]

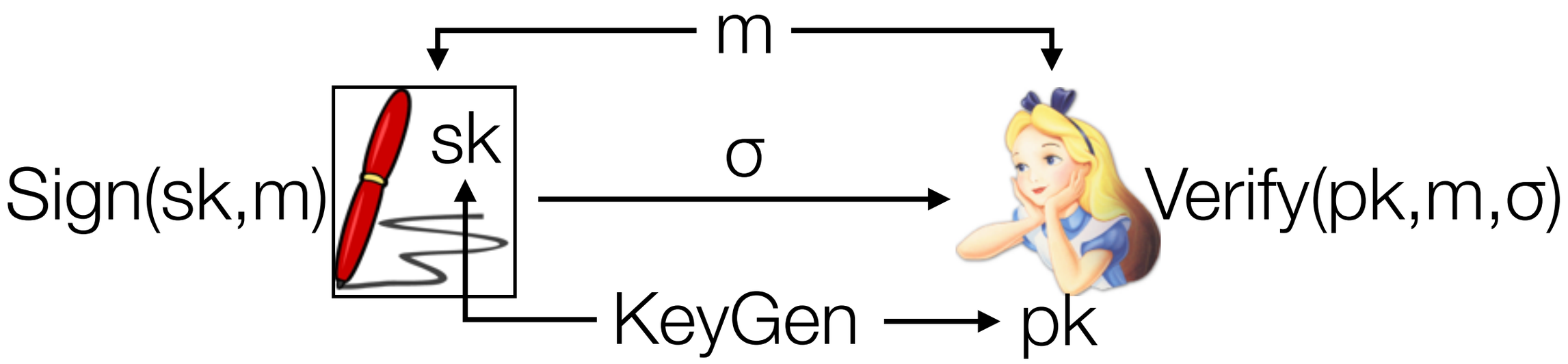
Standard definitions for signatures [GMR88]

Standard definitions for signatures [GMR88]

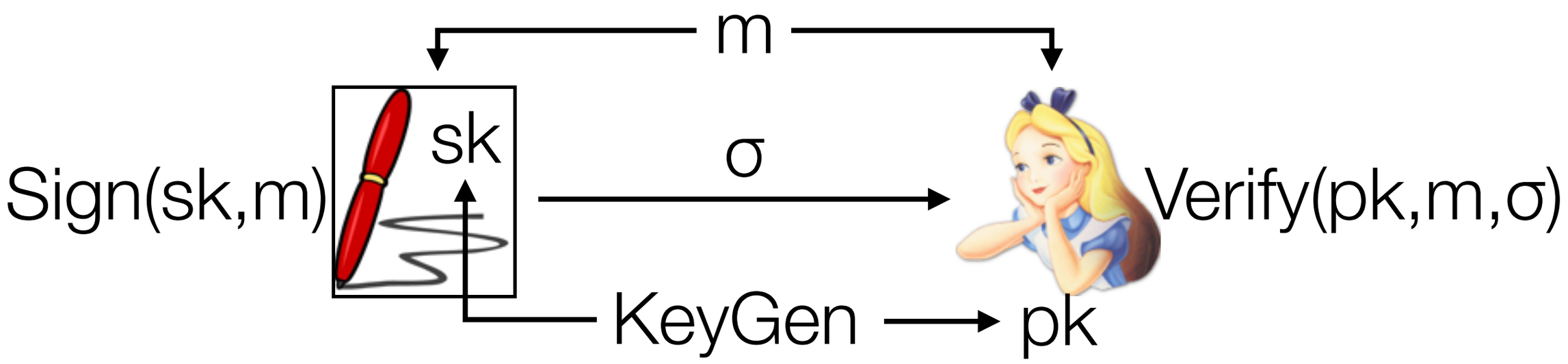
Standard definitions for signatures [GMR88]



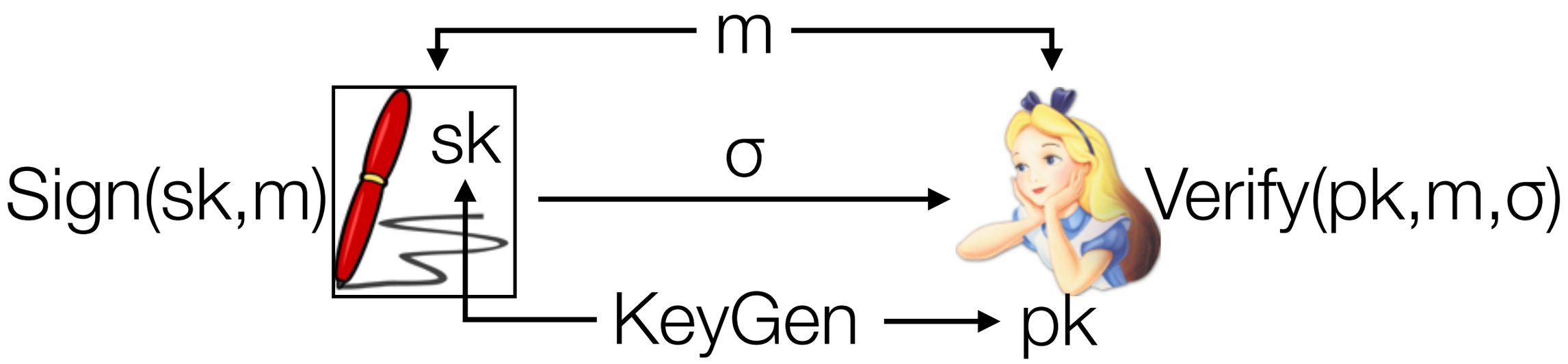
Standard definitions for signatures [GMR88]

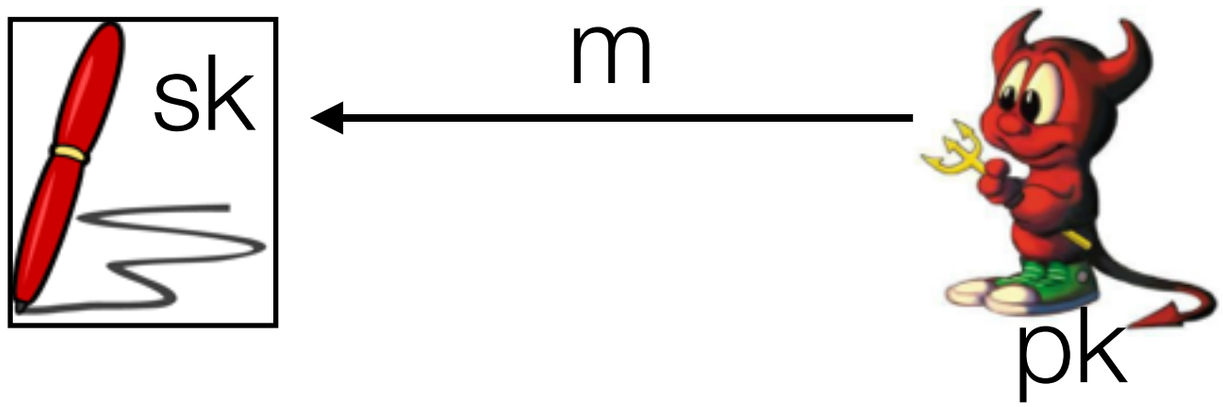


Standard definitions for signatures [GMR88]

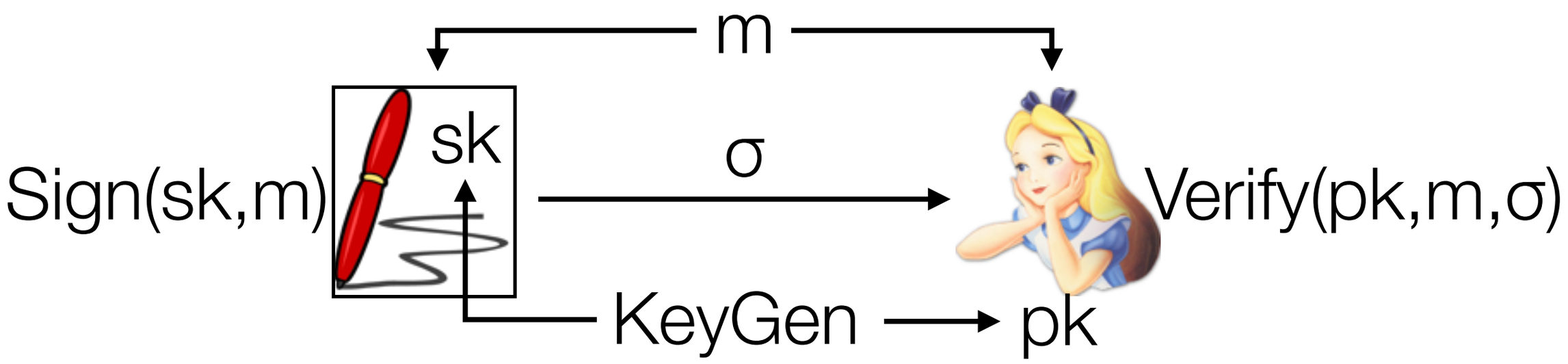


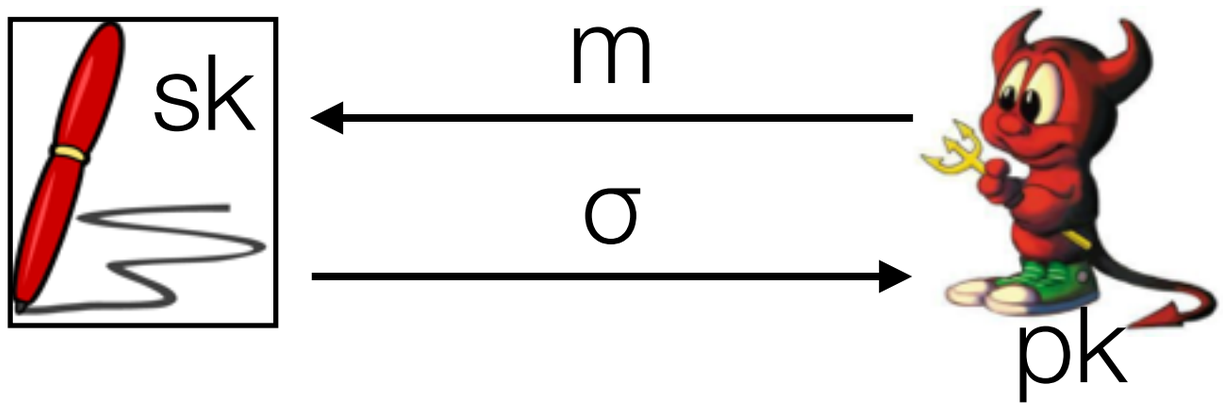
Standard definitions for signatures [GMR88]



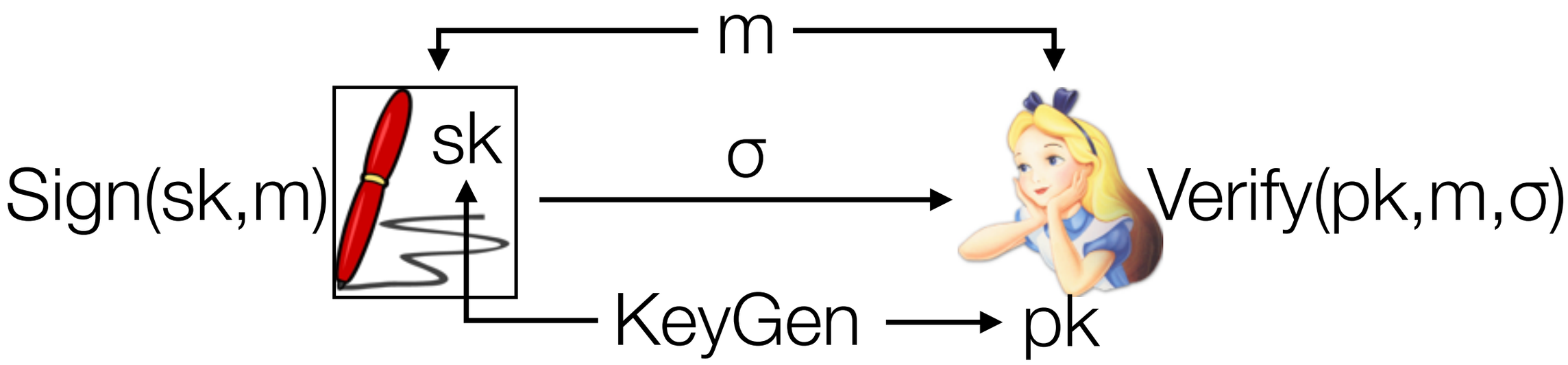


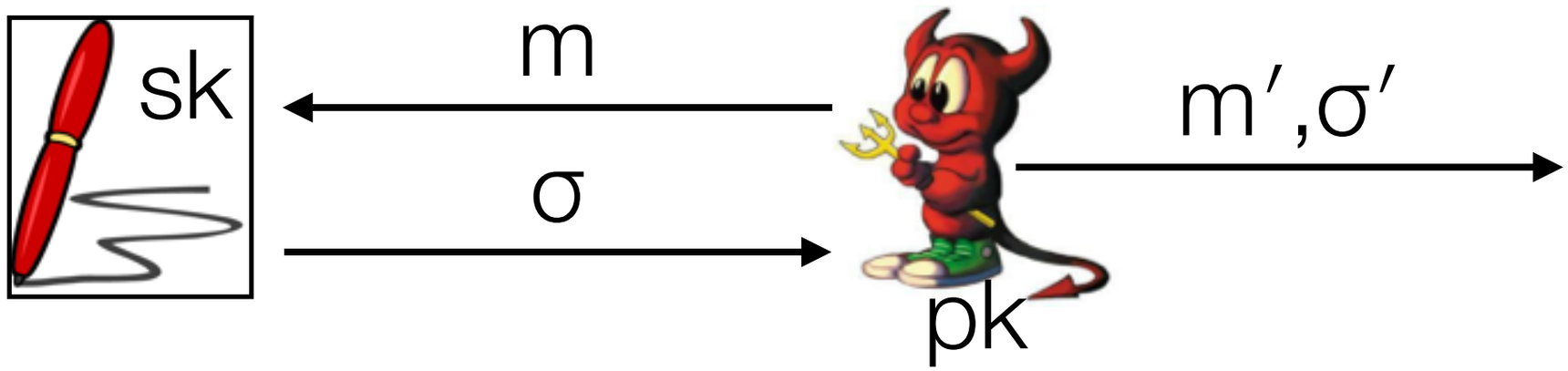
Standard definitions for signatures [GMR88]





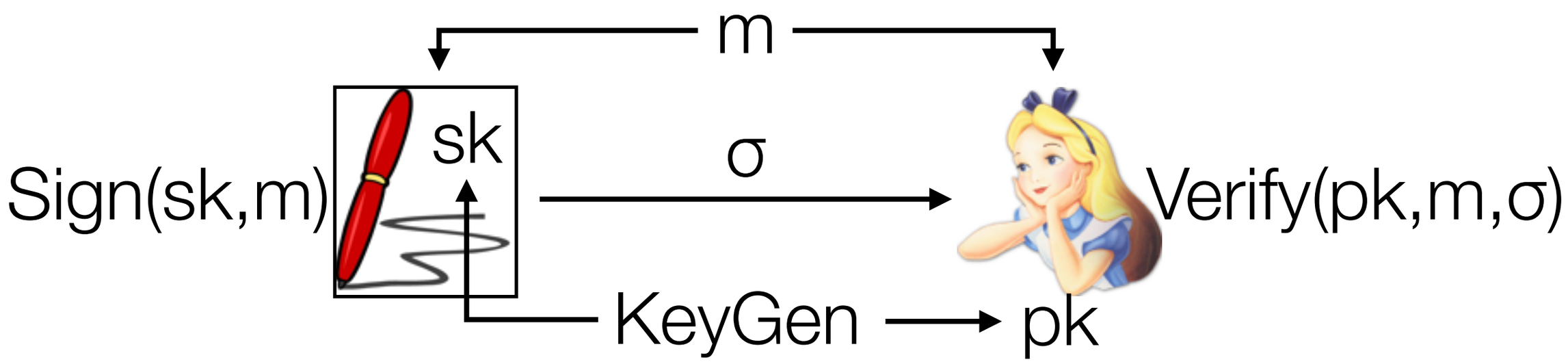
Standard definitions for signatures [GMR88]

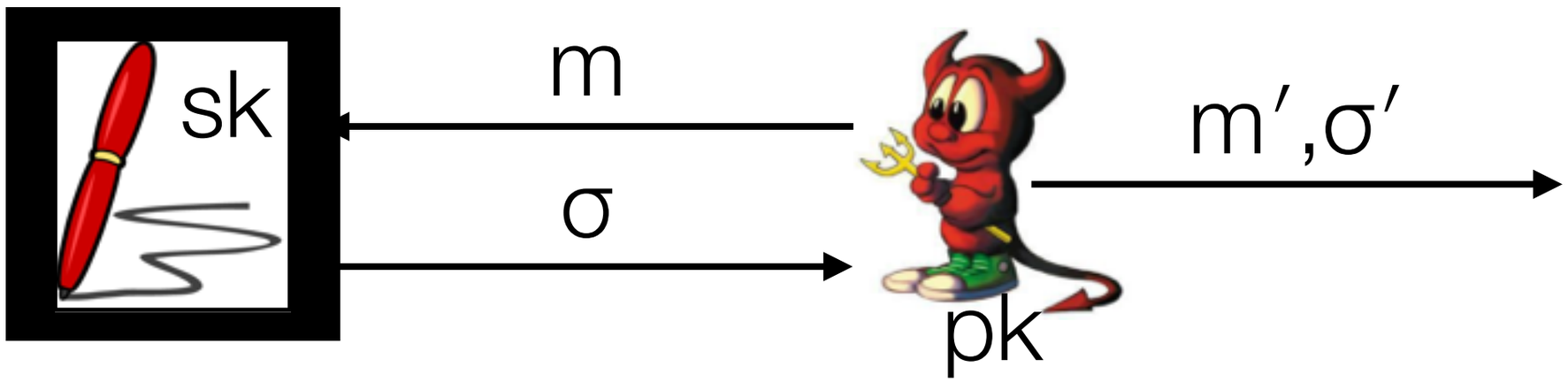




A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't query m′ to oracle

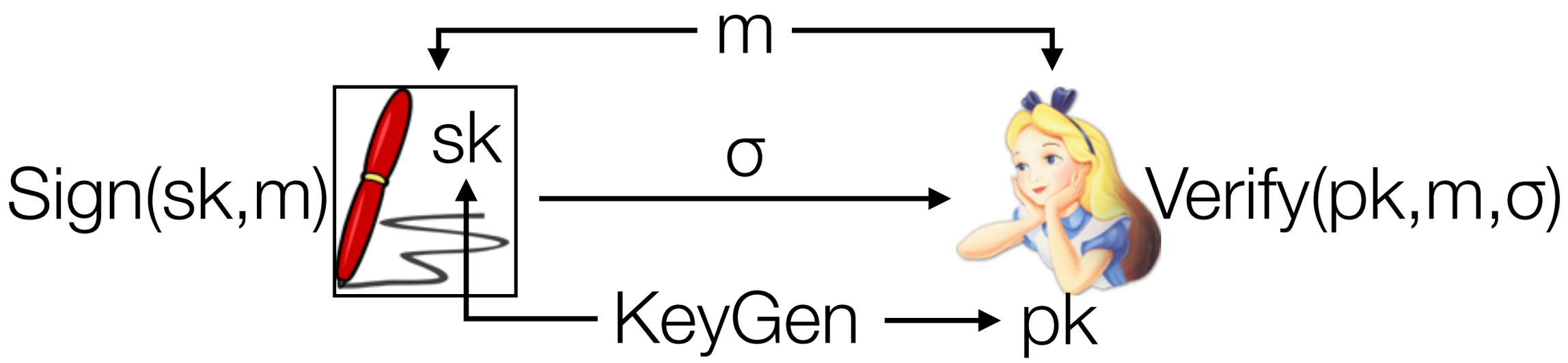
Standard definitions for signatures [GMR88]

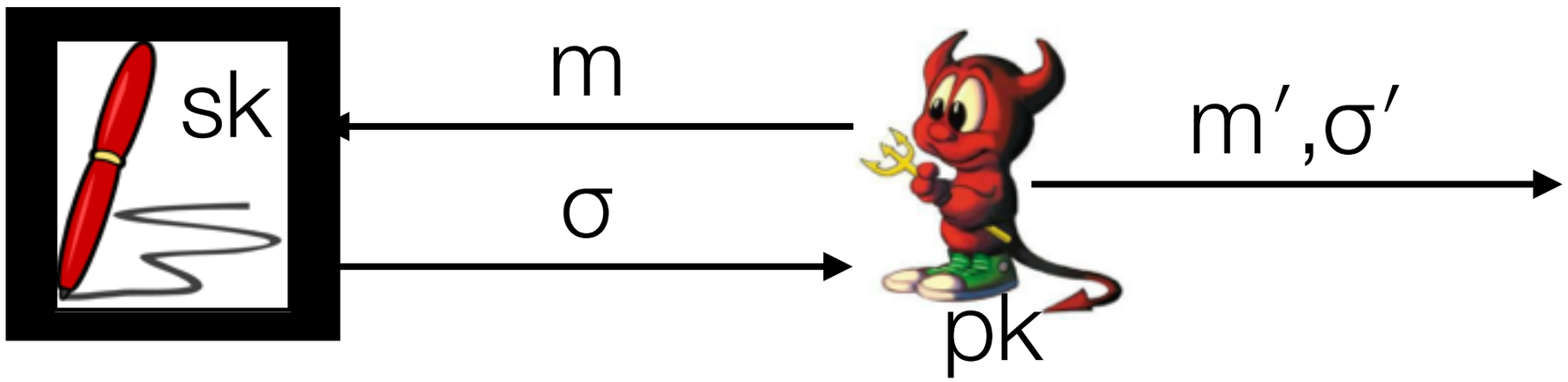




A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't query m′ to oracle

Standard definitions for signatures [GMR88]





A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't query m′ to oracle

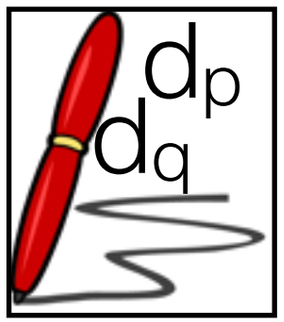
Problem: This assumption is violated by tampering [AK96,…], side channels [W91,KJJ99,…] and fault injection [BS97,BdML97,…]

Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)

Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)

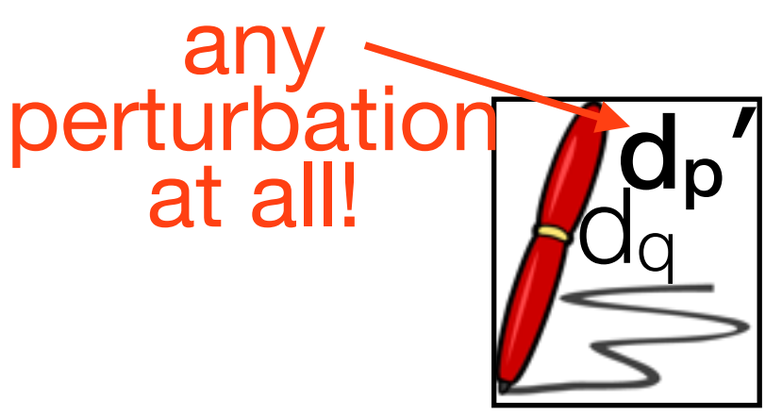


$\sigma = H(m)^d \bmod N$

N,e

Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)

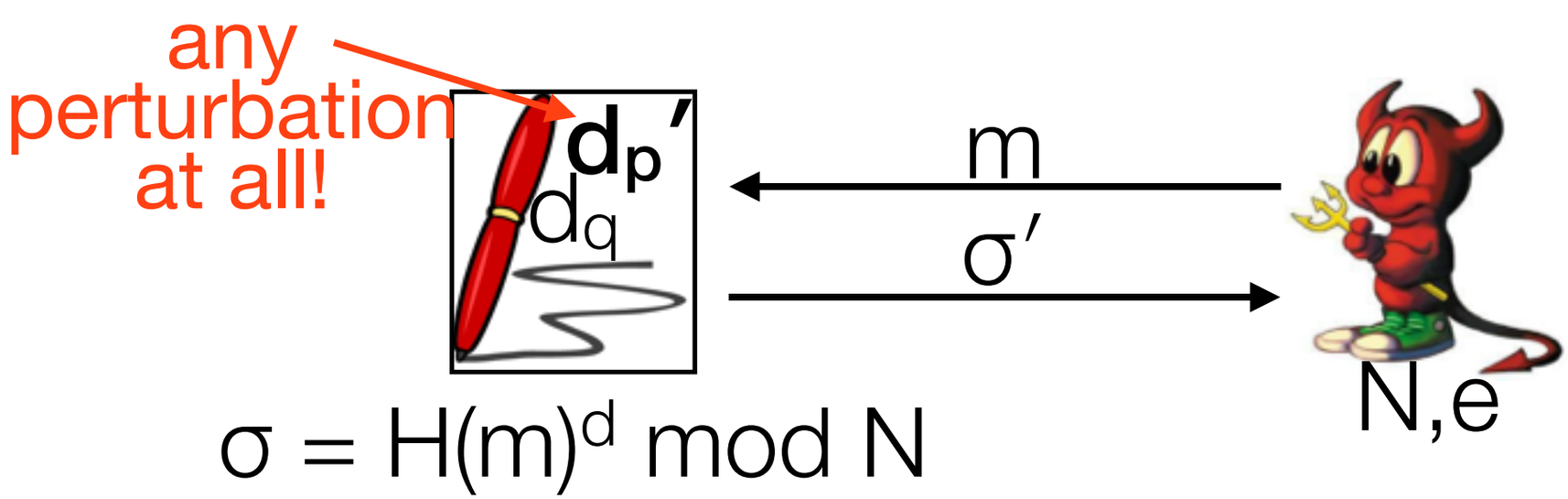


$\sigma = H(m)^d \bmod N$

N,e

Related key attacks (RKA)

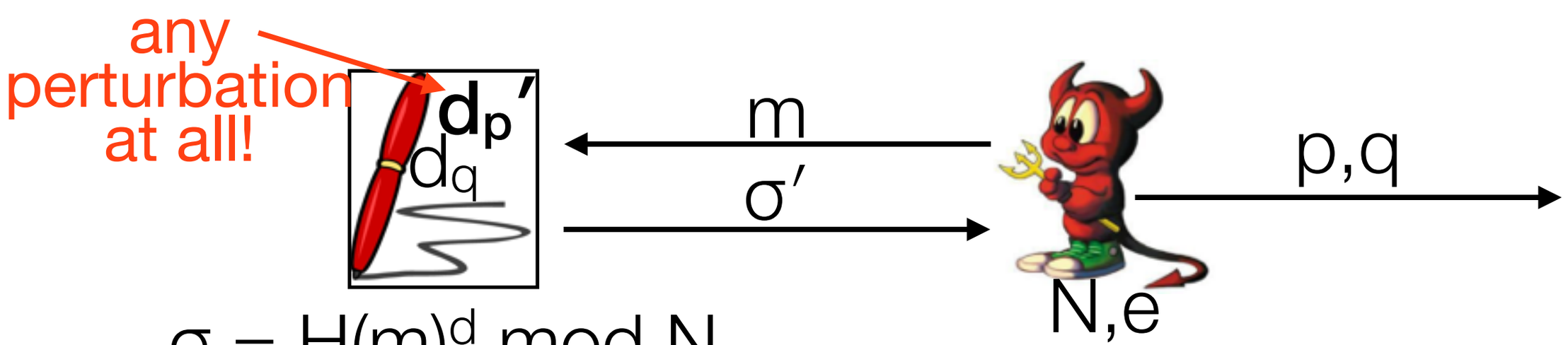
Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)



$$\sigma = H(m)^d \bmod N$$

Related key attacks (RKA)

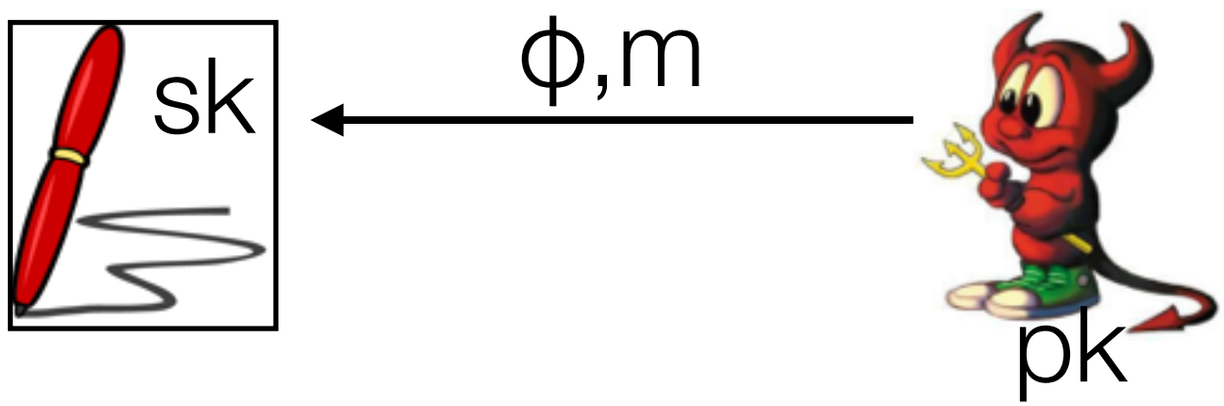
Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)



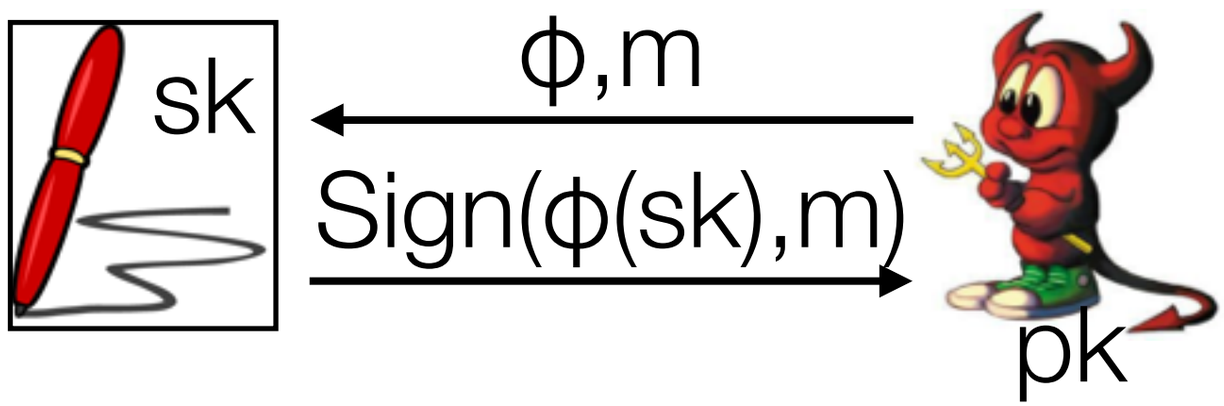
$$\sigma = H(m)^d \bmod N$$

ϕ-RKA-secure signatures [BCM11]

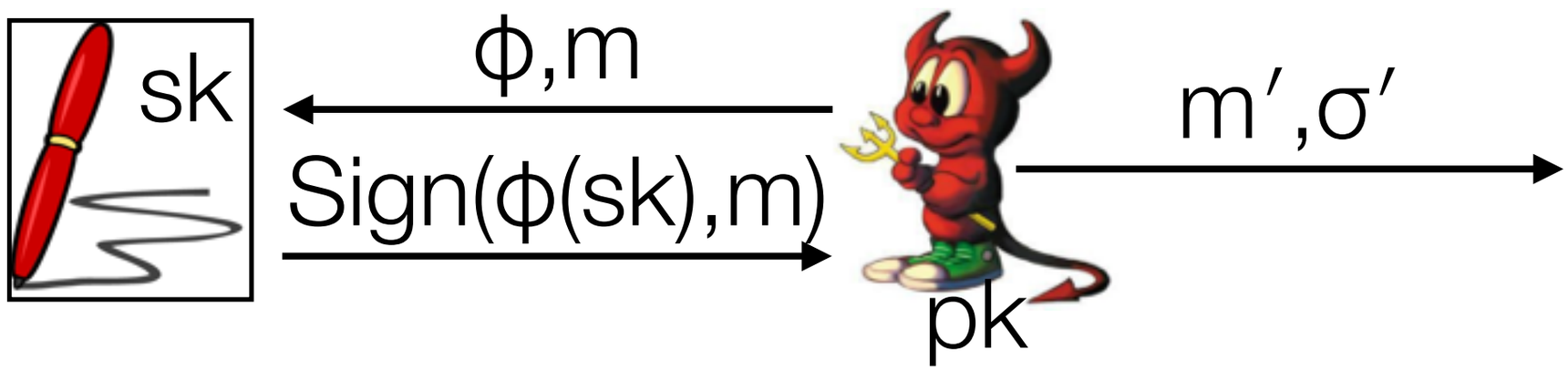
ϕ-RKA-secure signatures [BCM11]



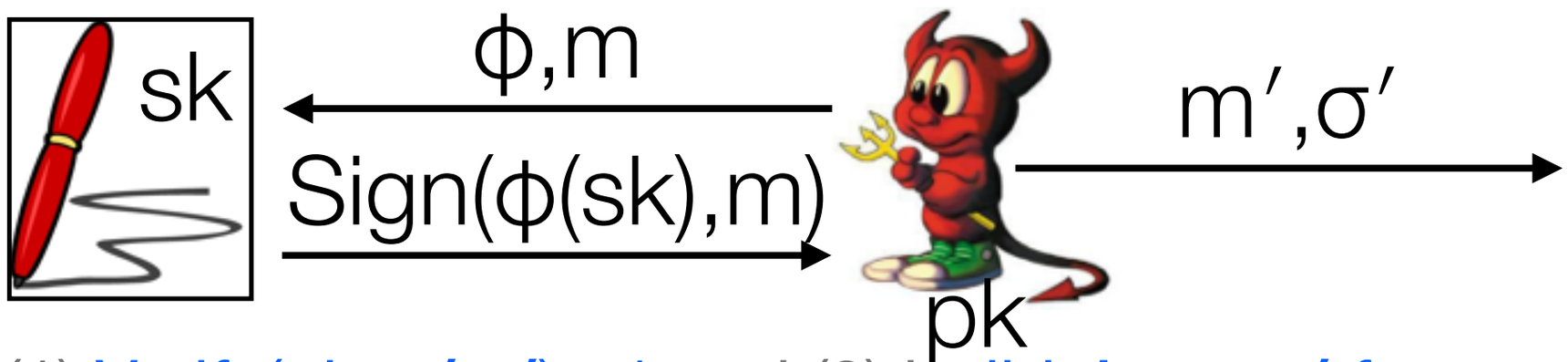
ϕ-RKA-secure signatures [BCM11]



ϕ-RKA-secure signatures [BCM11]

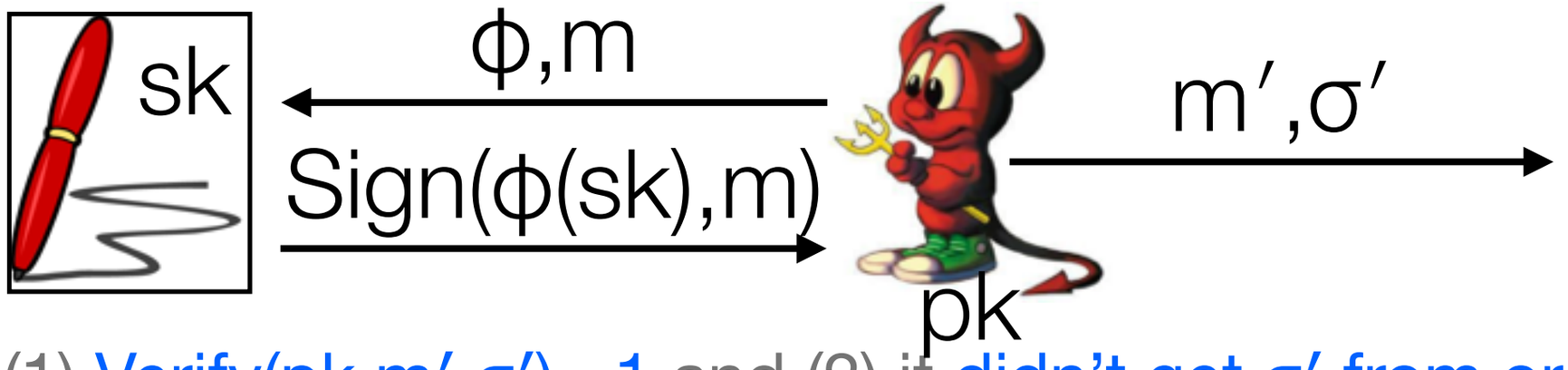


ϕ-RKA-secure signatures [BCM11]



A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)

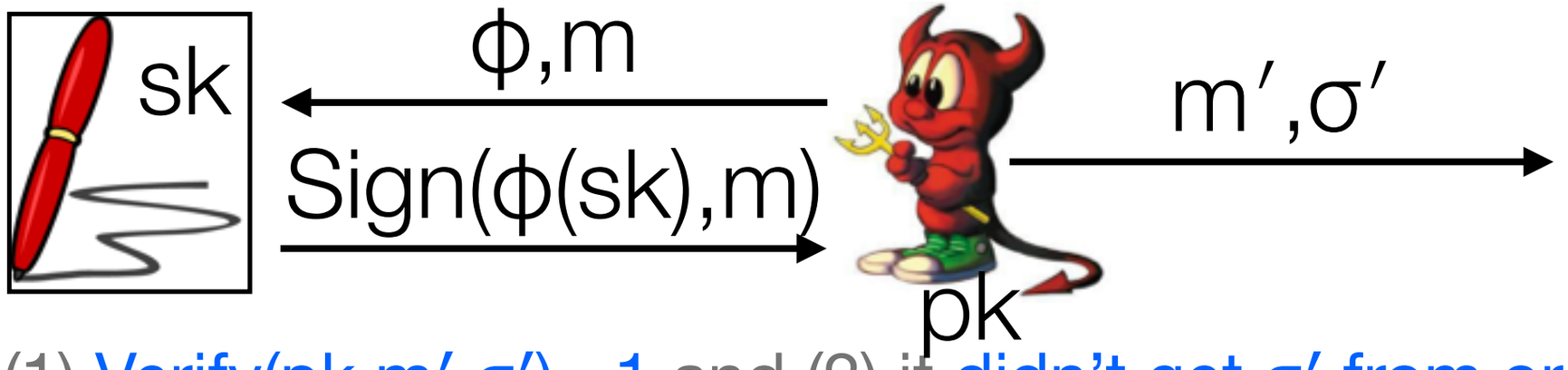
ϕ-RKA-secure signatures [BCM11]



A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)

Φ={id} gives standard unforgeability

ϕ-RKA-secure signatures [BCM11]

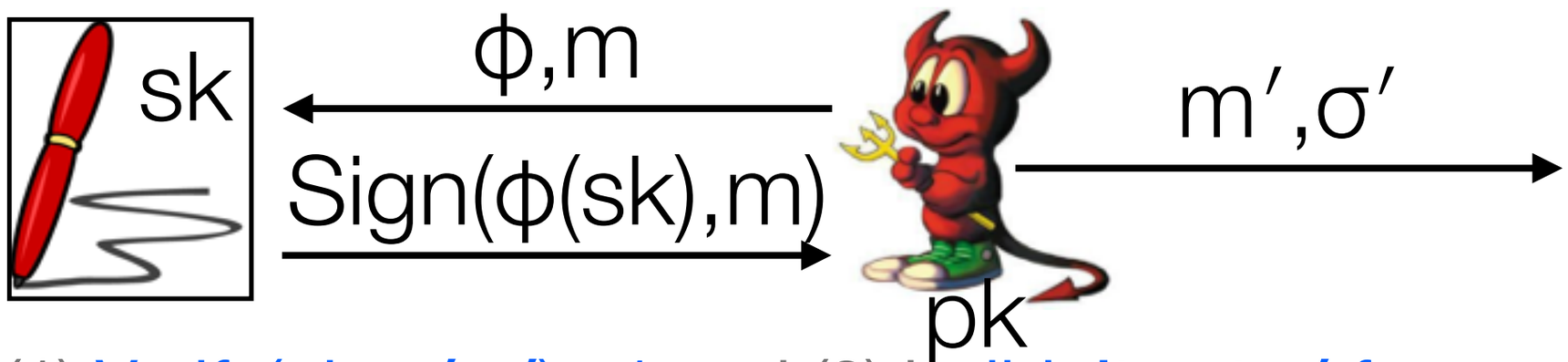


A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)

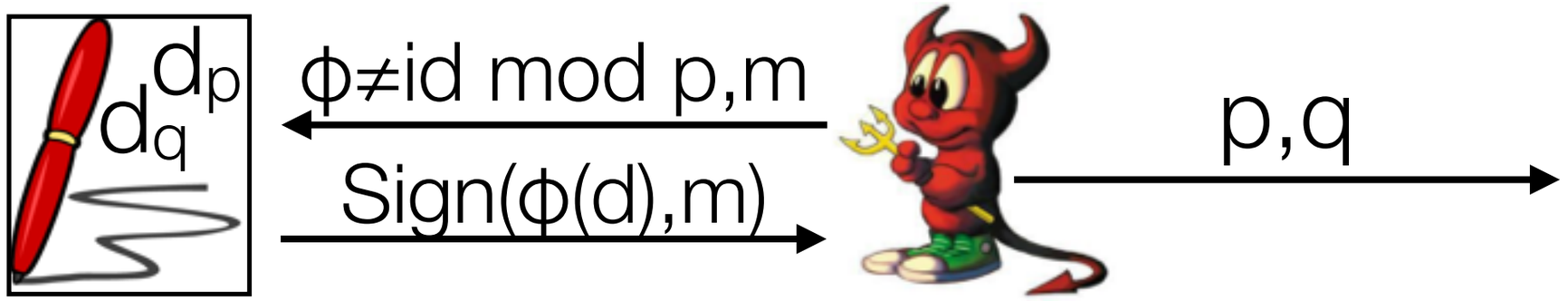
Φ={id} gives standard unforgeability

Φ={all functions} isn't possible [BK03]

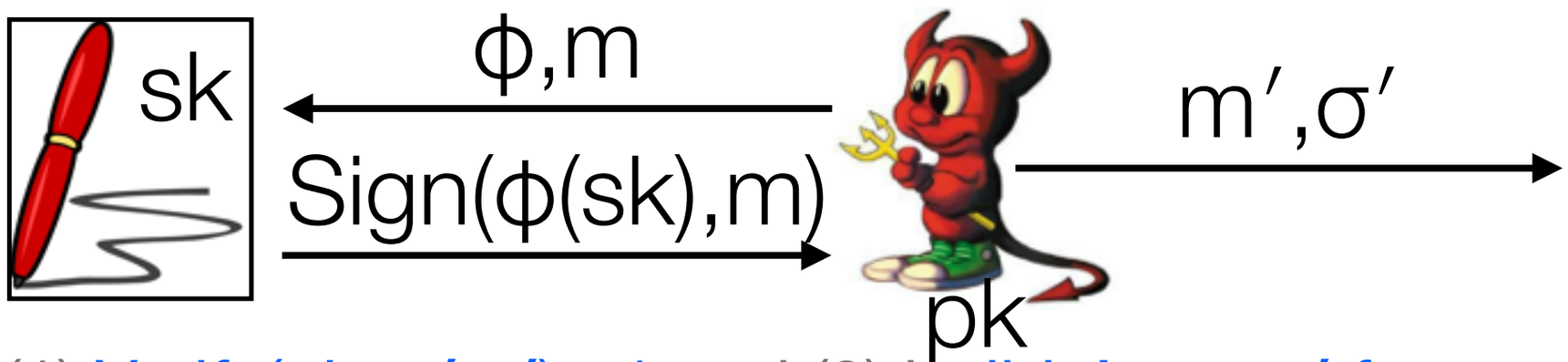
ϕ-RKA-secure signatures [BCM11]



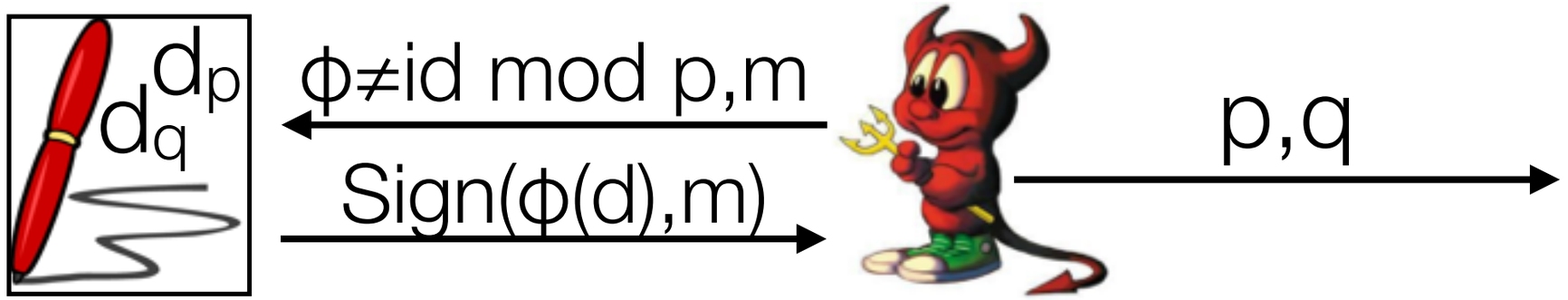
A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)



ϕ-RKA-secure signatures [BCM11]

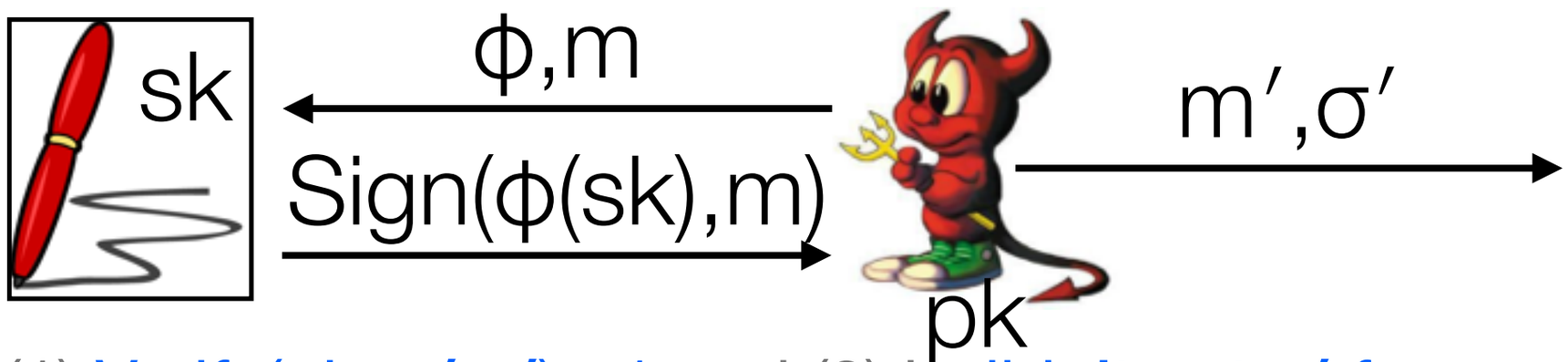


A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)

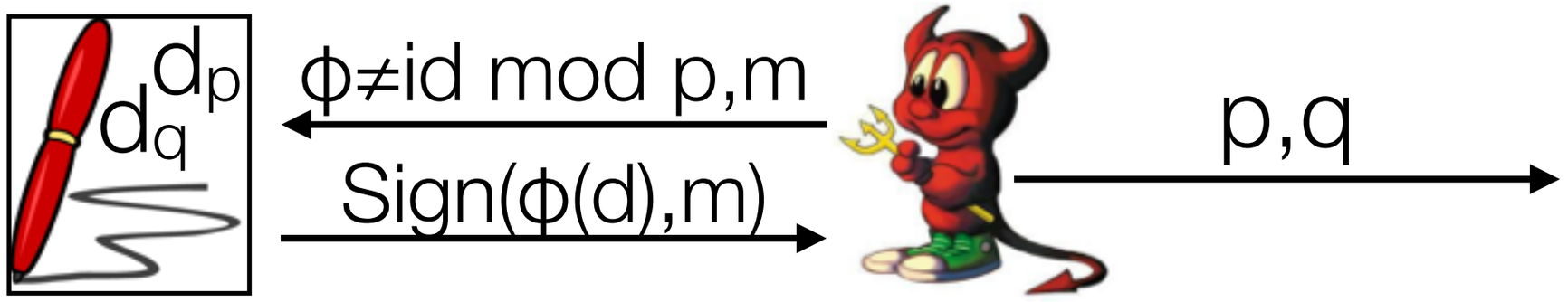


[BdML97] shows that RSA-CRT is not ϕ-RKA-secure for non-trivial ϕ

ϕ-RKA-secure signatures [BCM11]



A wins if (1) Verify(pk,m′,σ′) =1 and (2) it didn't get σ′ from oracle (when querying on ϕ=id)



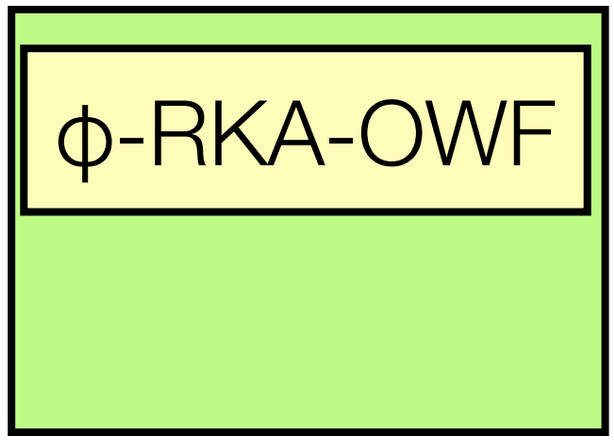
[BdML97] shows that RSA-CRT is not ϕ-RKA-secure for non-trivial ϕ

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

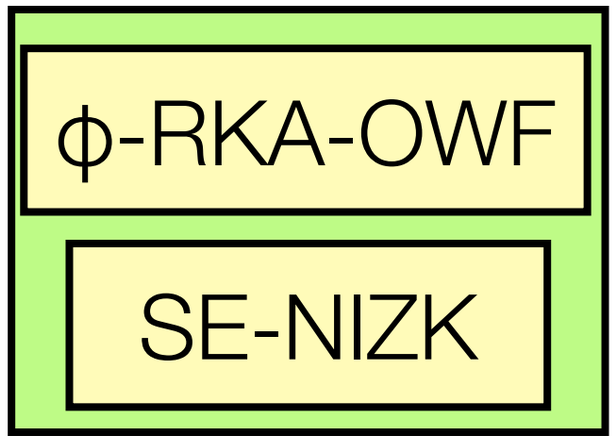
Our construction [Bellare **M** Thomson Eurocrypt14]

ϕ-RKA-OWF

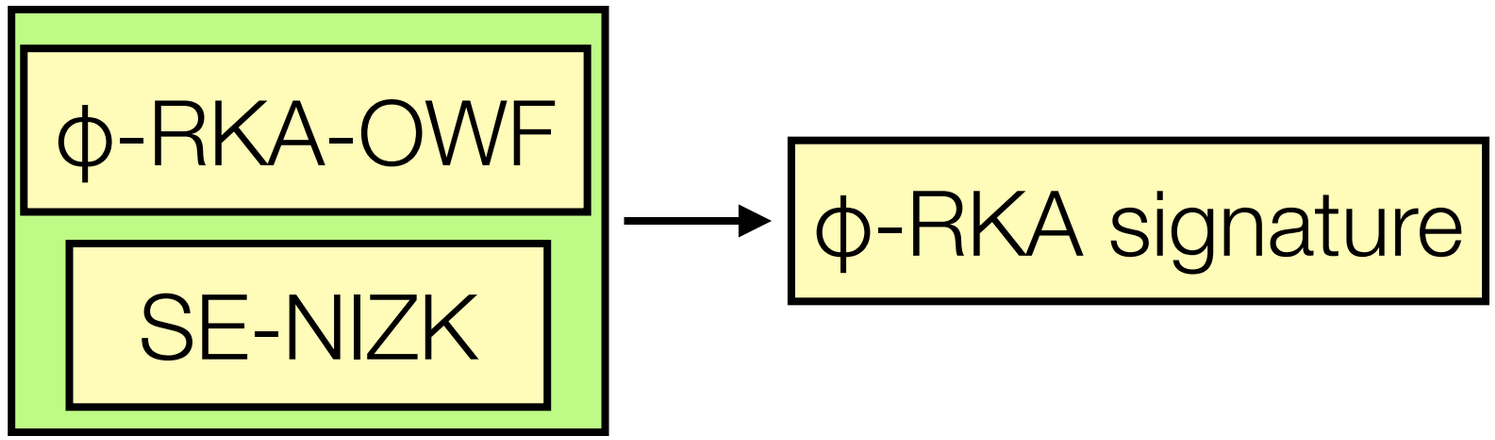
Our construction [Bellare **M** Thomson Eurocrypt14]



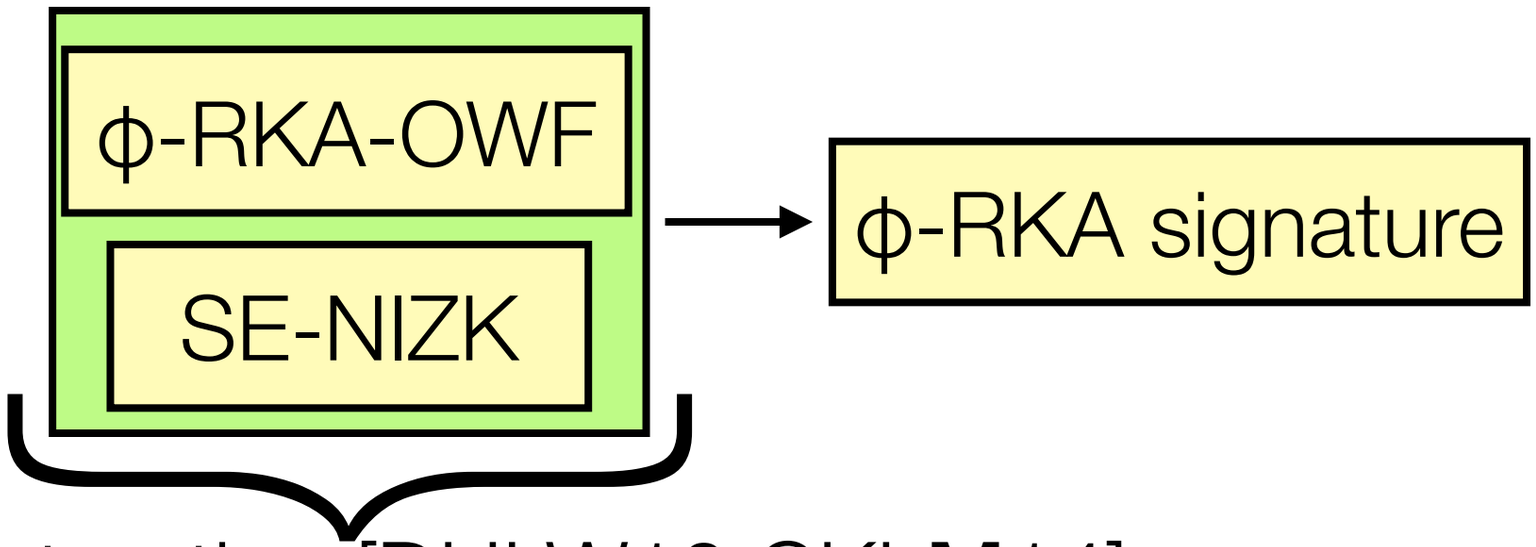
Our construction [Bellare **M** Thomson Eurocrypt14]



Our construction [Bellare **M** Thomson Eurocrypt14]



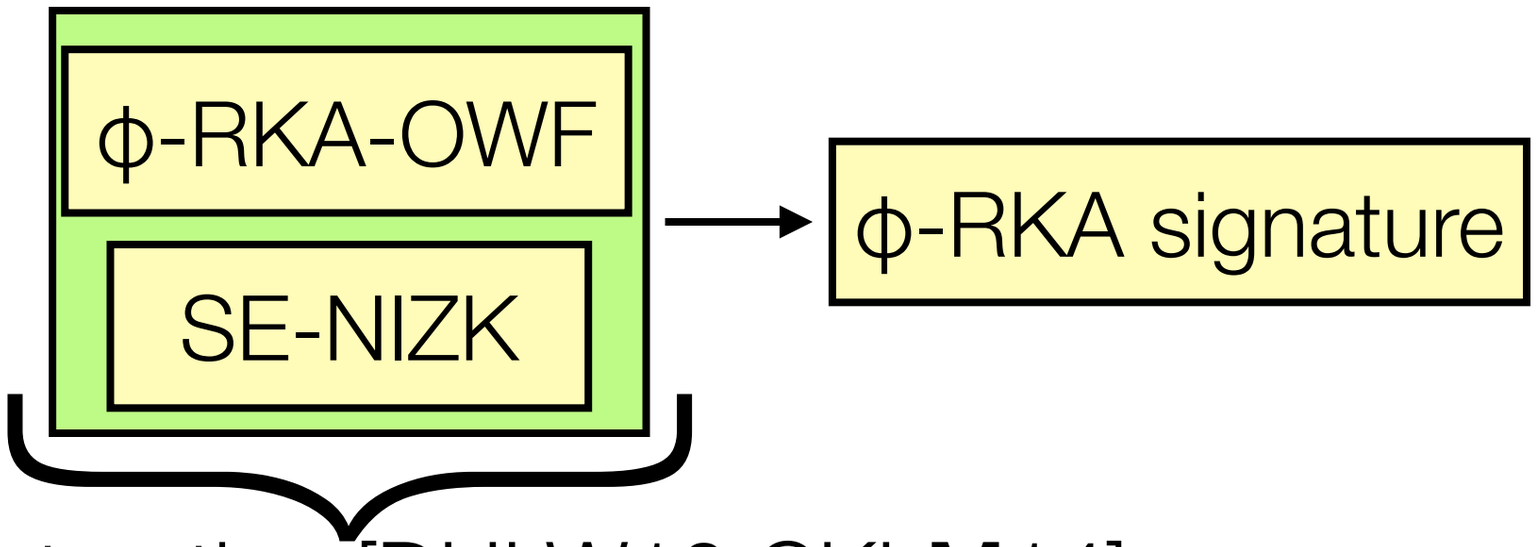
Our construction [Bellare **M** Thomson Eurocrypt14]



simple construction [DHLW10,CKL**M**14]

- KeyGen: crs←CRSGen; x←Dom(f); y←f(x)
  return (pk=(crs,y),sk=x)
- Sign(sk,m): return Prove(crs,y||m,x)
- Verify(pk,σ,m): return Verify(crs,y||m,σ)

Our construction [Bellare **M** Thomson Eurocrypt14]

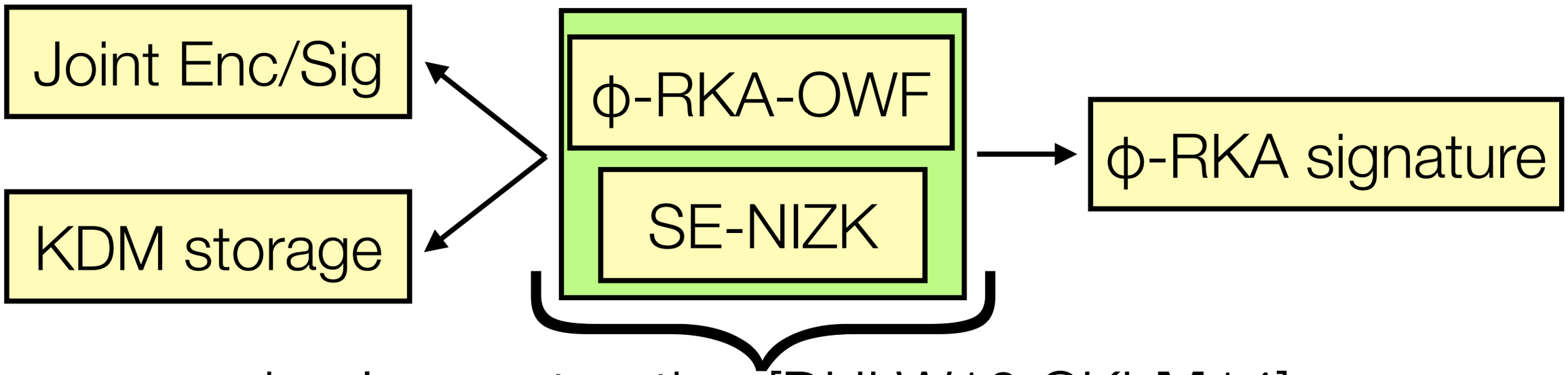


simple construction [DHLW10,CKL**M**14]

- KeyGen: crs←CRSGen; x←Dom(f); y←f(x)
  return (pk=(crs,y),sk=x)
- Sign(sk,m): return Prove(crs,y||m,x)
- Verify(pk,σ,m): return Verify(crs,y||m,σ)

proof of knowledge of secret key

Our construction [Bellare **M** Thomson Eurocrypt14]

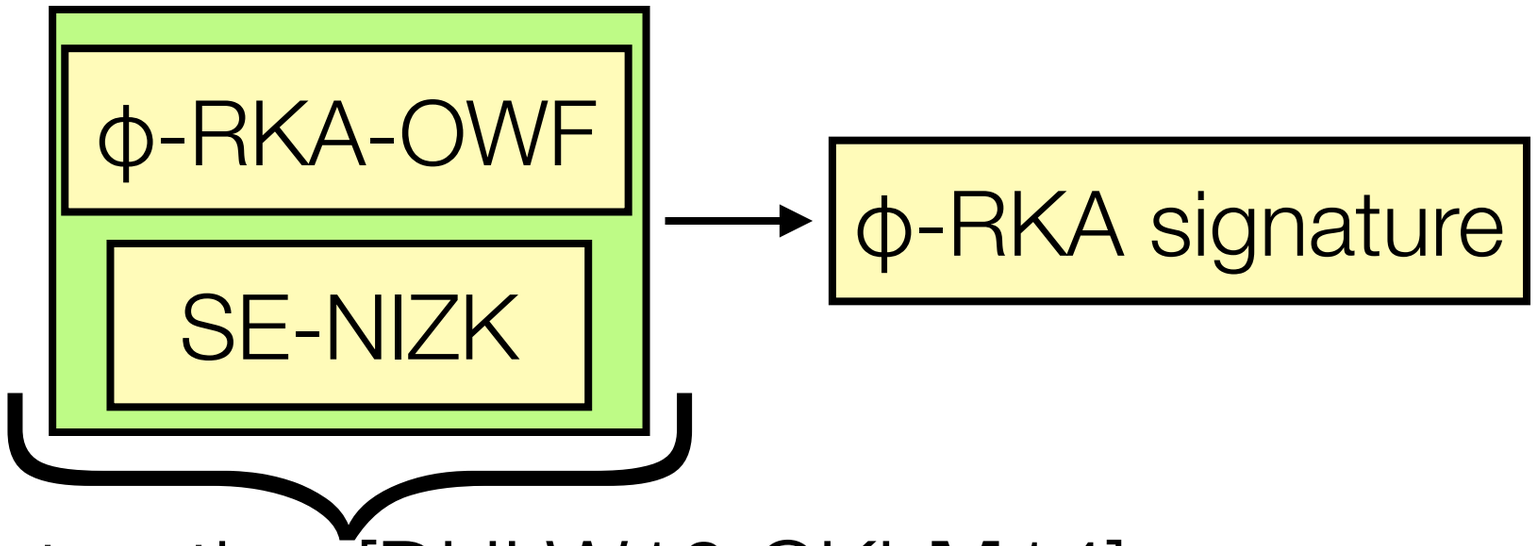


simple construction [DHLW10,CKL**M**14]

- KeyGen: crs←CRSGen; x←Dom(f); y←f(x)
  return (pk=(crs,y),sk=x)
- Sign(sk,m): return Prove(crs,y||m,x)
- Verify(pk,σ,m): return Verify(crs,y||m,σ)

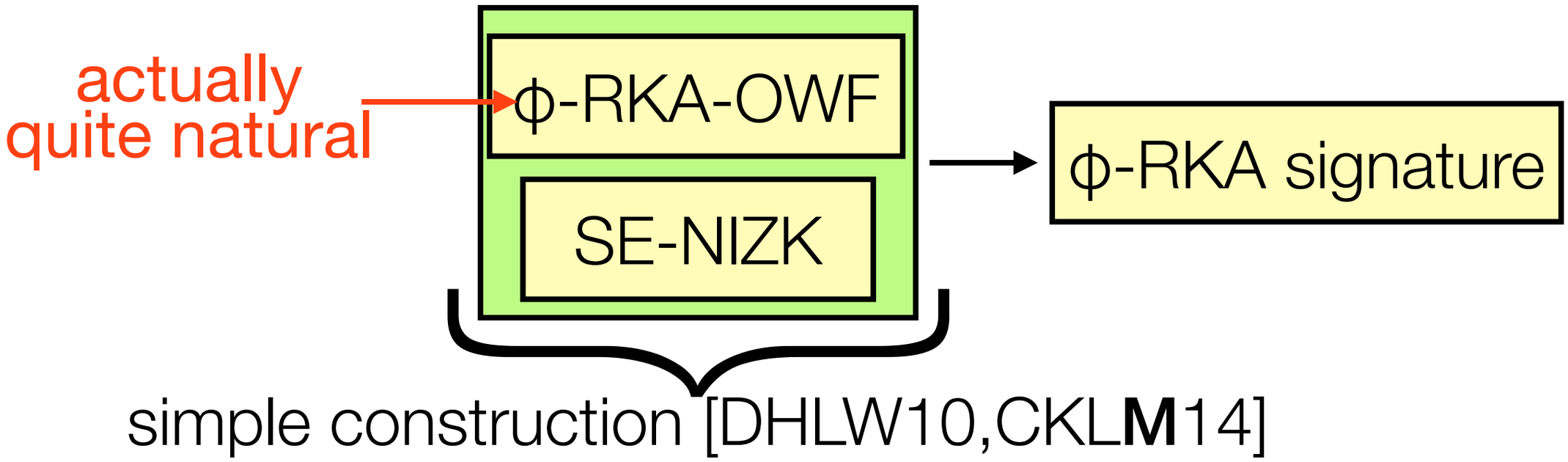
proof of knowledge of secret key

Our construction [Bellare **M** Thomson Eurocrypt14]



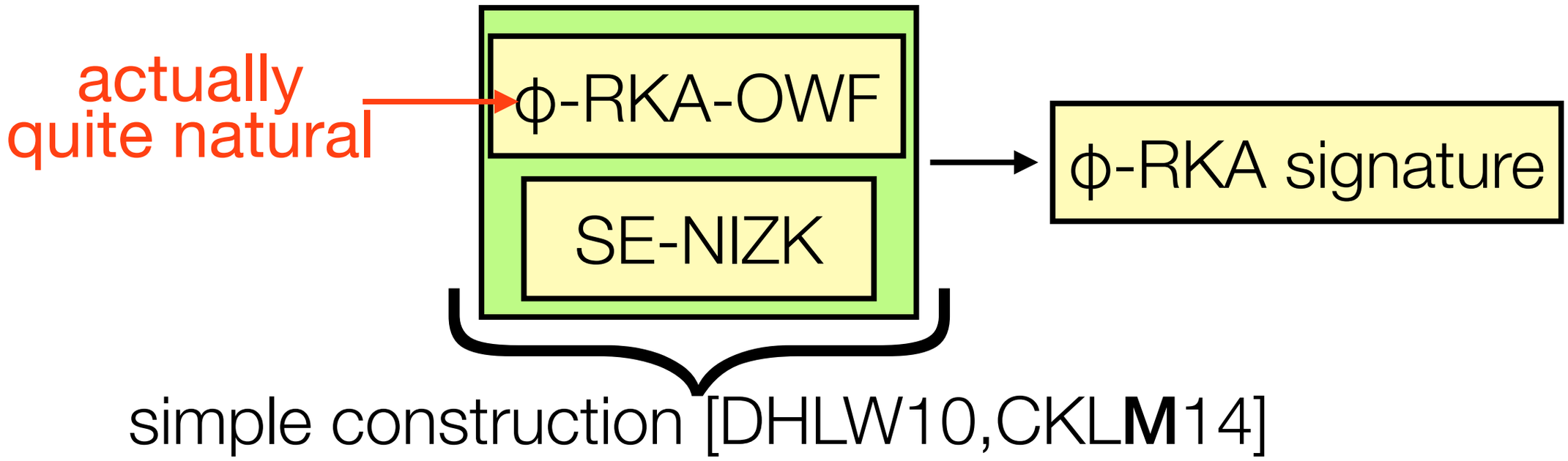
Signature inherits Φ-RKA security from one-way function

Our construction [Bellare **M** Thomson Eurocrypt14]



Signature inherits Φ-RKA security from one-way function

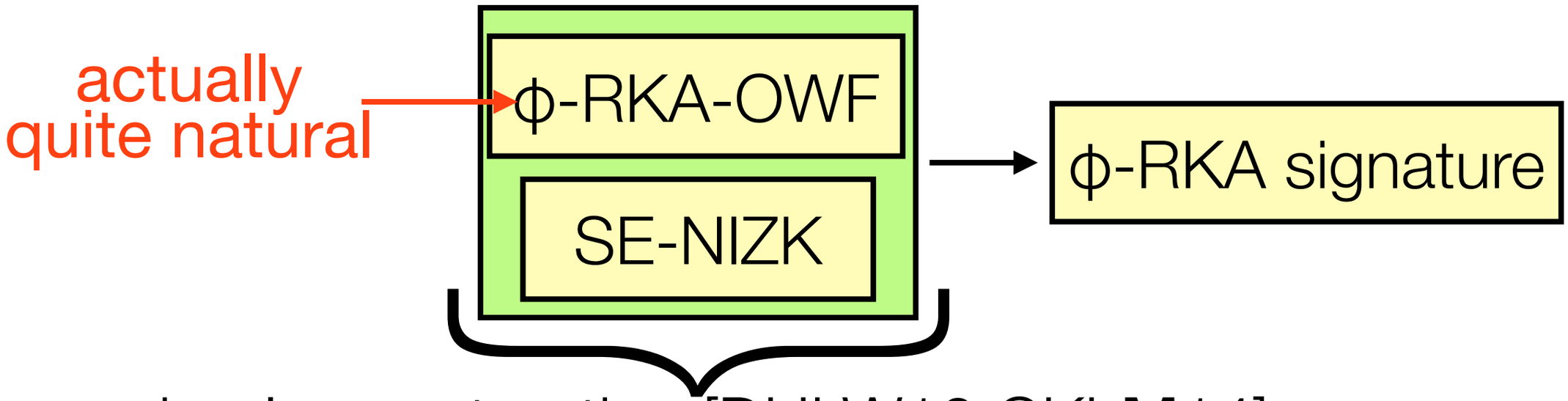
Our construction [Bellare **M** Thomson Eurocrypt14]



Signature inherits Φ-RKA security from one-way function

Many natural one-way functions (e.g., RSA) are Φ-RKA-secure with no additional assumptions

Our construction [Bellare **M** Thomson Eurocrypt14]



Signature inherits Φ-RKA security from one-way function

Many natural one-way functions (e.g., RSA) are Φ-RKA-secure with no additional assumptions

Creating new RKA-secure signatures is easier!

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Φ-RKA-secure one-way functions are natural

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Φ-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Φ-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x)=g^x$) is secure w.r.t. linear functions

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Φ-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x)=g^x$) is secure w.r.t. linear functions
- Learning with errors ($f(s,e) = As+e \bmod q$) is secure w.r.t. addition

Takeaway

Problem: ϕ-RKA schemes are really hard to construct
(for interesting classes ϕ; for most primitives)

Φ-RKA-secure one-way functions are natural

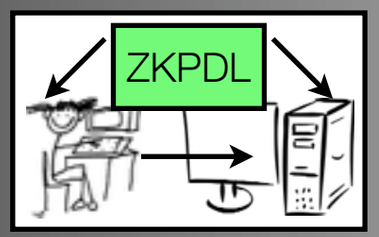
- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x)=g^x$) is secure w.r.t. linear functions
- Learning with errors ($f(s,e) = As+e \bmod q$) is secure w.r.t. addition

Our result can pave the way for easier RKA constructions

My research

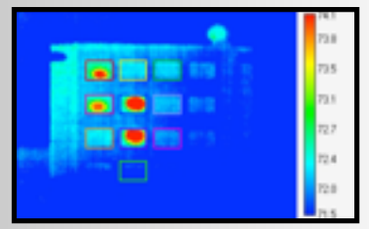

[**M**SF10,L**M**13]

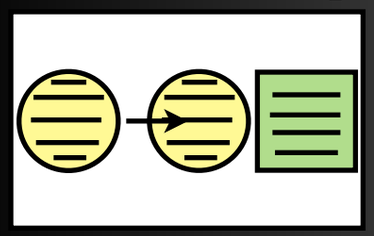

[B**M**T14]

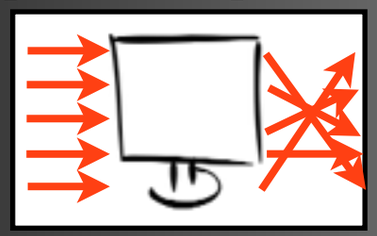


[**M**EKHL10]

[**M**MCS11]

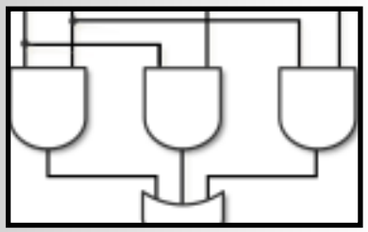

[M**M**S11]


[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]

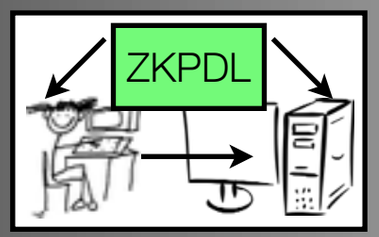

[**M**P+13,HD**M**+14]


[O**M**SK13]

My research

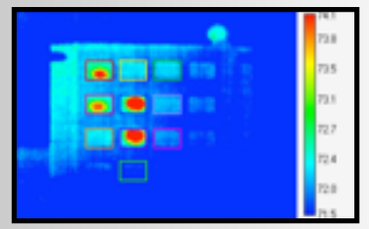

[**M**SF10,L**M**13]

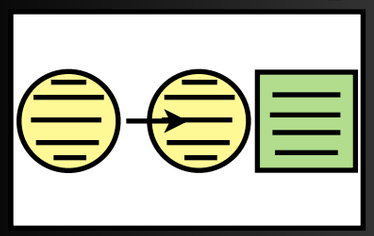

[B**M**T14]

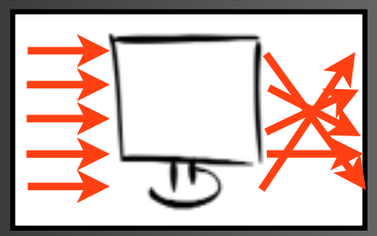


[**M**EKHL10]

[**M**MCS11]

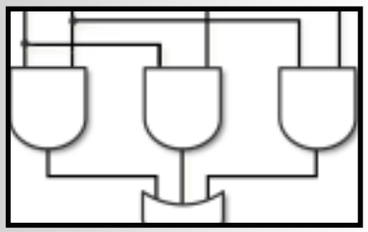

[M**M**S11]

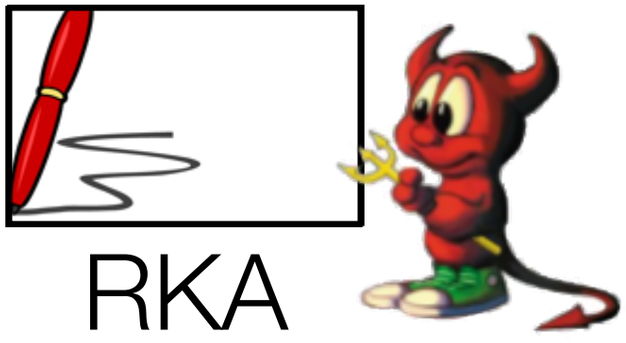

[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]



What is Bitcoin?

What is Bitcoin?

What is Bitcoin?

Centralized

What is Bitcoin?

Centralized

Decentralized

What is Bitcoin?

Centralized

Real-world identities

Decentralized

What is Bitcoin?

Centralized

Real-world identities

Decentralized

Pseudonyms

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Decentralized

Pseudonyms

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Decentralized

Pseudonyms

Public transactions

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Regulated

Decentralized

Pseudonyms

Public transactions

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Regulated

Decentralized

Pseudonyms

Public transactions

Unregulated*

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Regulated

Not anonymous

Decentralized

Pseudonyms

Public transactions

Unregulated*

What is Bitcoin?

Centralized

Real-world identities

Non-public transactions

Regulated

Not anonymous

Decentralized

Pseudonyms

Public transactions

Unregulated*

Potentially anonymous

Why study Bitcoin?

Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Ponzi-Scheme Charge Is Good News for Bitcoin

Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Ponzi-Scheme Charge Is Good News for Bitcoin

Estimated 18 percent of US drug users bought from Silk Road, says study

Why study Bitcoin?

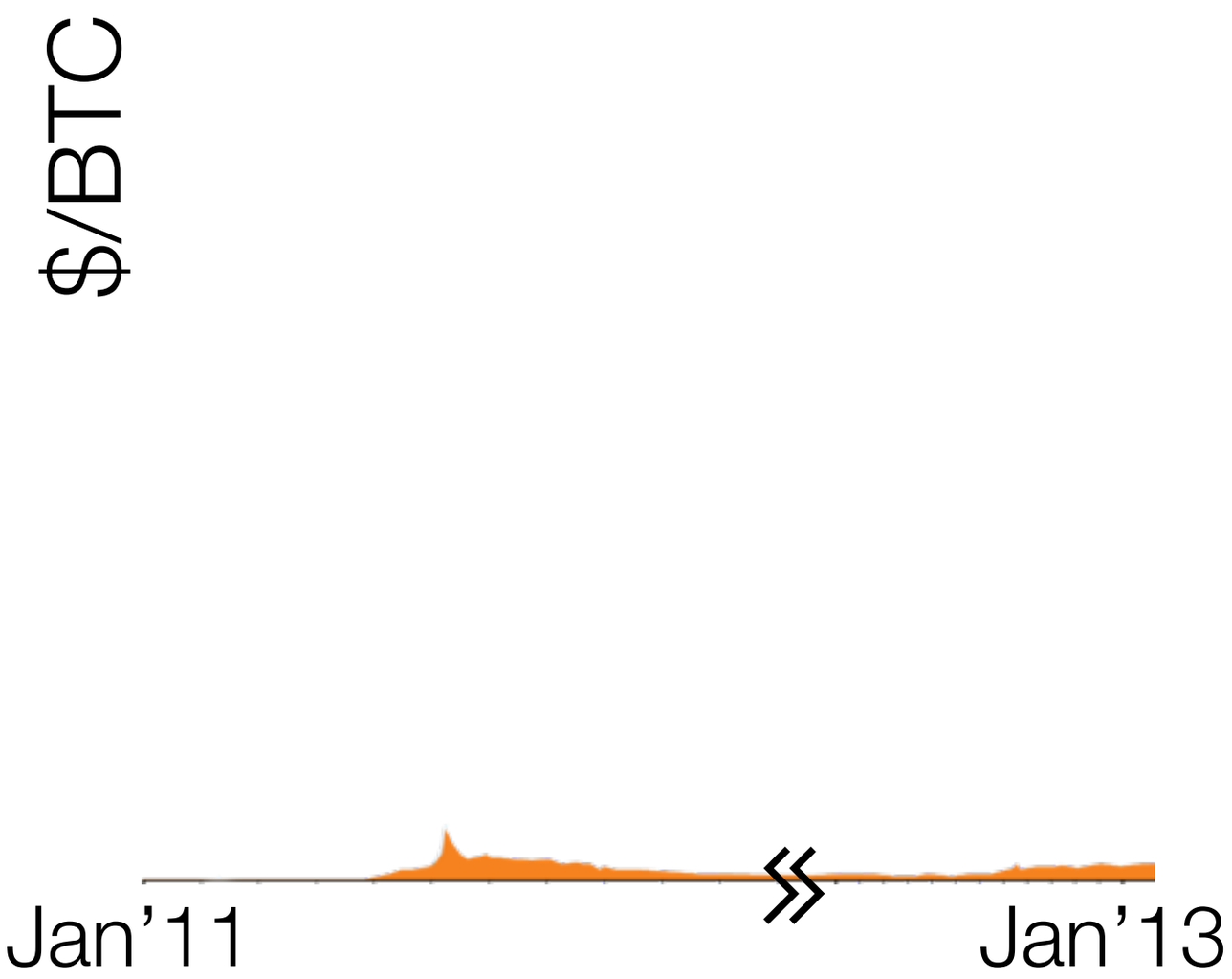
(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Ponzi-Scheme Charge Is Good News for Bitcoin

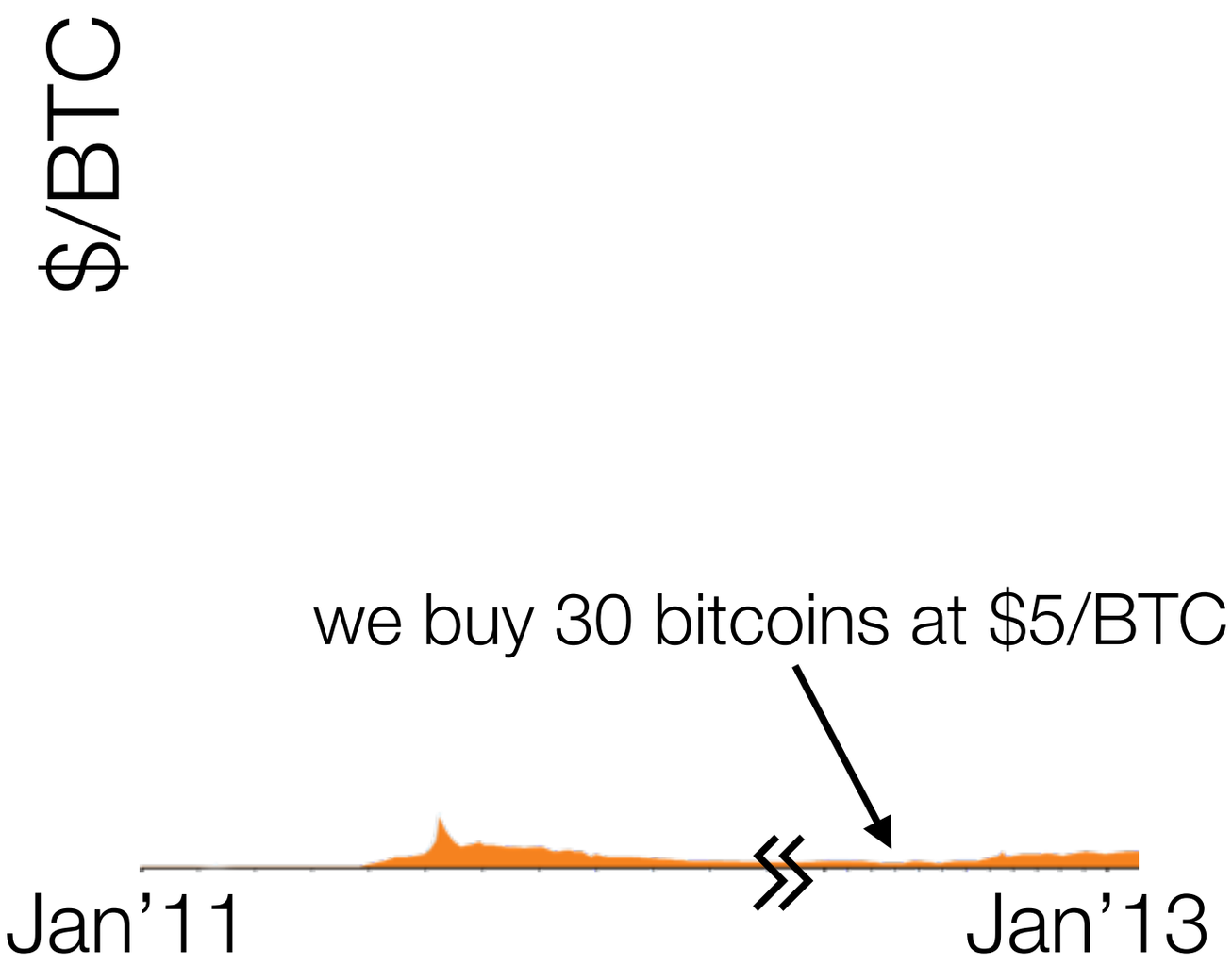
Estimated 18 percent of US drug users bought from Silk Road, says study

How much anonymity does Bitcoin really provide?

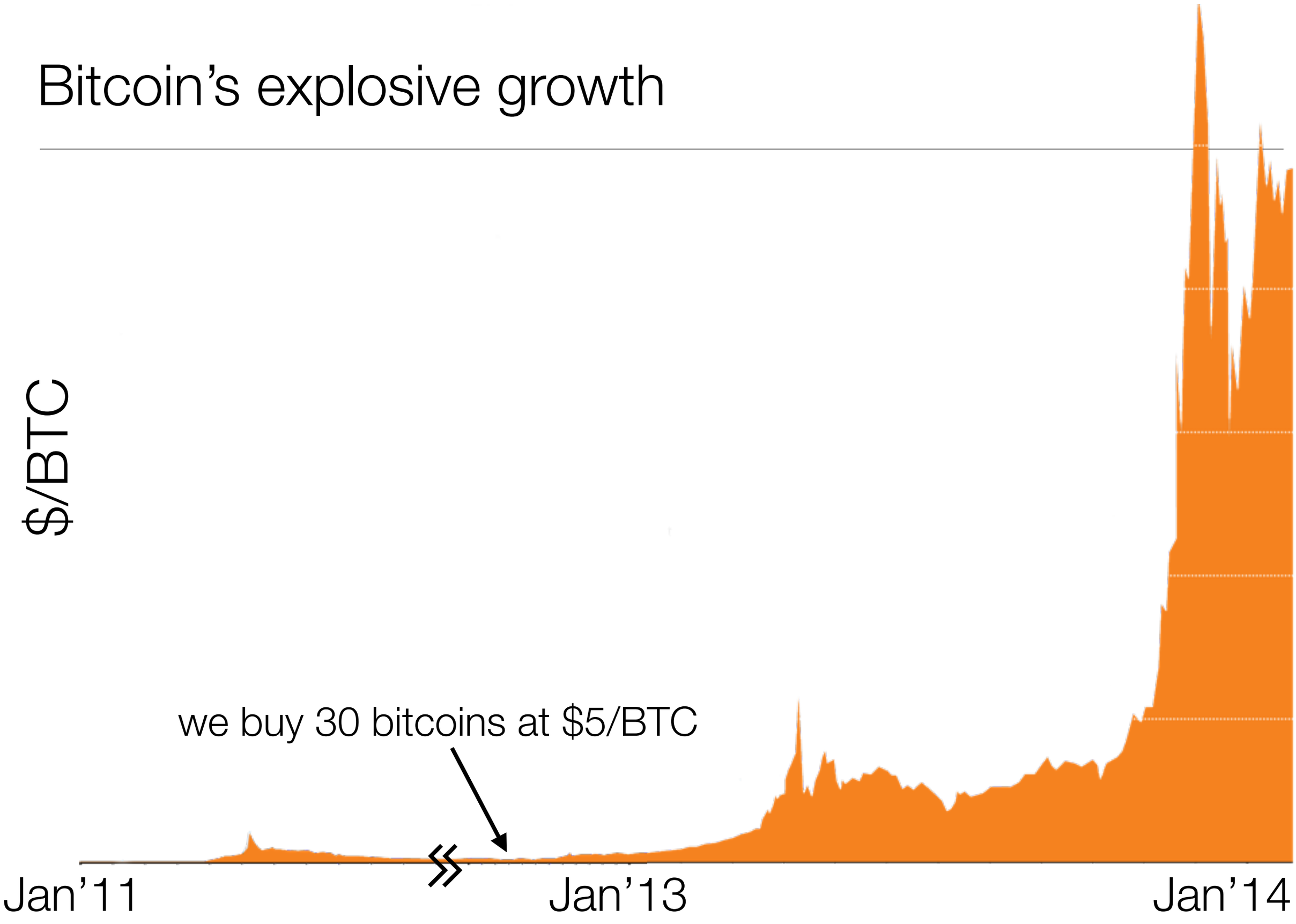
Bitcoin’s explosive growth



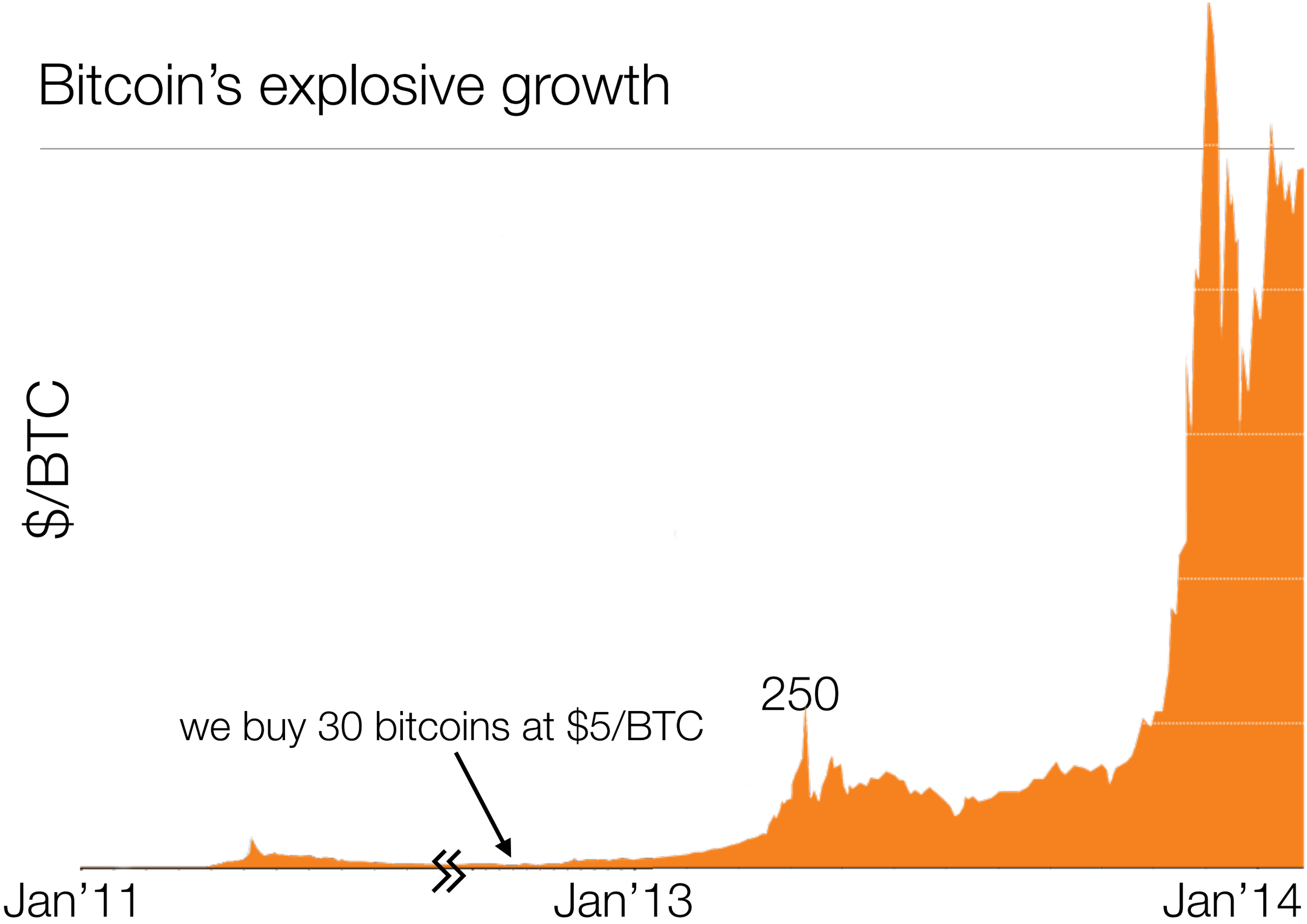
Bitcoin’s explosive growth



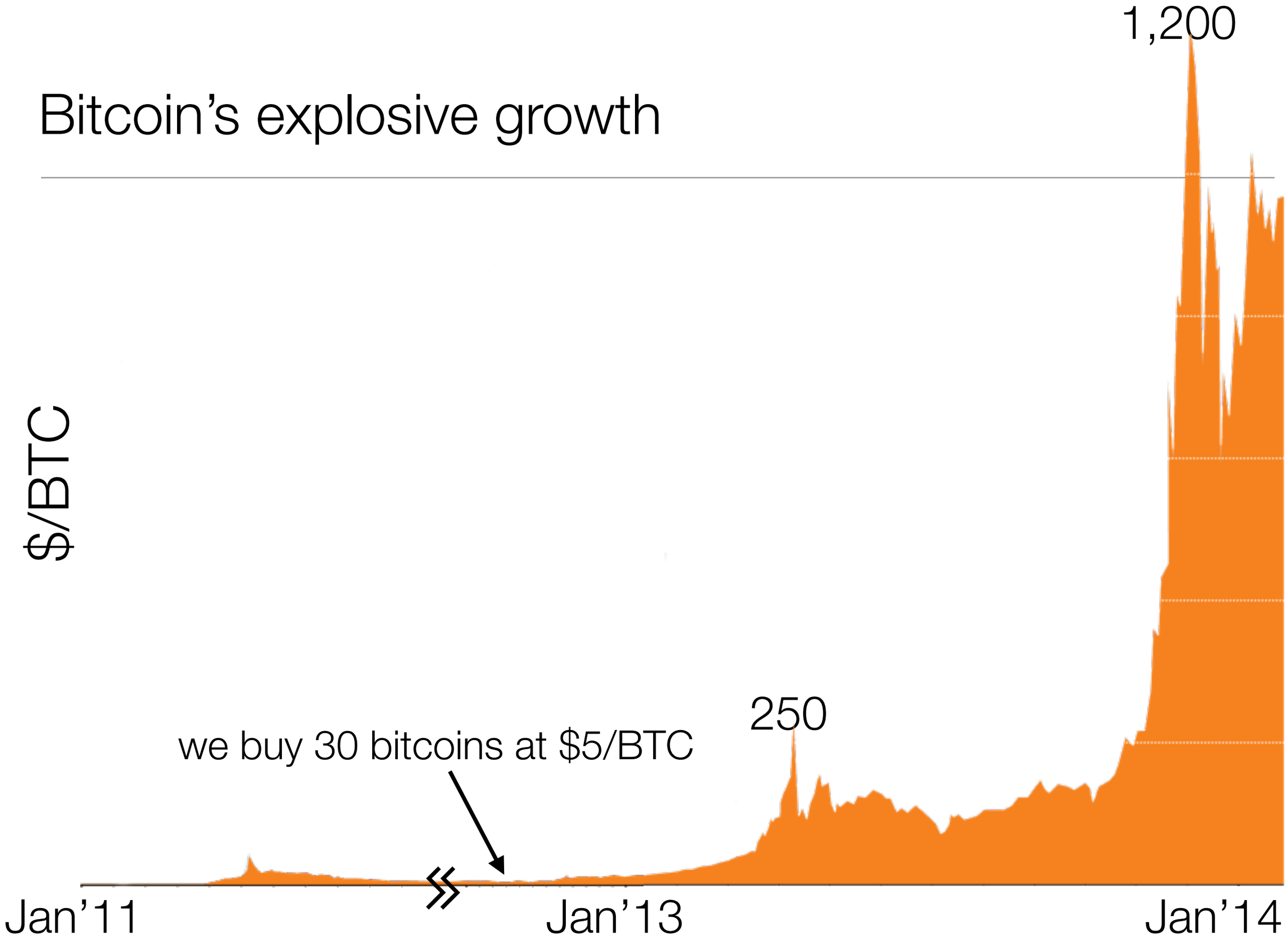
Bitcoin’s explosive growth



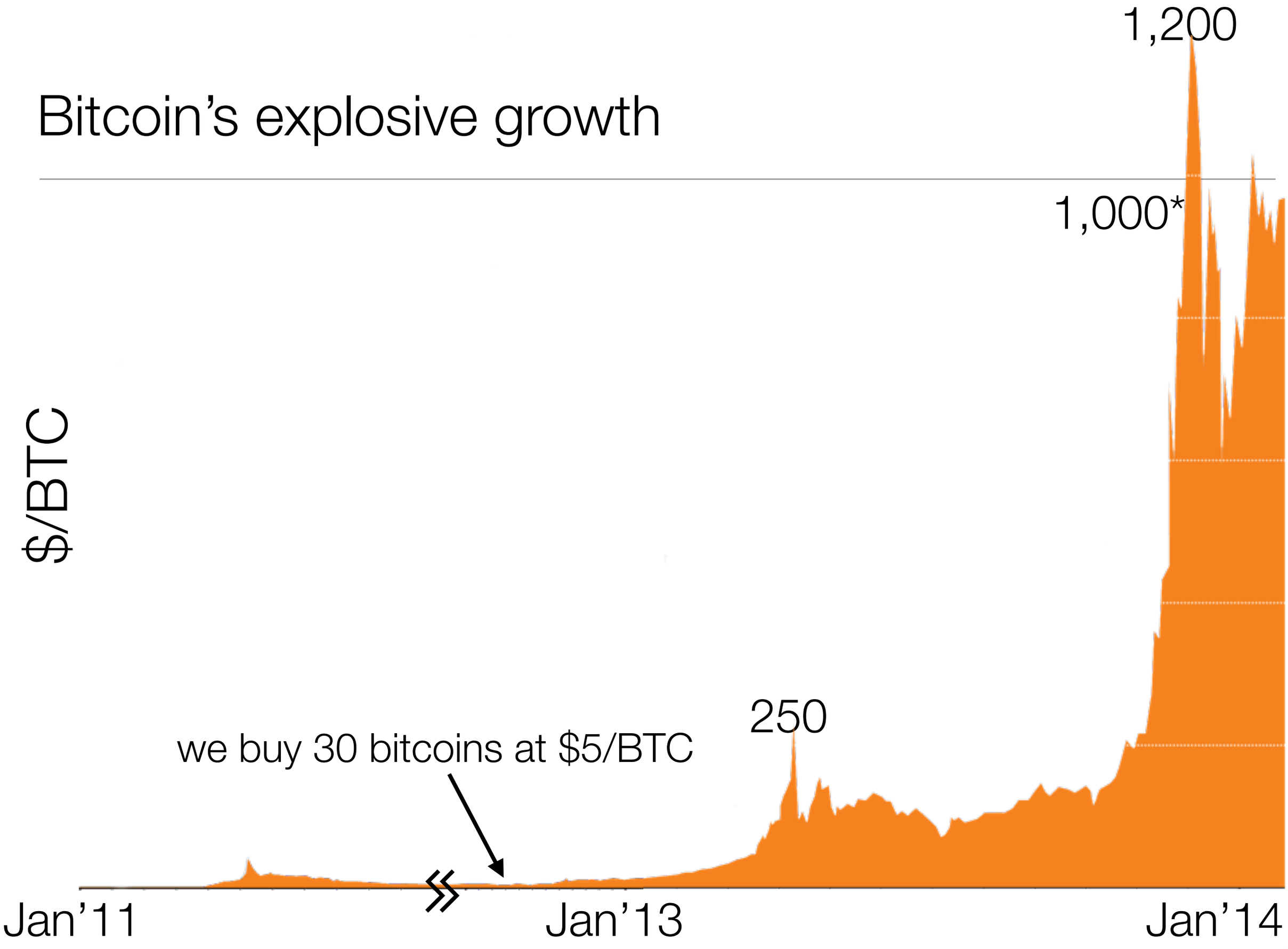
Bitcoin’s explosive growth



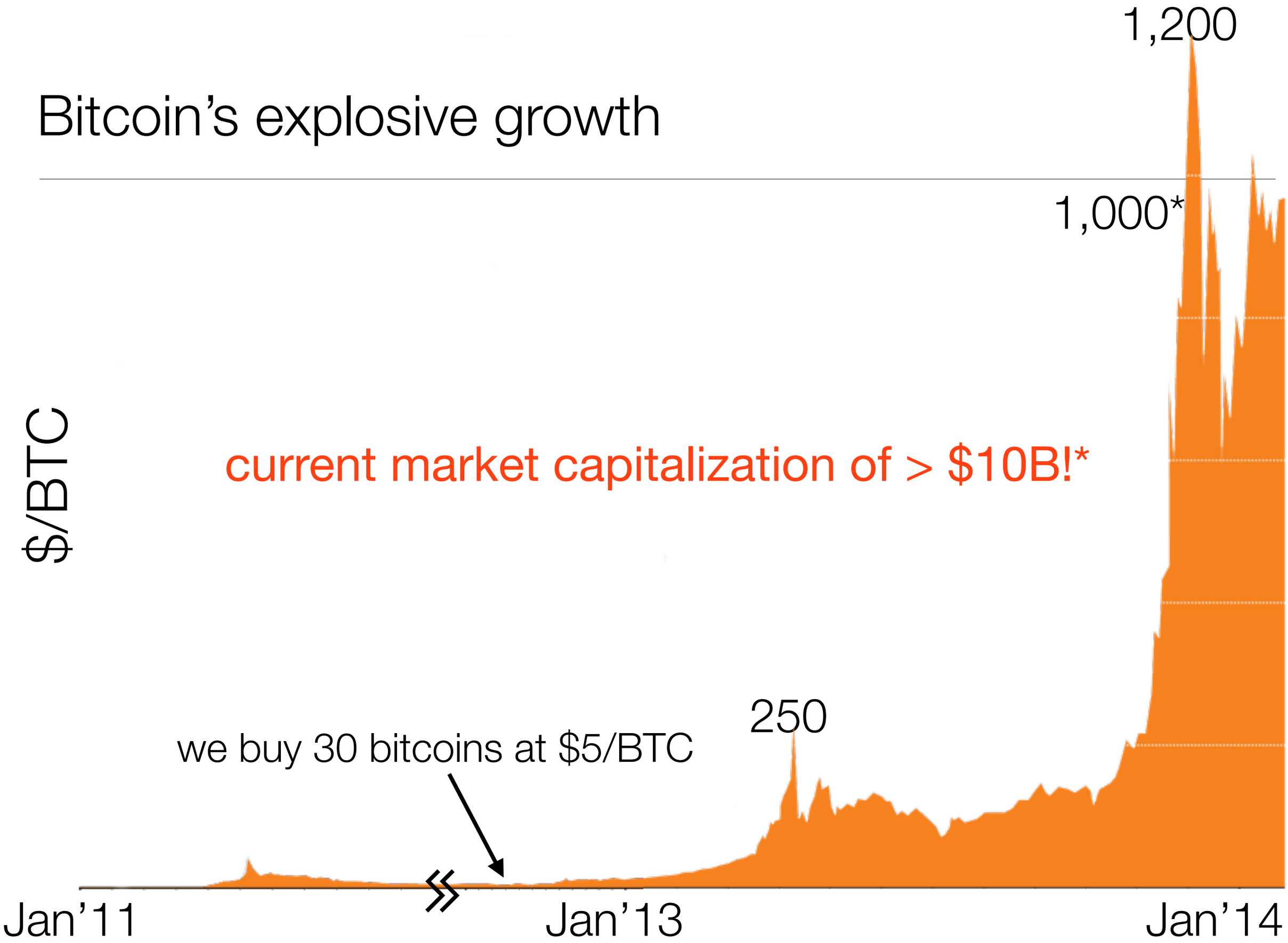
Bitcoin’s explosive growth



Bitcoin’s explosive growth



Bitcoin's explosive growth



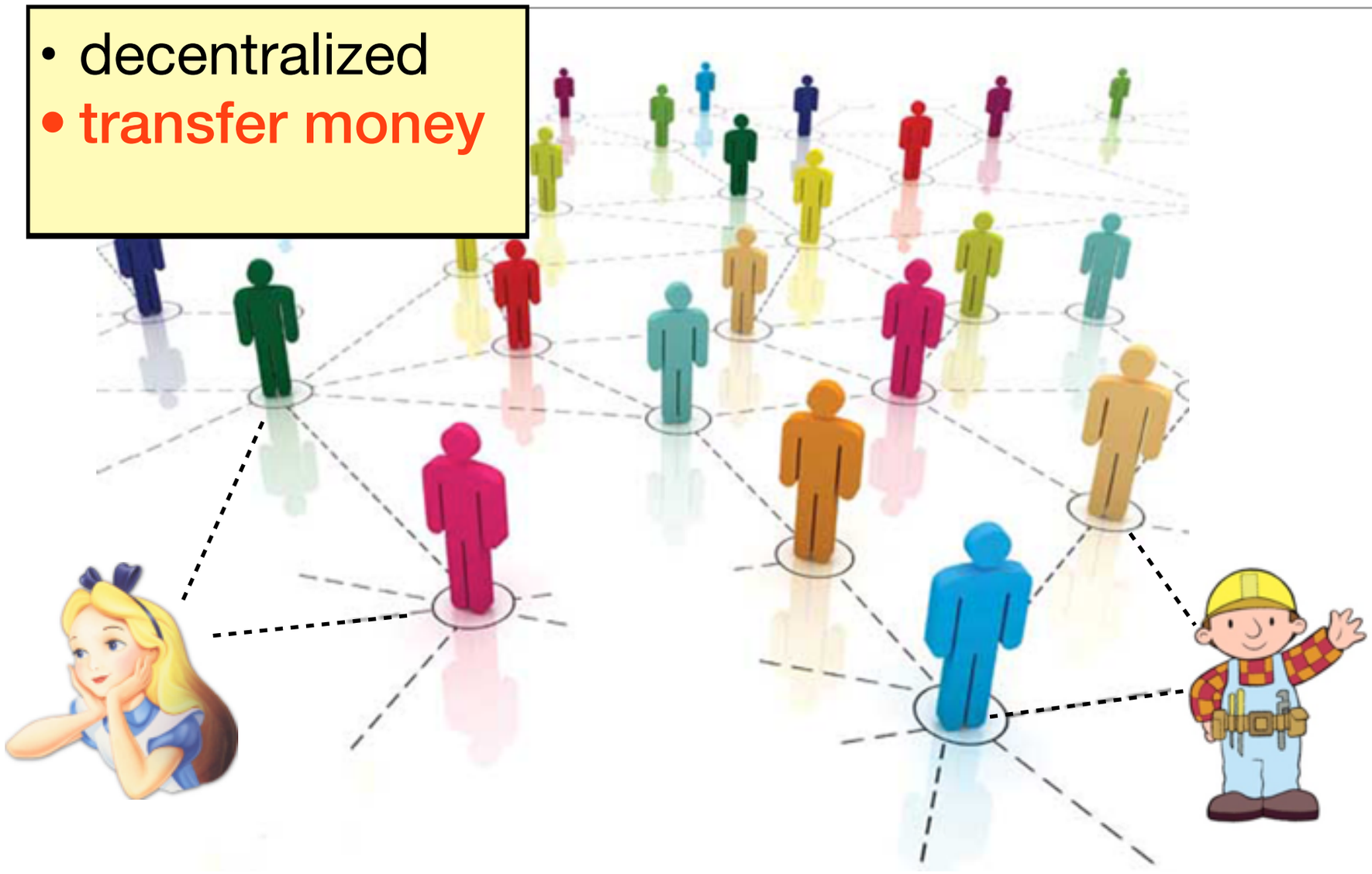
How does Bitcoin work?

How does Bitcoin work?

- **decentralized**

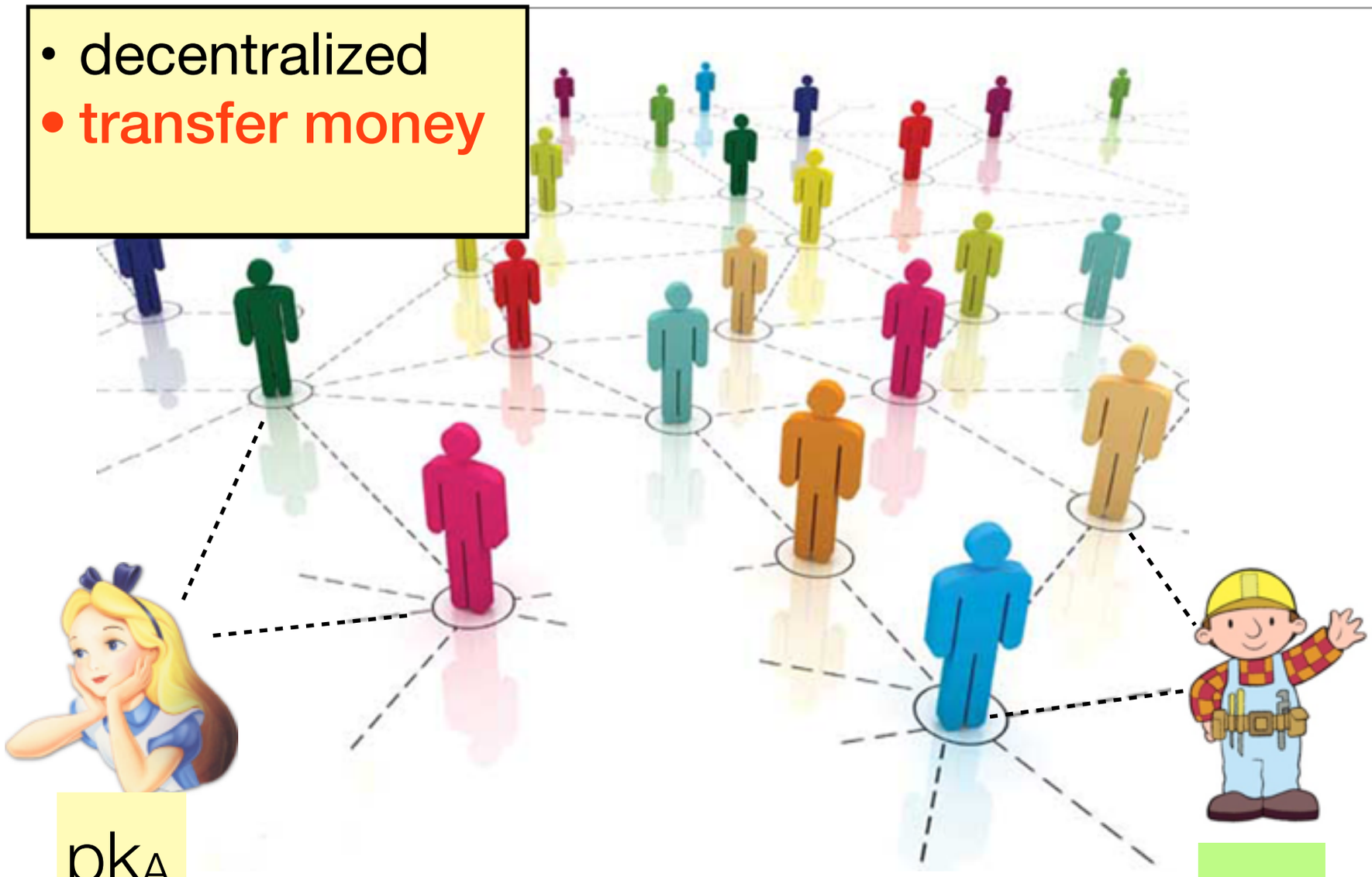
How does Bitcoin work?

- decentralized
- **transfer money**



How does Bitcoin work?

- decentralized
- **transfer money**



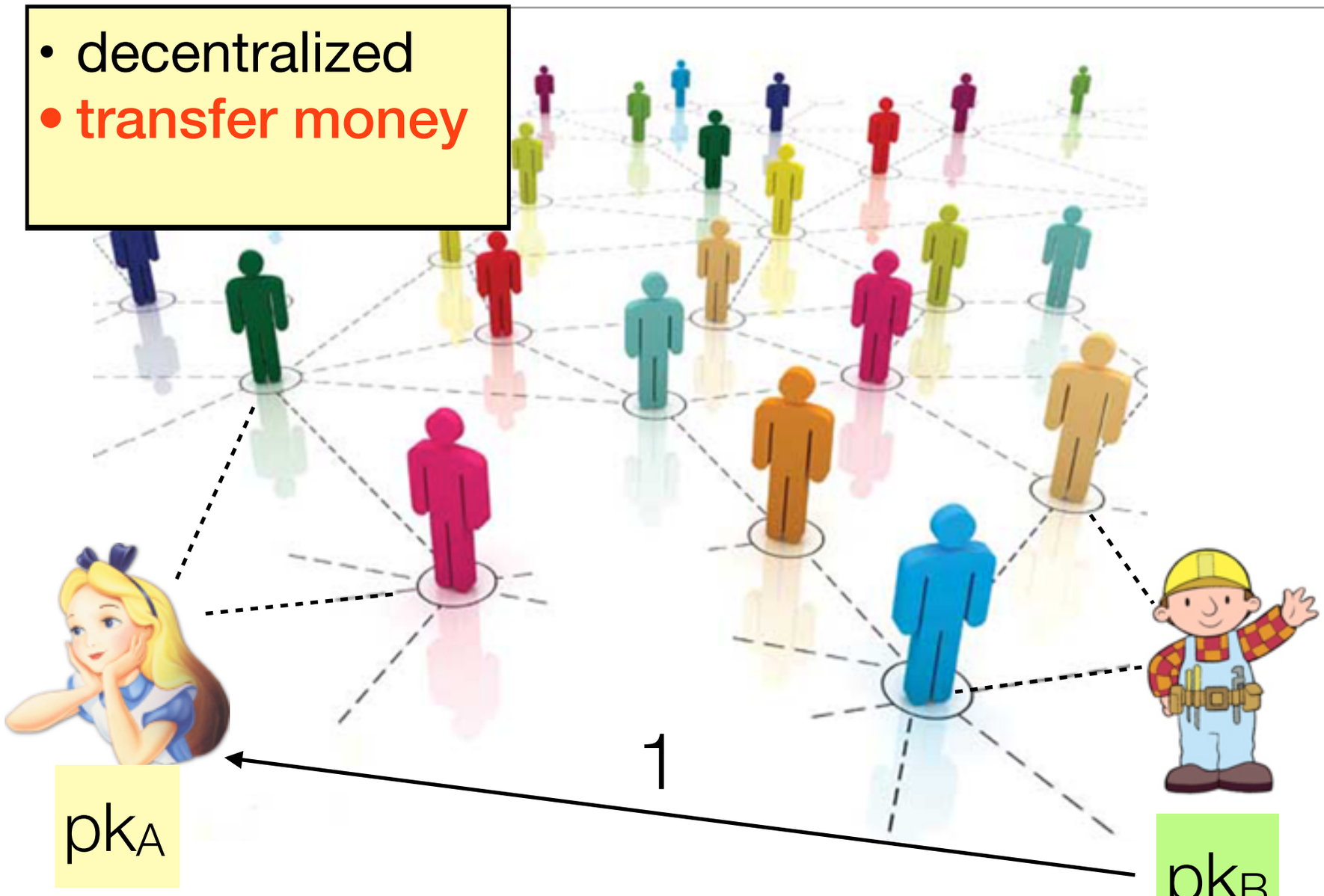
pk_A

pk_B

public key
address

How does Bitcoin work?

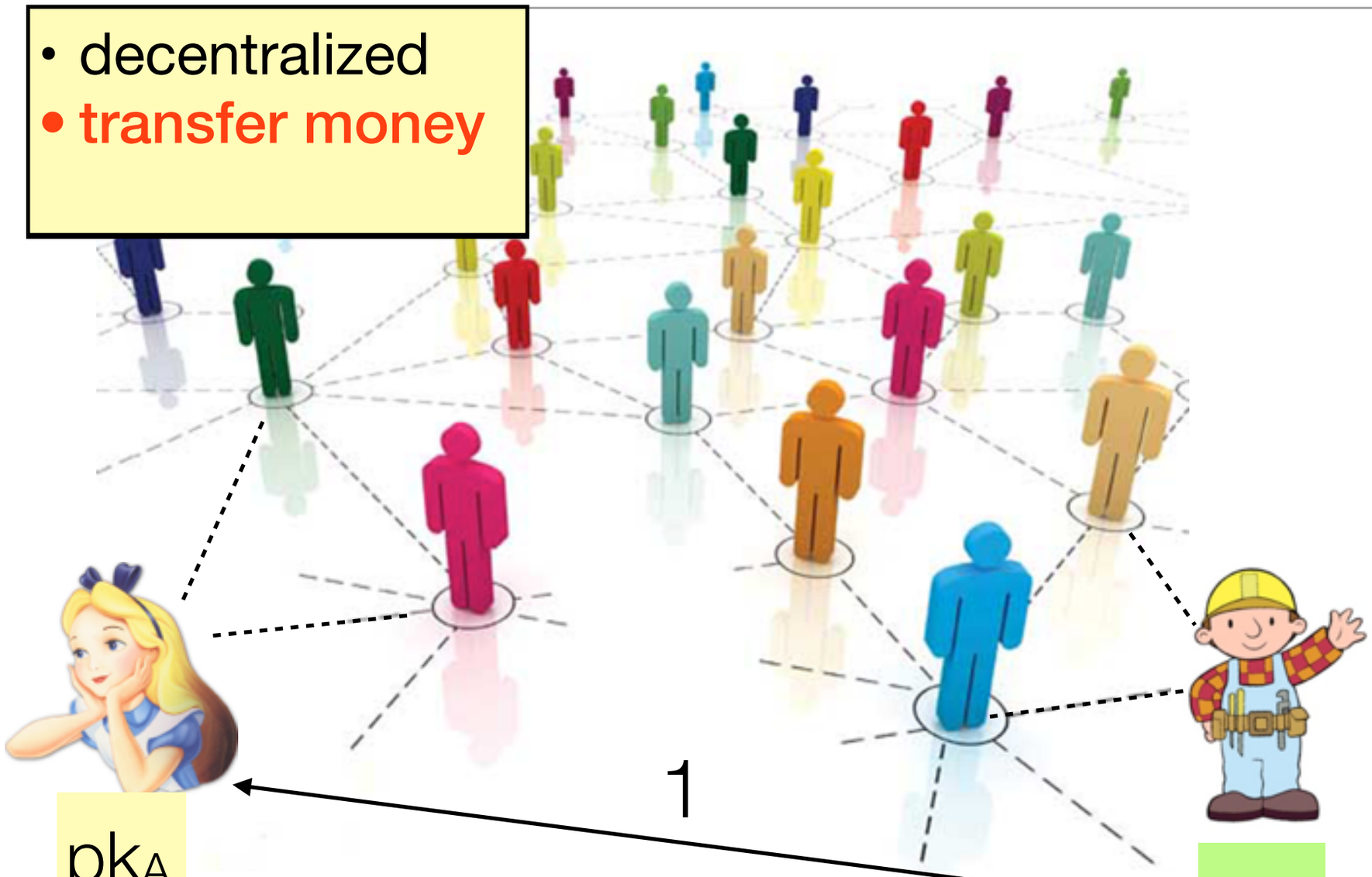
- decentralized
- **transfer money**



public key
address

How does Bitcoin work?

- decentralized
- **transfer money**



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

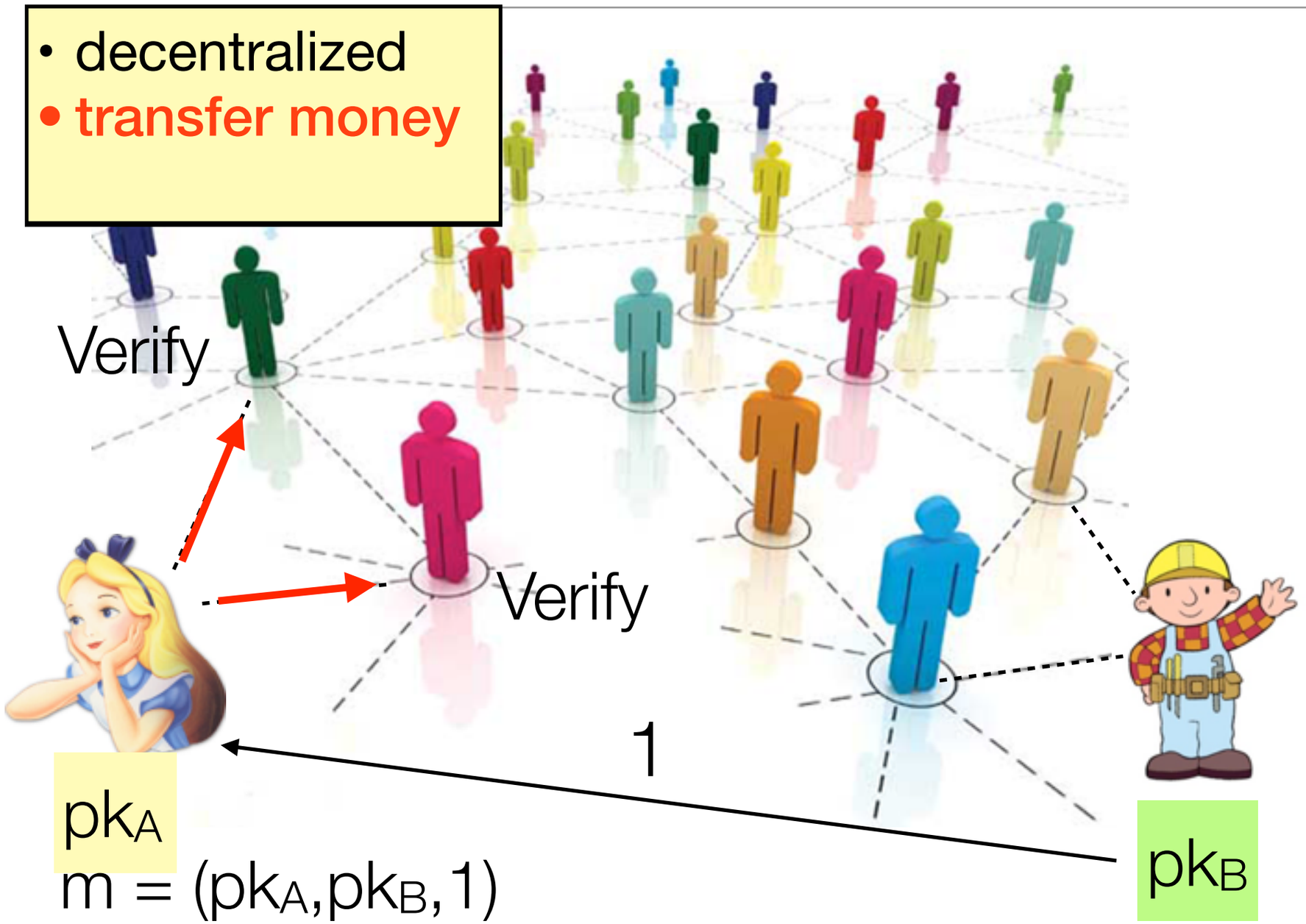
pk_B

public key

address

How does Bitcoin work?

- decentralized
- **transfer money**



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

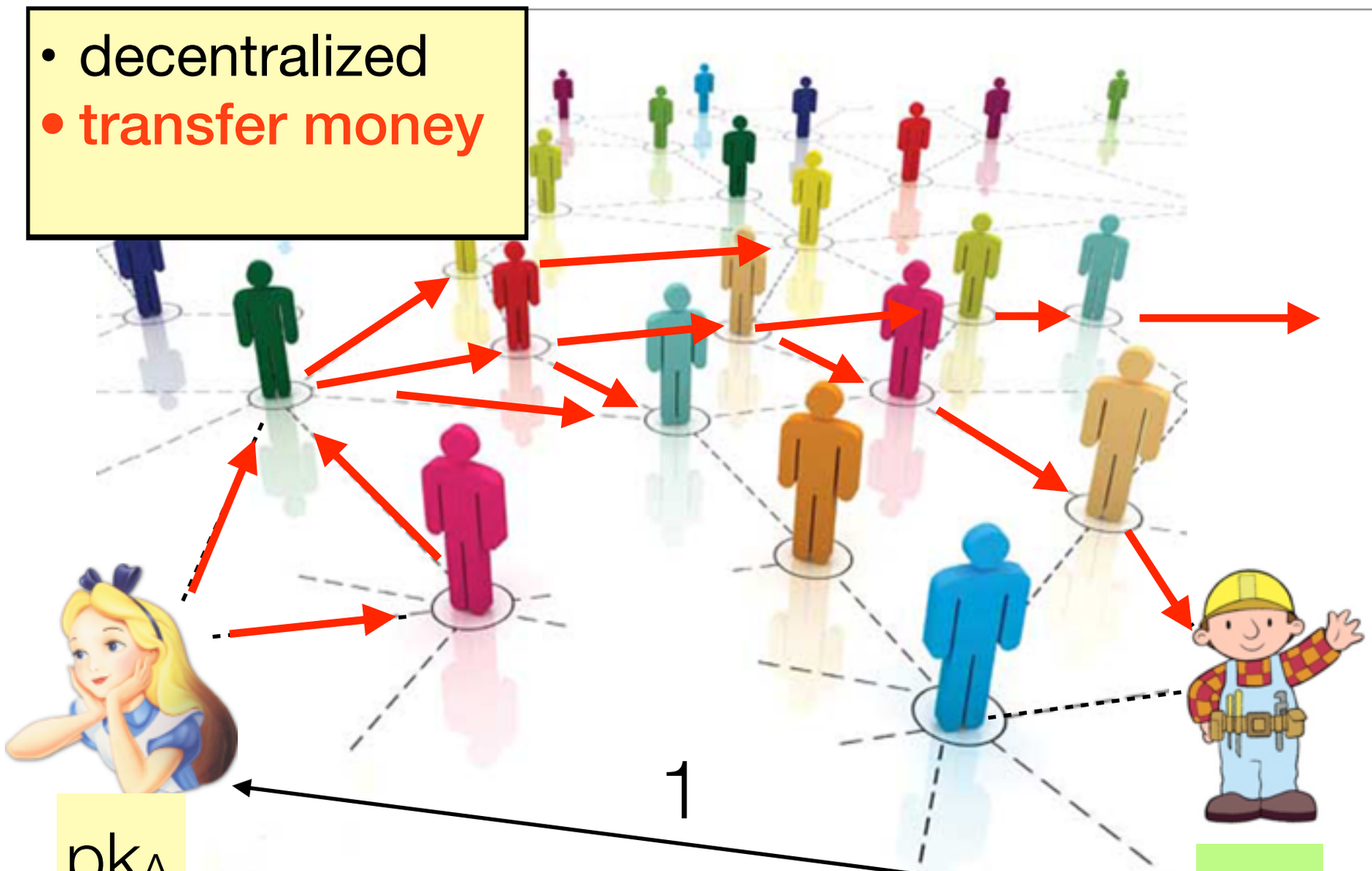
pk_B

public key

address

How does Bitcoin work?

- decentralized
- **transfer money**



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

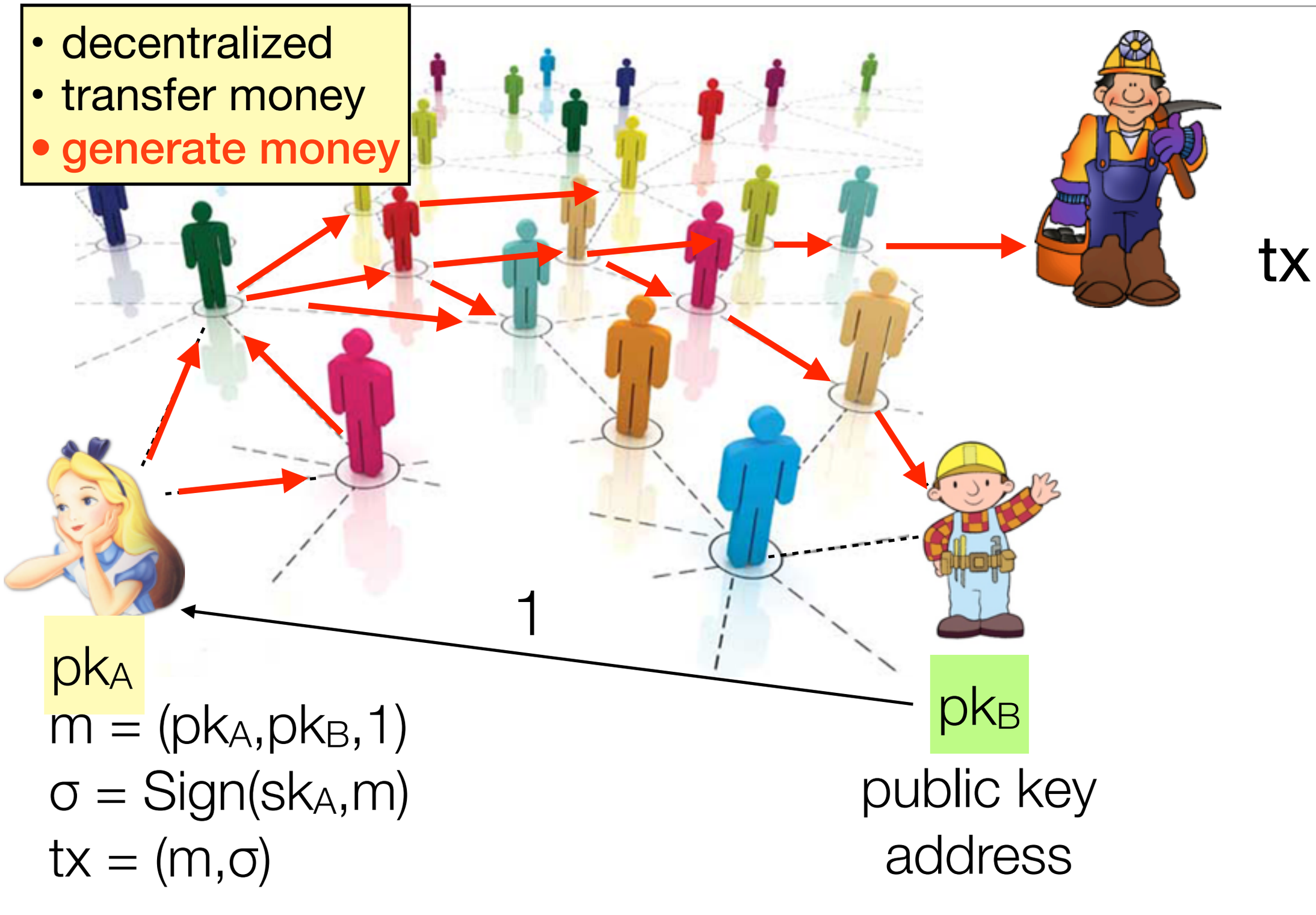
pk_B

public key

address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



$m = (pk_A, pk_B, 1)$

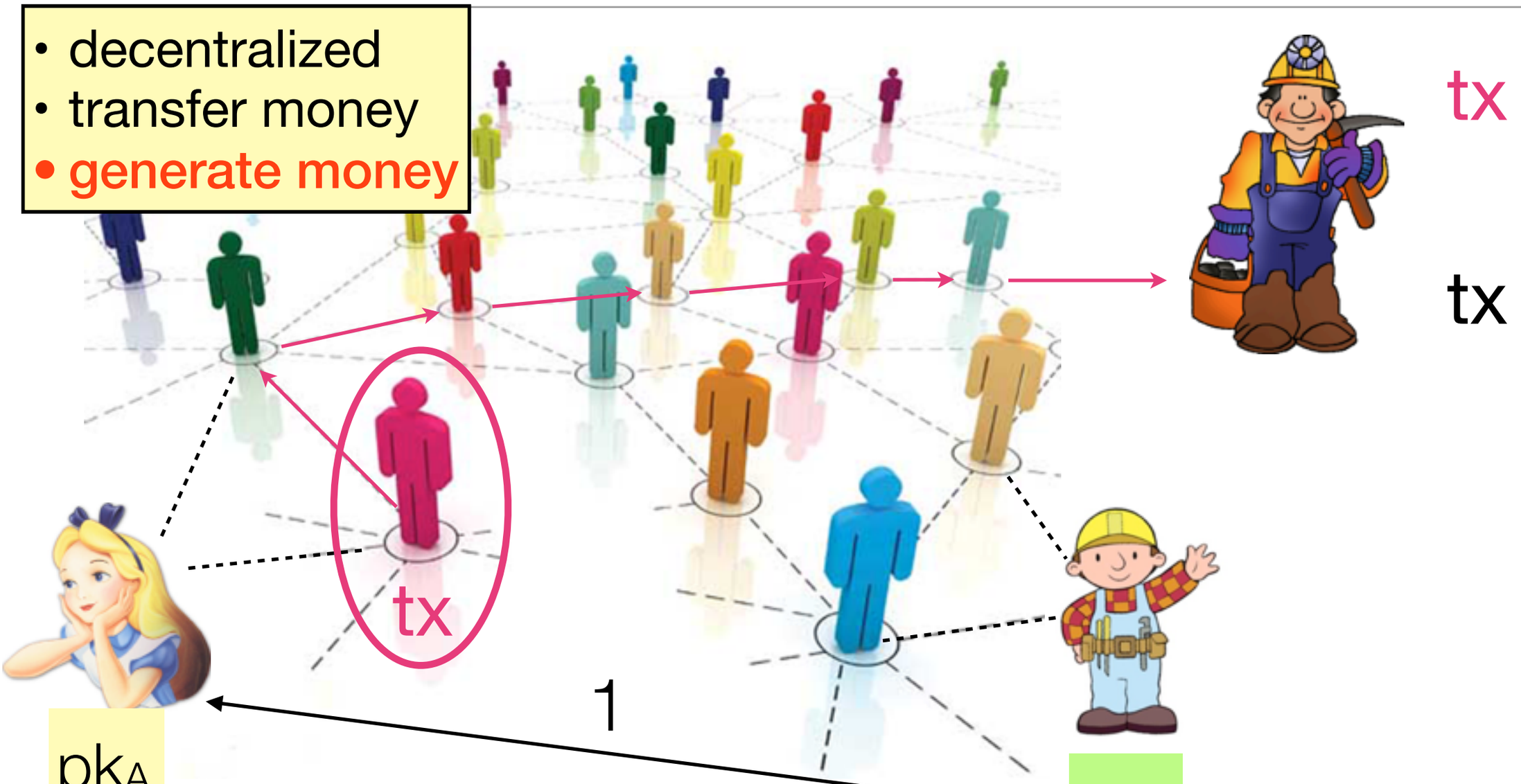
$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

public key
address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



$m = (pk_A, pk_B, 1)$

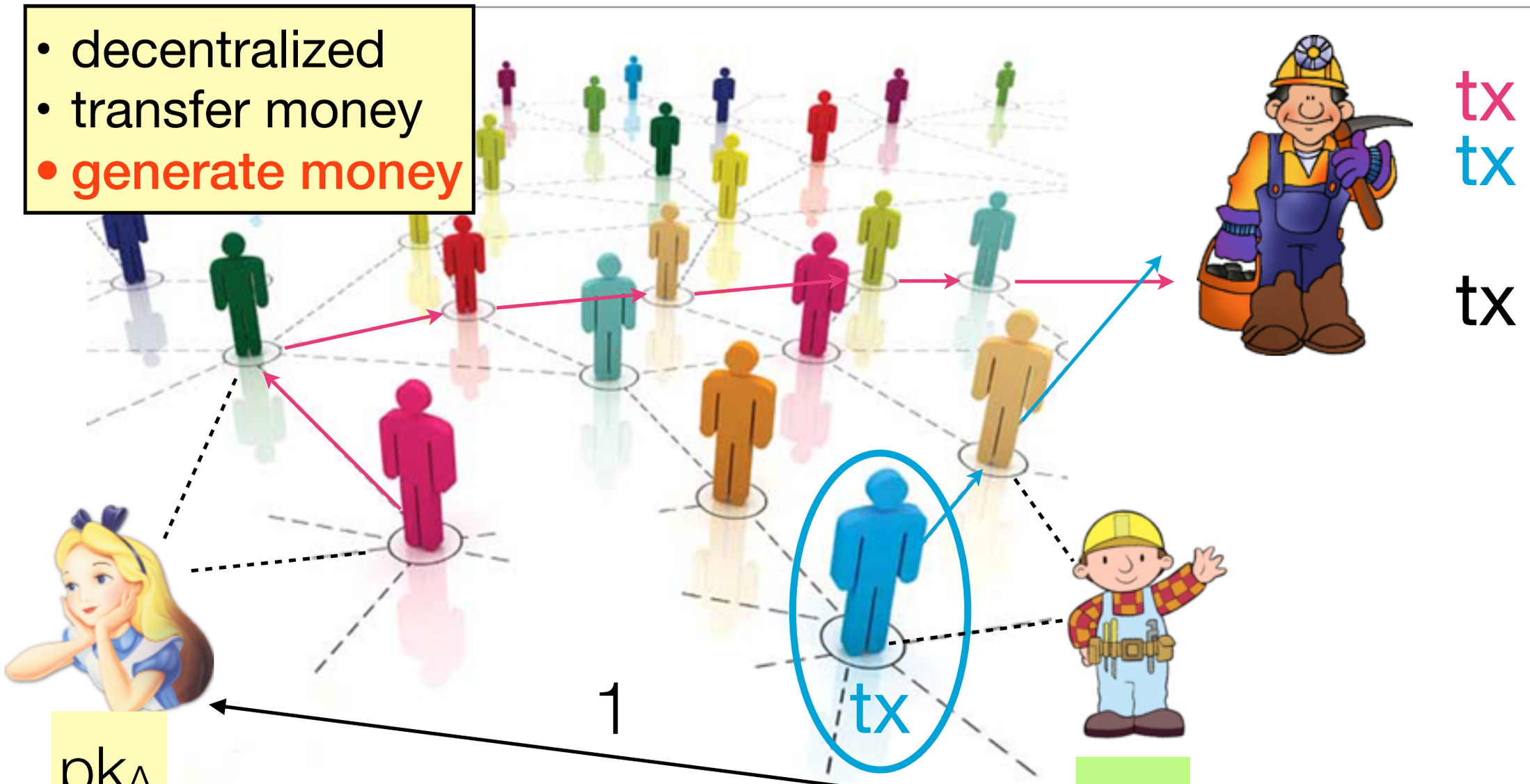
$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

public key
address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

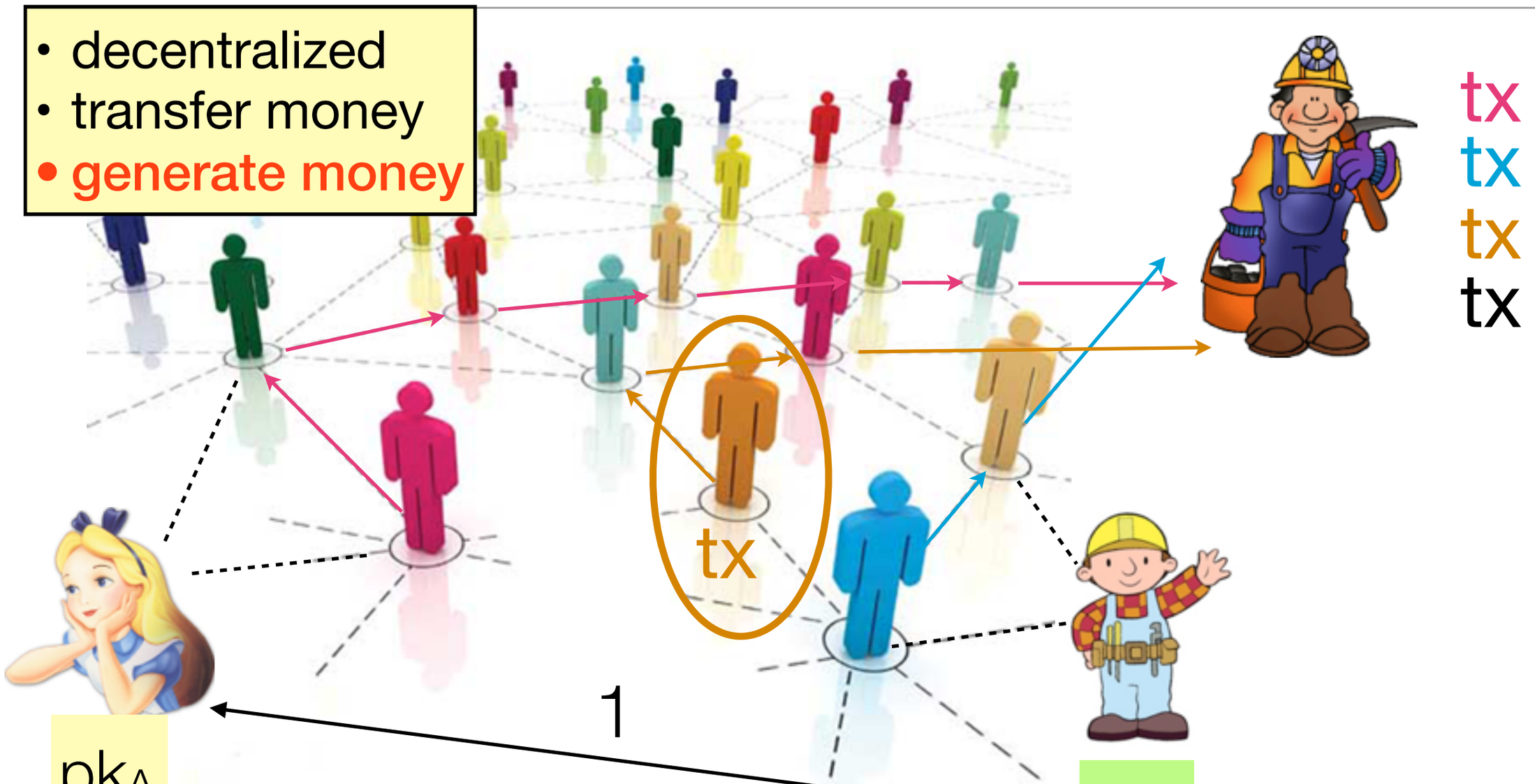
pk_B

public key

address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



$m = (pk_A, pk_B, 1)$

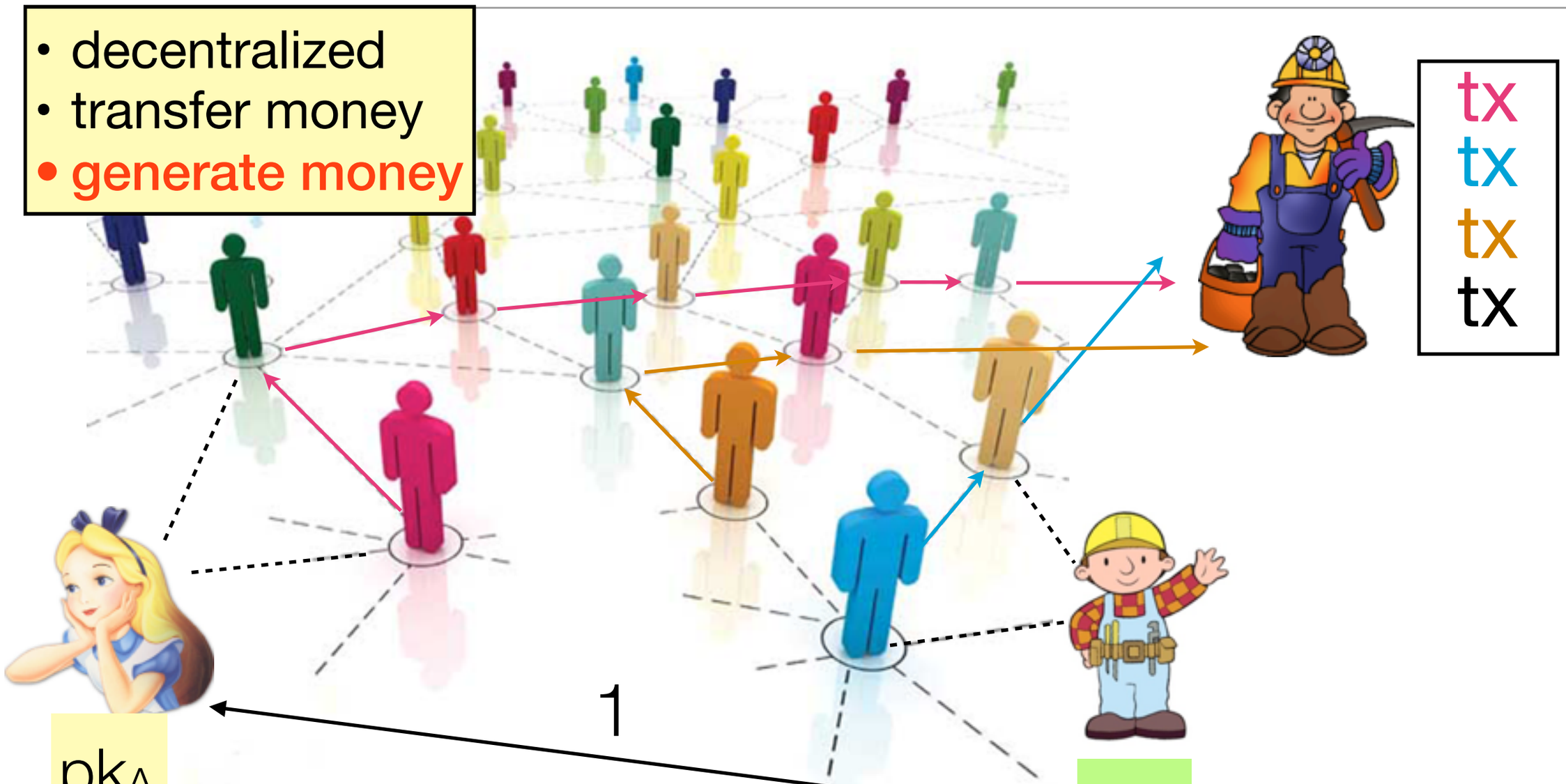
$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

public key
address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

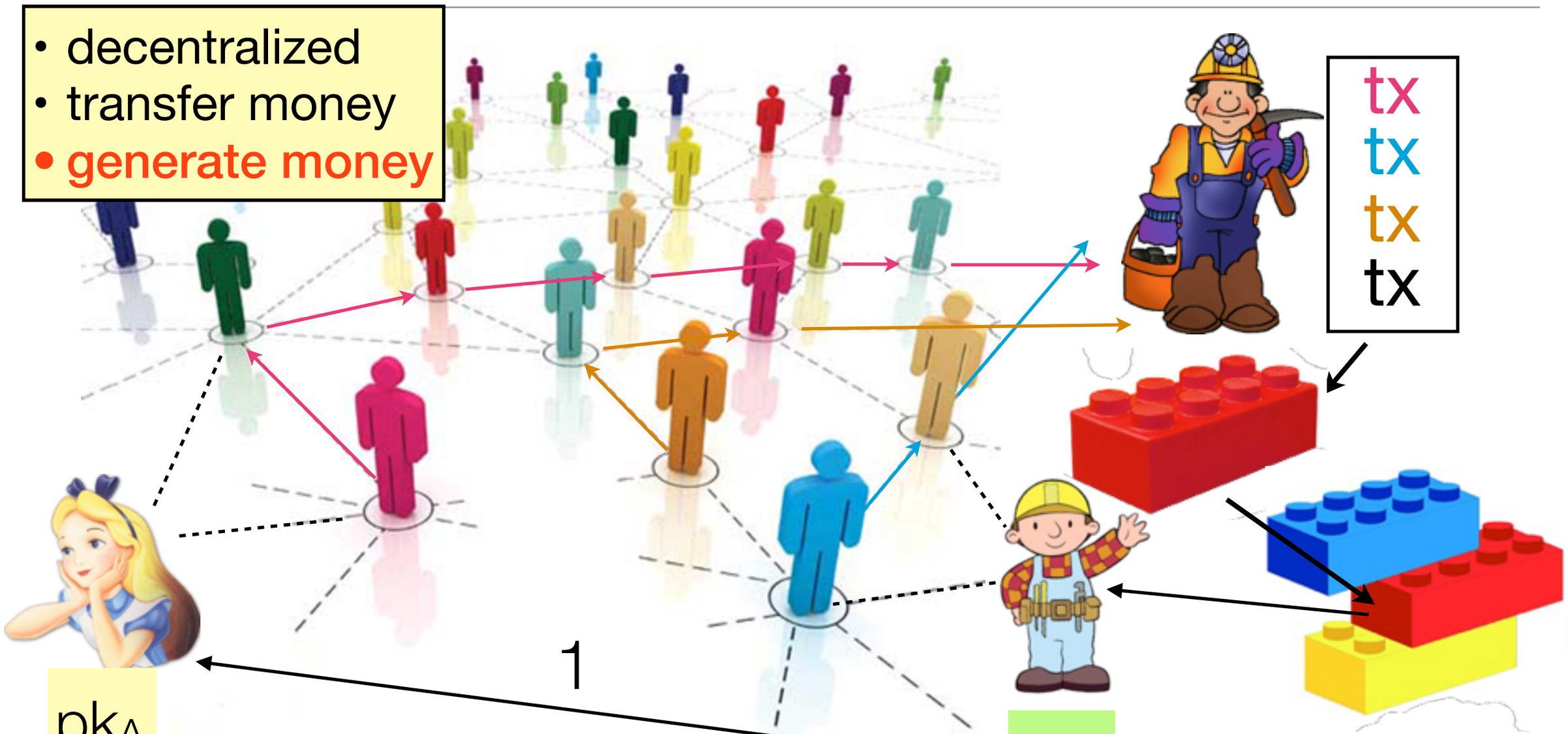
pk_B

public key

address

How does Bitcoin work?

- decentralized
- transfer money
- generate money



pk_A

$m = (pk_A, pk_B, 1)$

$\sigma = Sign(sk_A, m)$

$tx = (m, \sigma)$

pk_B

public key

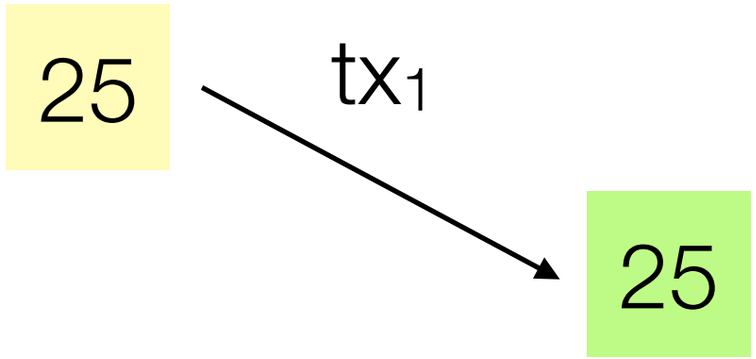
address

How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- **prevent double-spending**

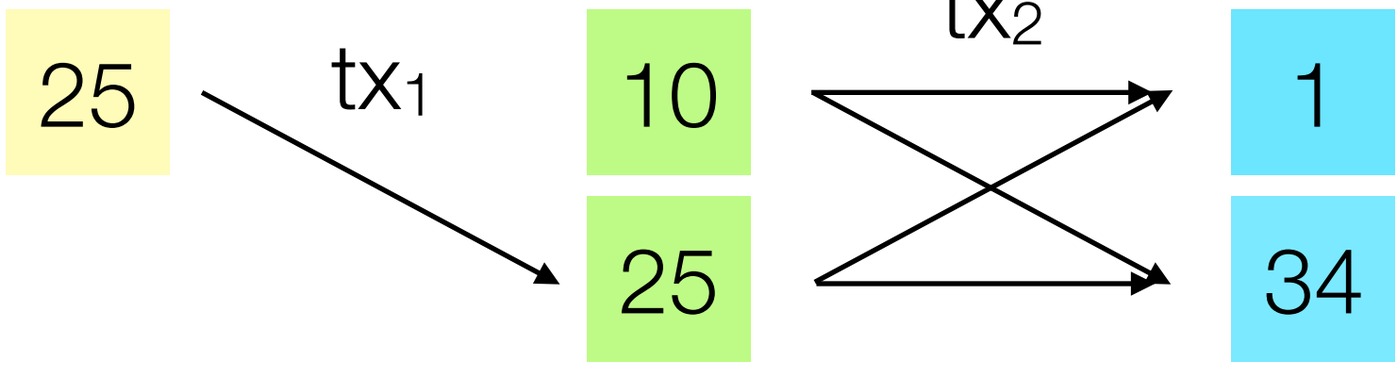
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- **prevent double-spending**



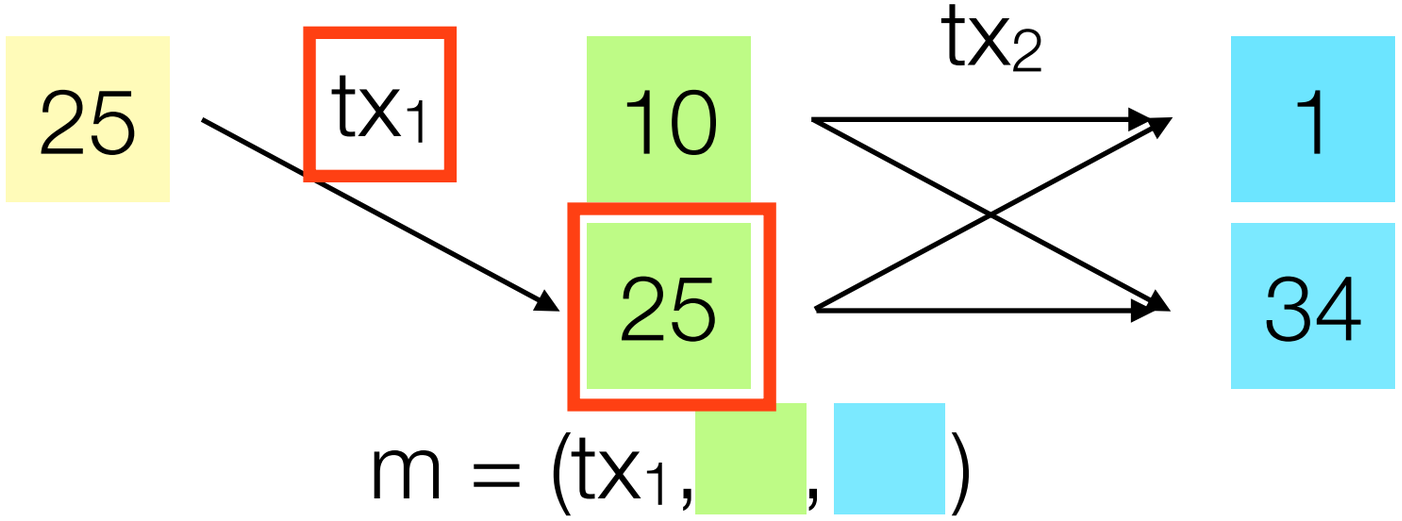
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- **prevent double-spending**



How do bitcoins get spent?

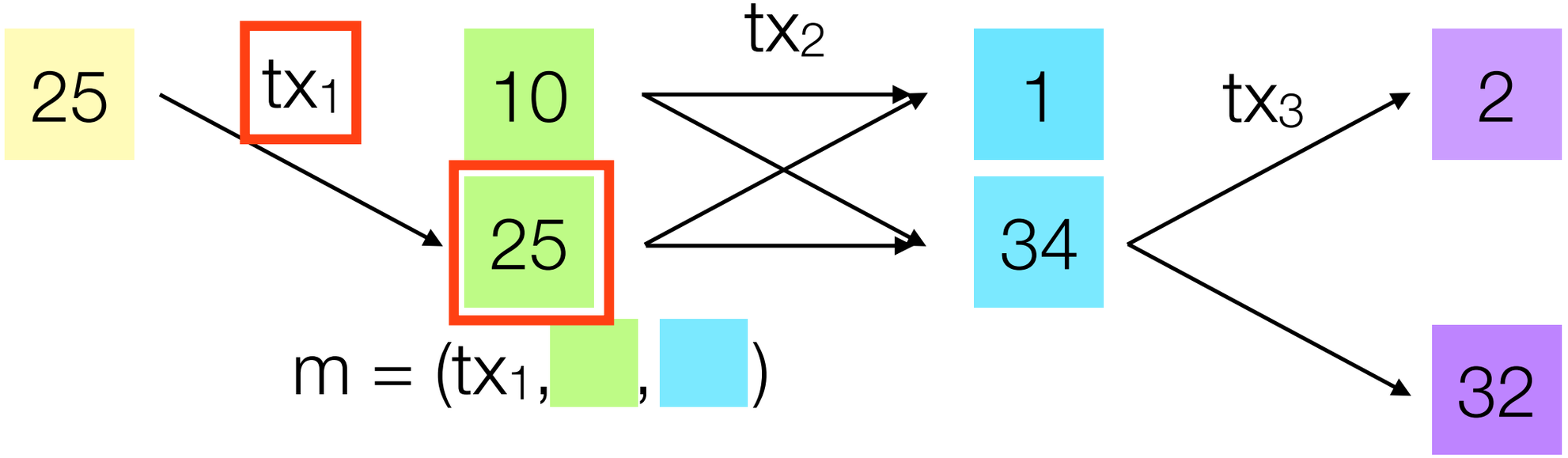
- decentralized
- transfer money
- generate money
- **prevent double-spending**



To spend bitcoins, a user must indicate the previous transaction

How do bitcoins get spent?

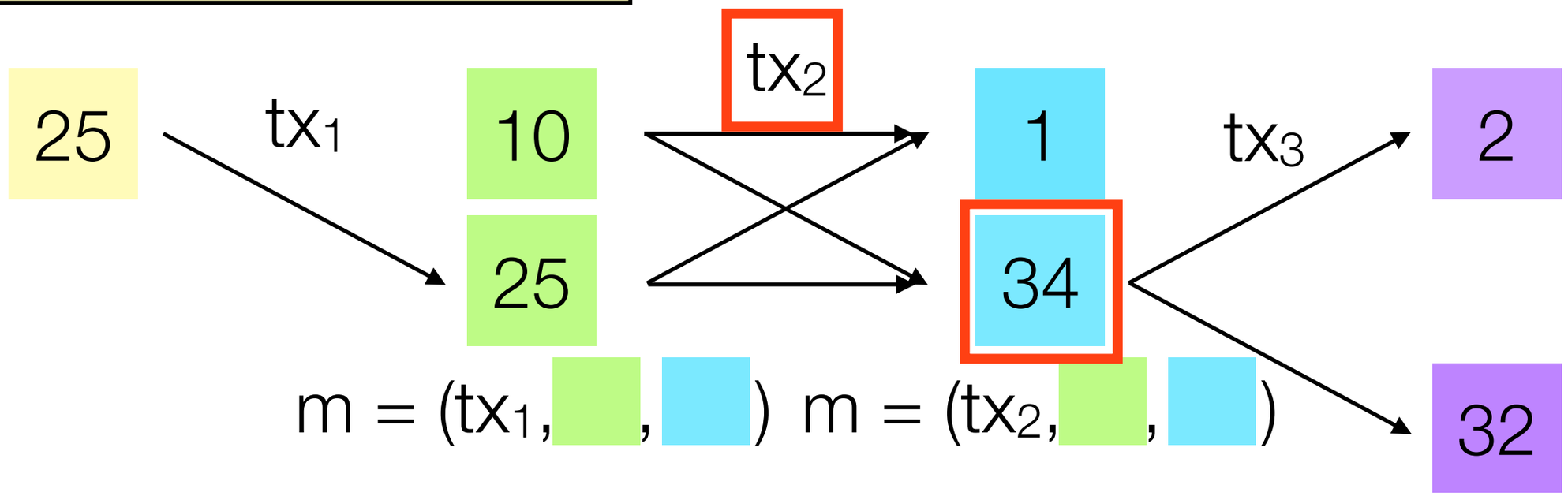
- decentralized
- transfer money
- generate money
- **prevent double-spending**



To spend bitcoins, a user must indicate the previous transaction

How do bitcoins get spent?

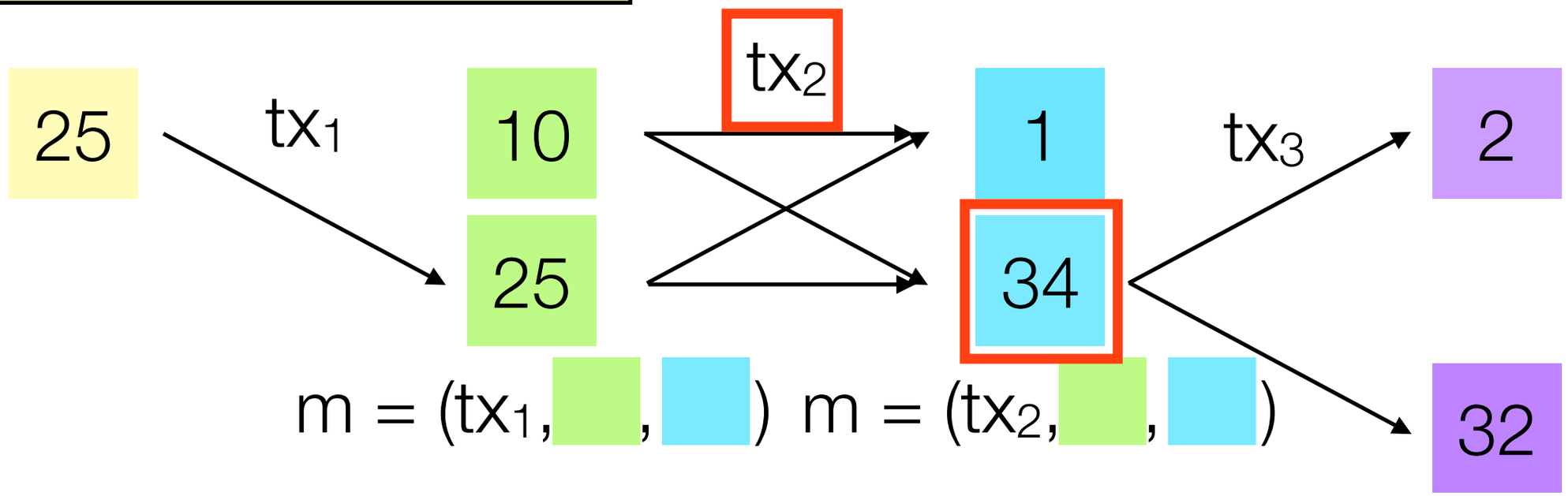
- decentralized
- transfer money
- generate money
- **prevent double-spending**



To spend bitcoins, a user must indicate the previous transaction

How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- **prevent double-spending**



To spend bitcoins, a user must indicate the previous transaction

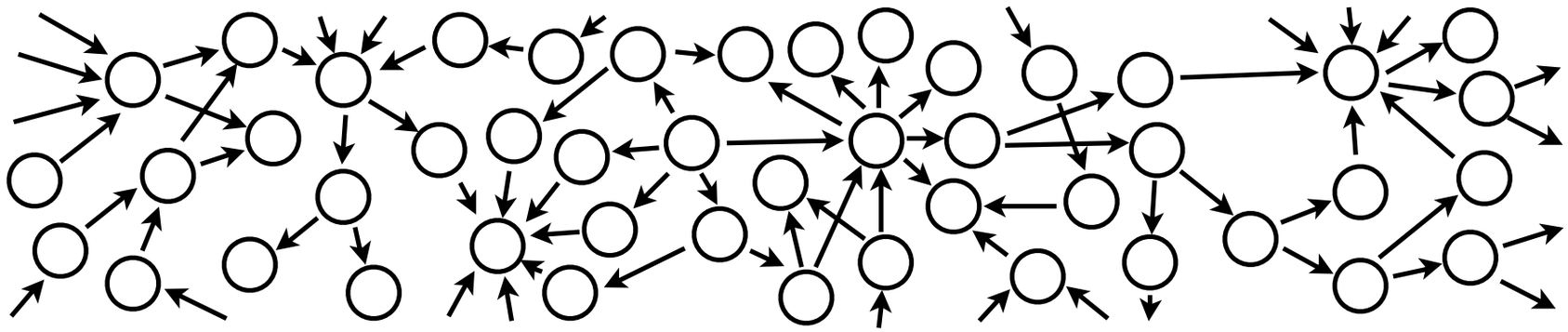
All bitcoins received in a transaction must be spent all at once

How to identify users? [**M**PGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys

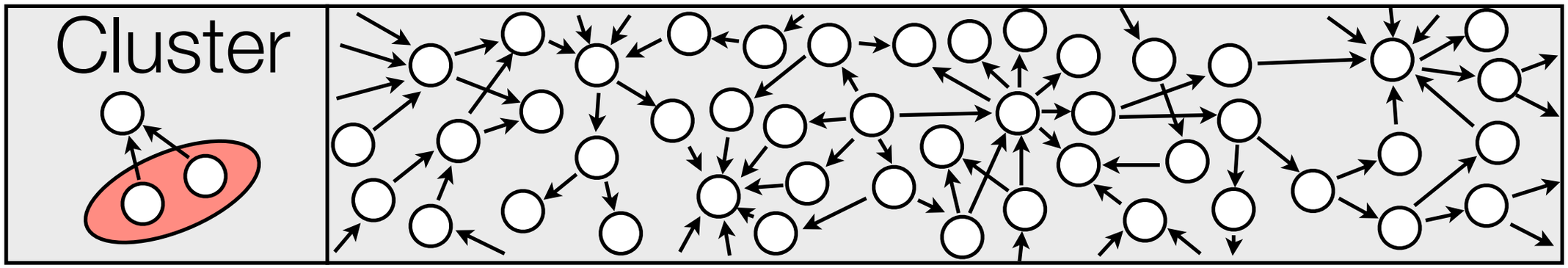
How to identify users? [**M**PGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys



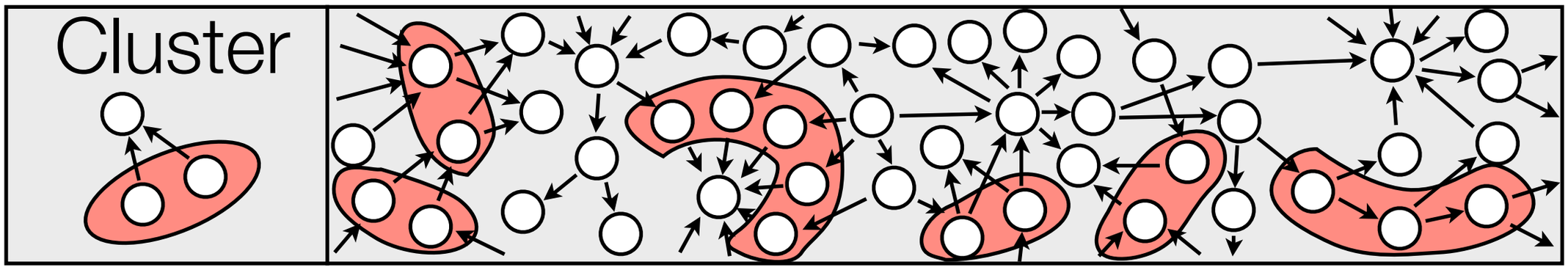
How to identify users? [**M**PGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys



How to identify users? [**M**PGLMVS IMC13]

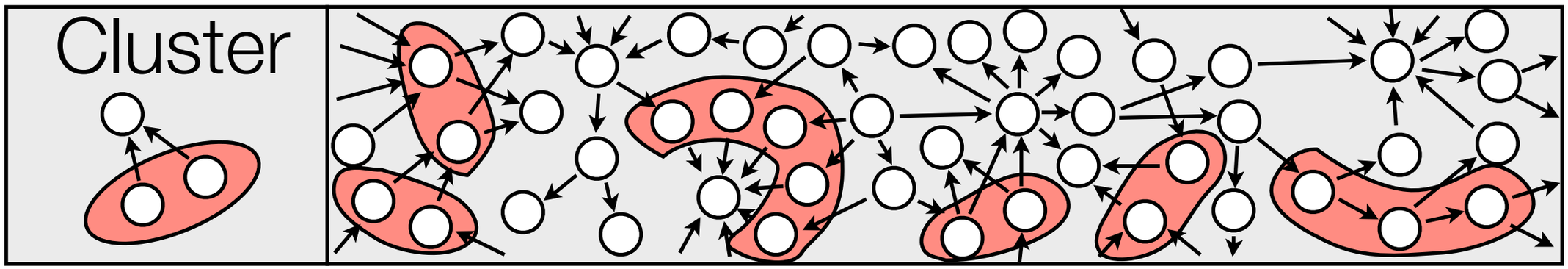
Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys



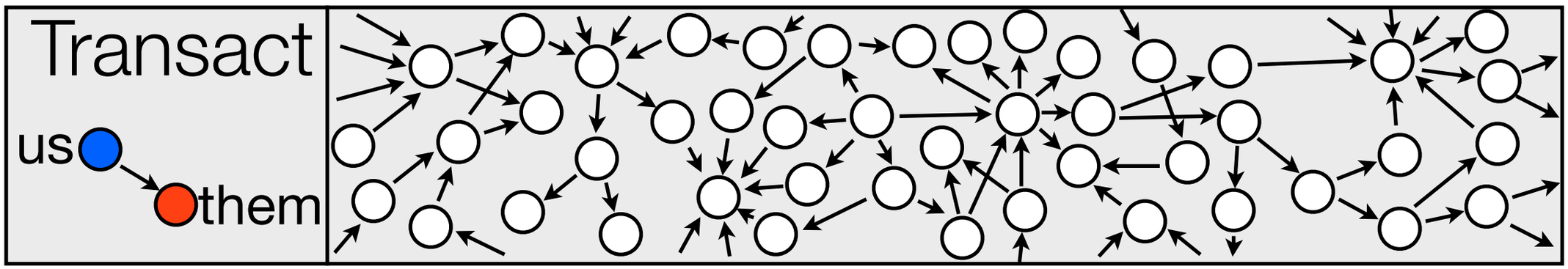
Collapse into a more manageable graph of clusters of public keys representing distinct entities

How to identify users? [**M**PGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys

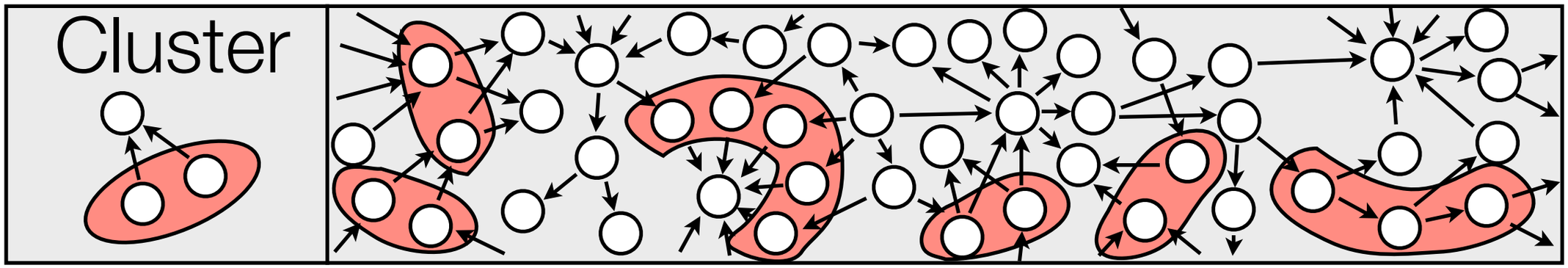


Collapse into a more manageable graph of clusters of public keys representing distinct entities

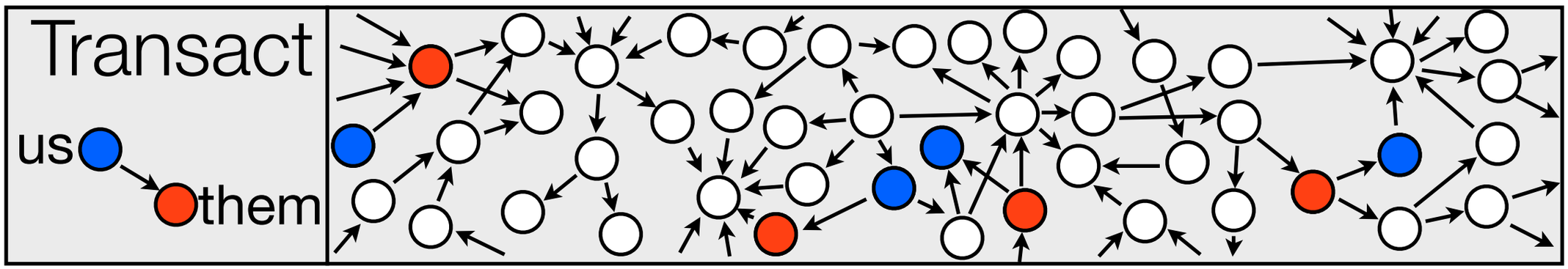


How to identify users? [**M**PGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys

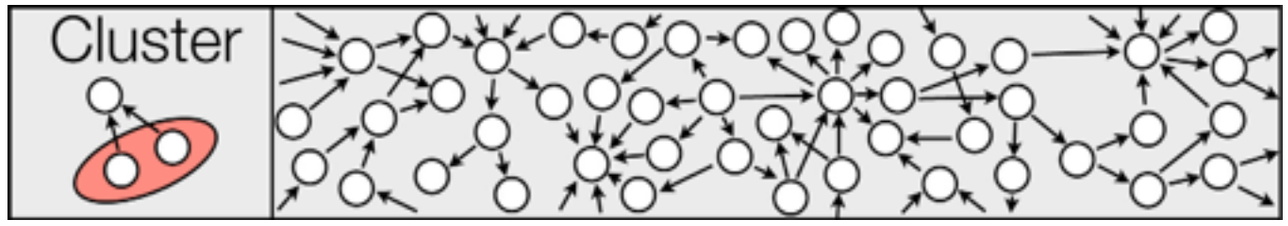


Collapse into a more manageable graph of clusters of public keys representing distinct entities

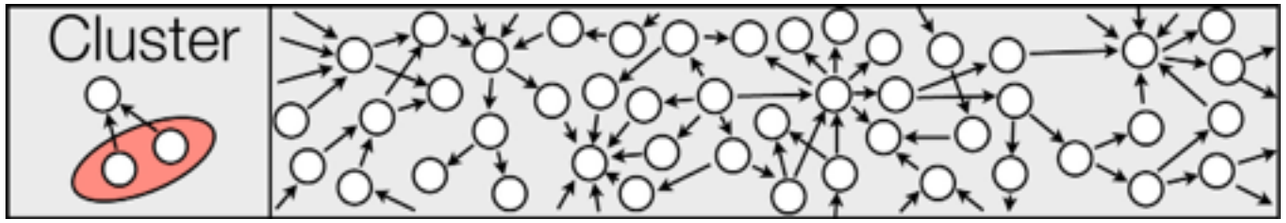


Collect ground truth data by participating in transactions

Clustering by inputs



Clustering by inputs



1

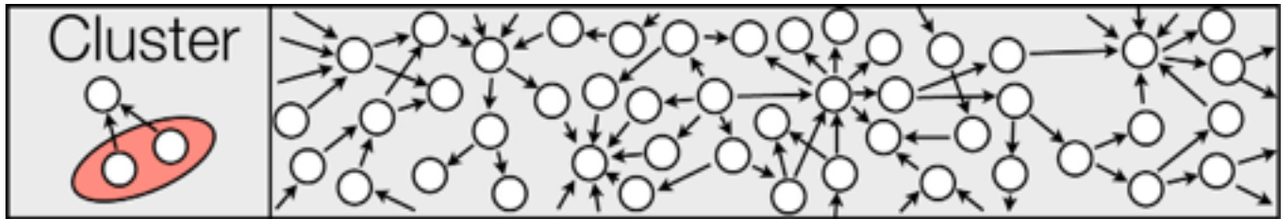
2

3

7

15

Clustering by inputs



1

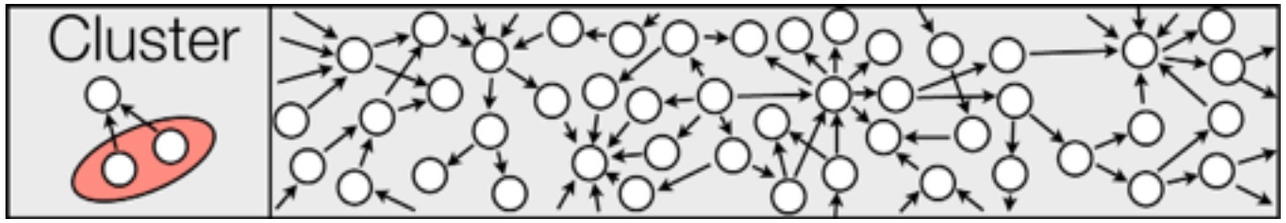
2

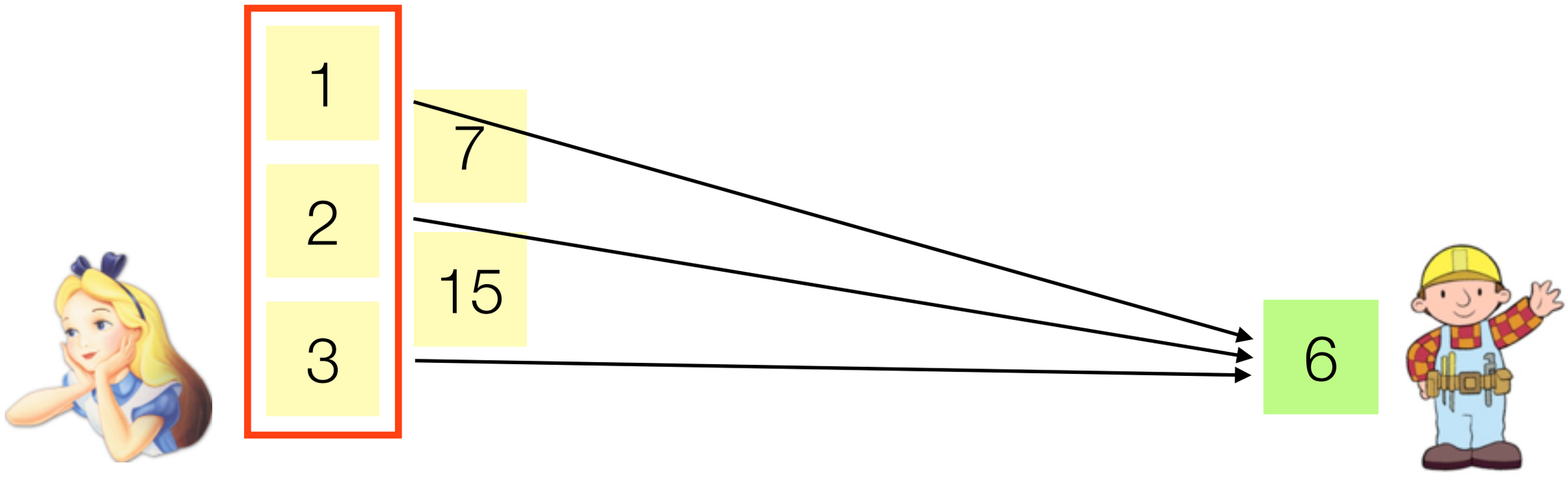
3

7

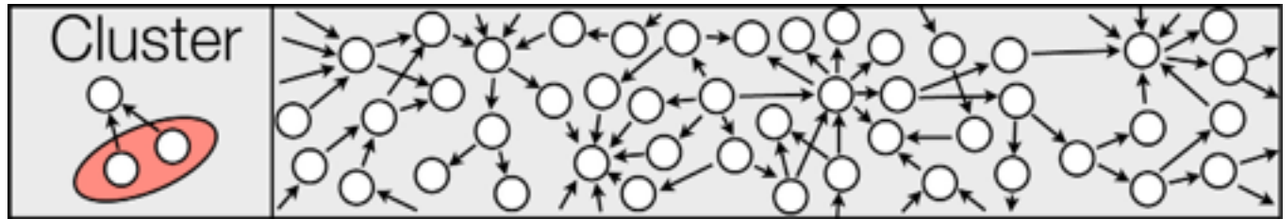
15

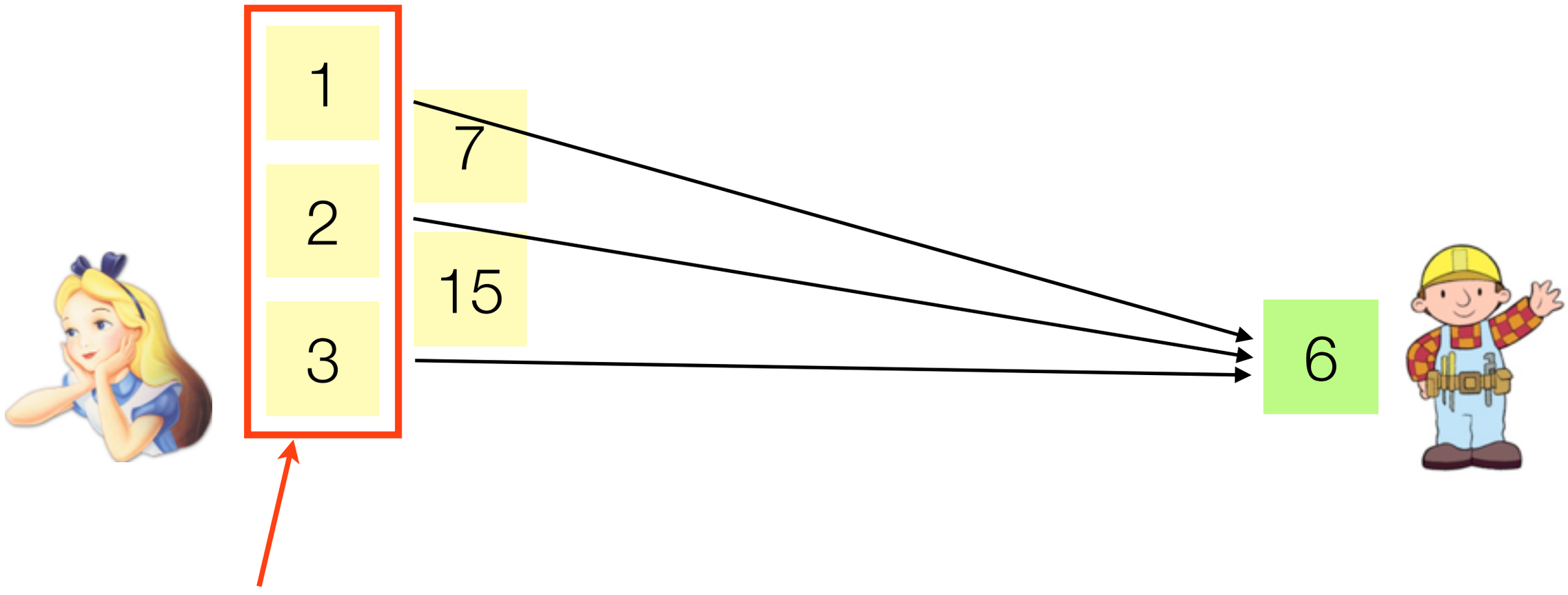
Clustering by inputs





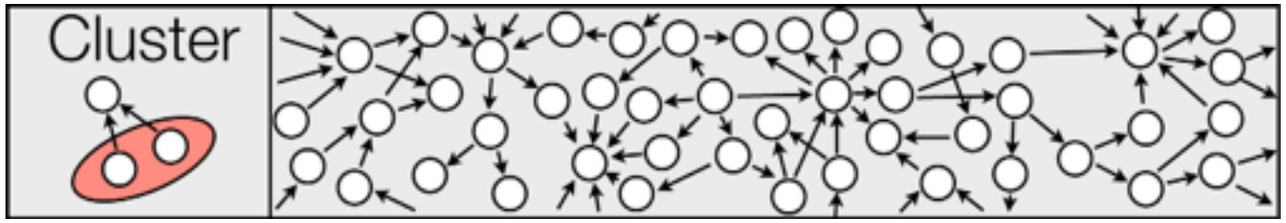
Clustering by inputs





Heuristic #1: the same user controls these addresses [N08,RH11,RS13,A+13]

Change addresses



1

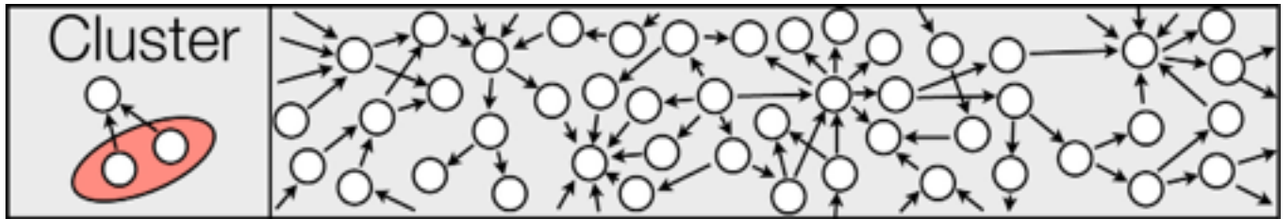
2

3

7

15

Change addresses



1

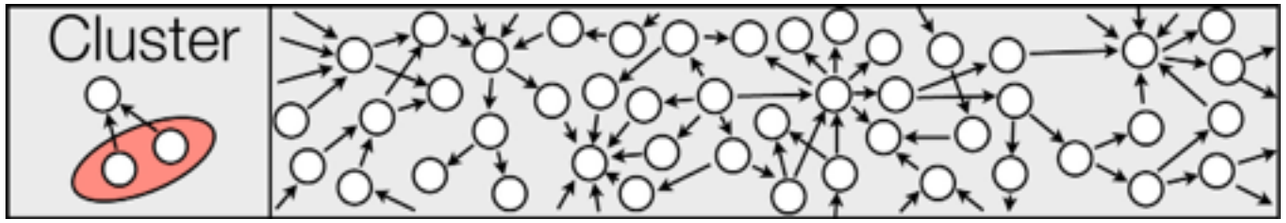
2

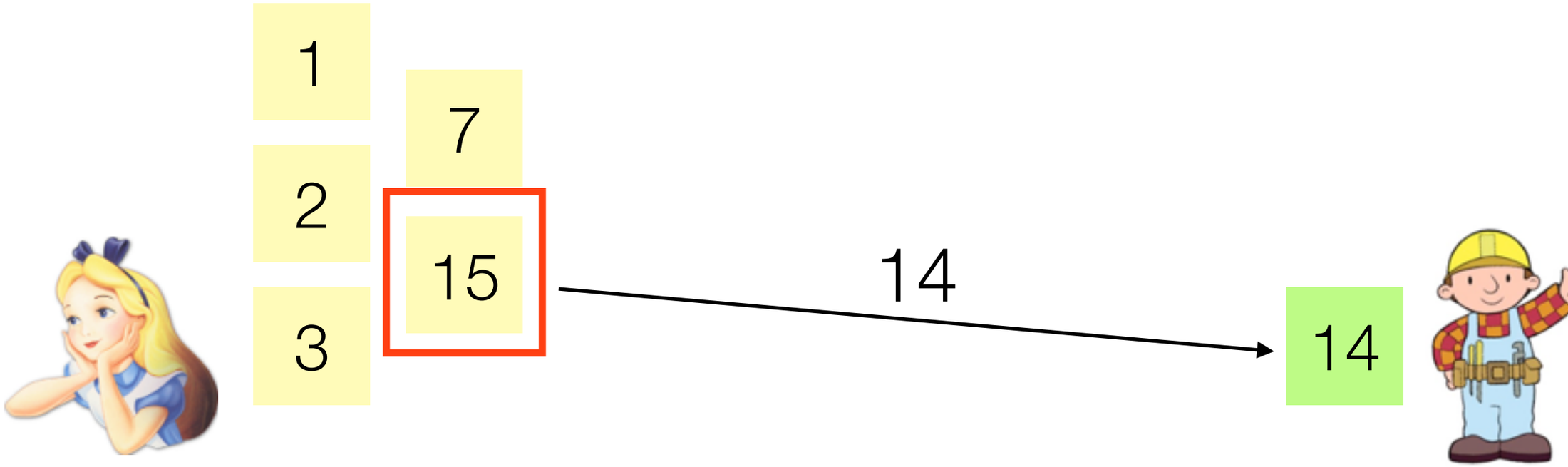
3

7

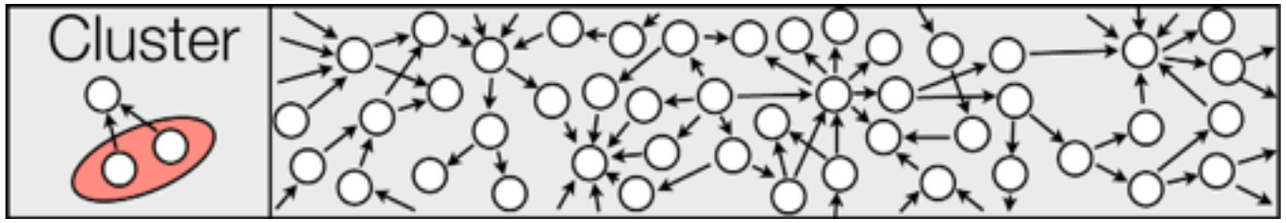
15

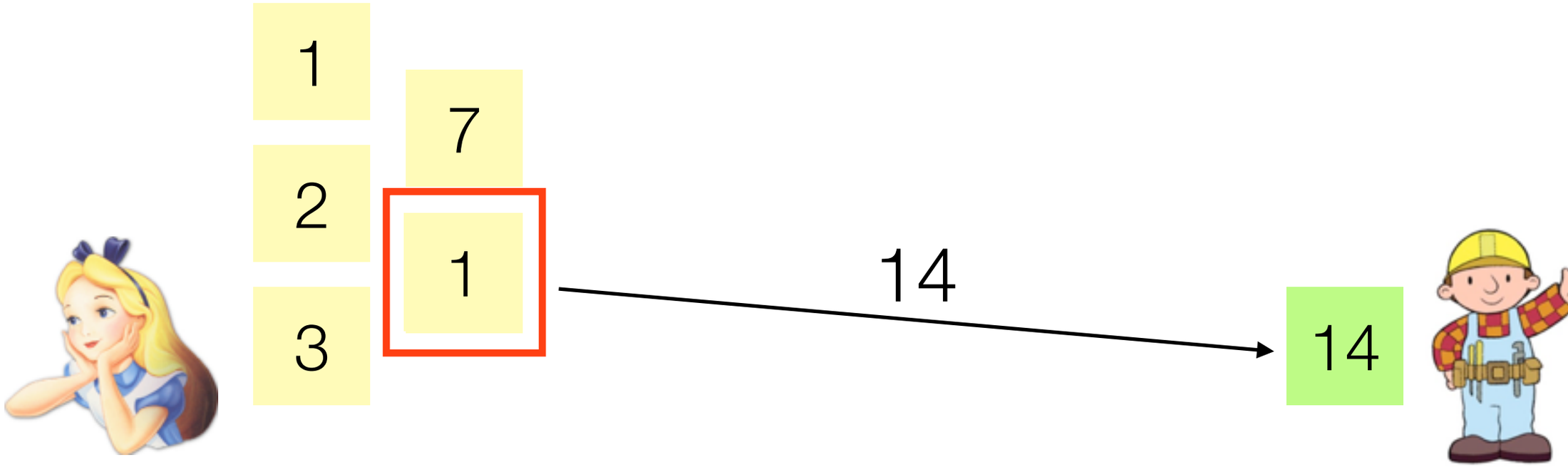
Change addresses



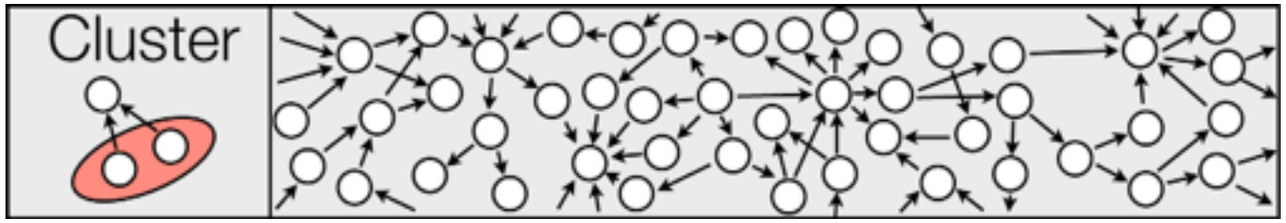


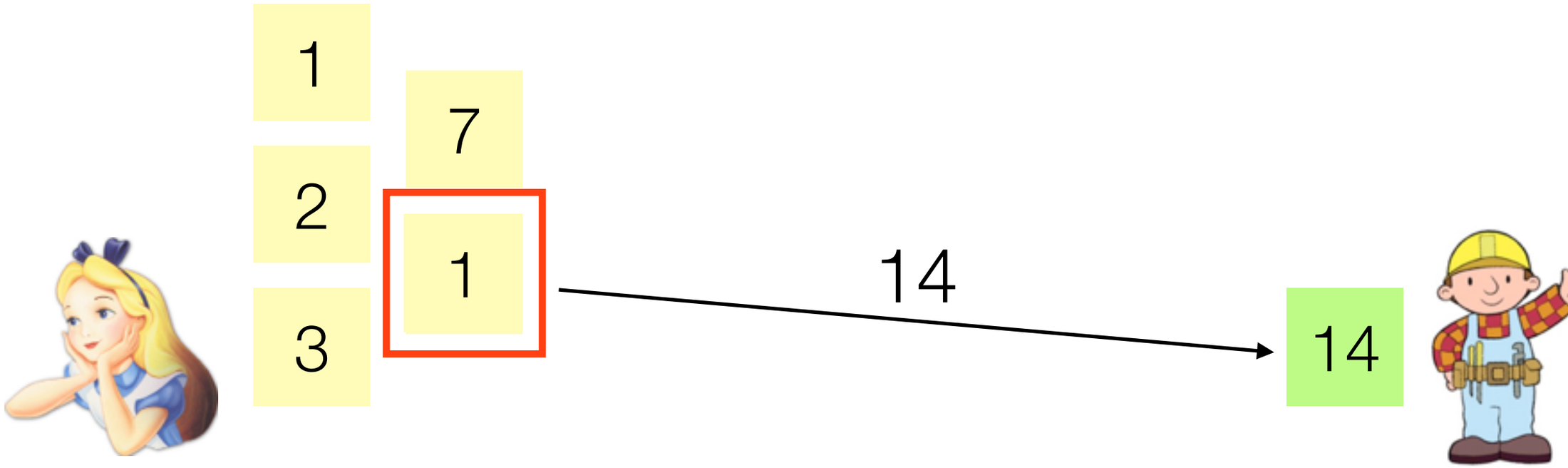
Change addresses





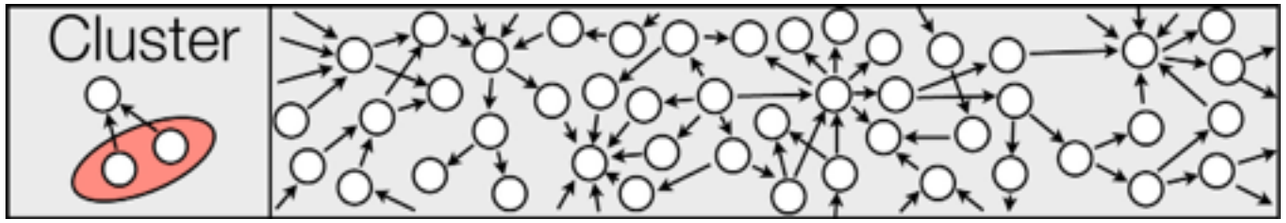
Change addresses





All bitcoins received in a transaction must be spent all at once

Change addresses



1

2

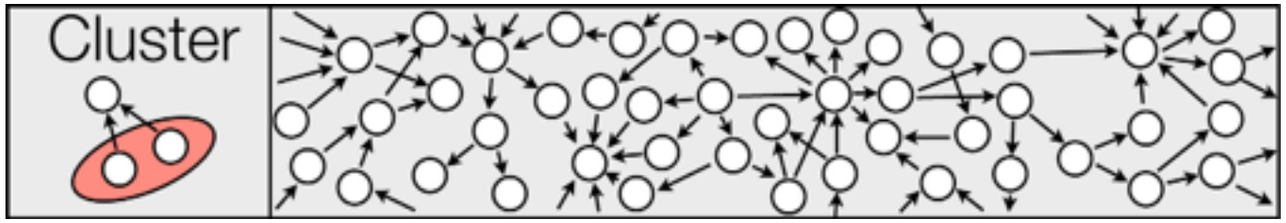
3

7

15

All bitcoins received in a transaction must be spent all at once

Change addresses



1

2

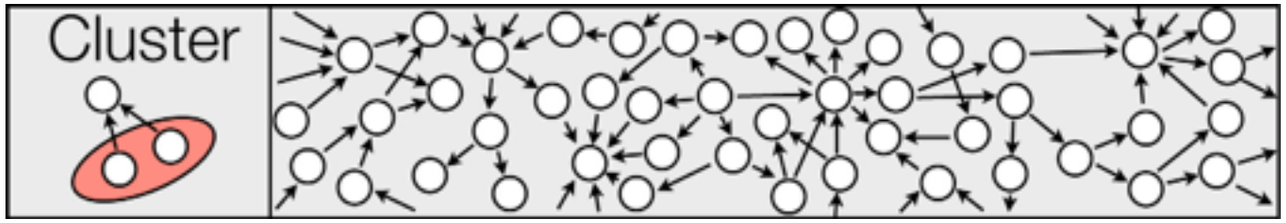
3

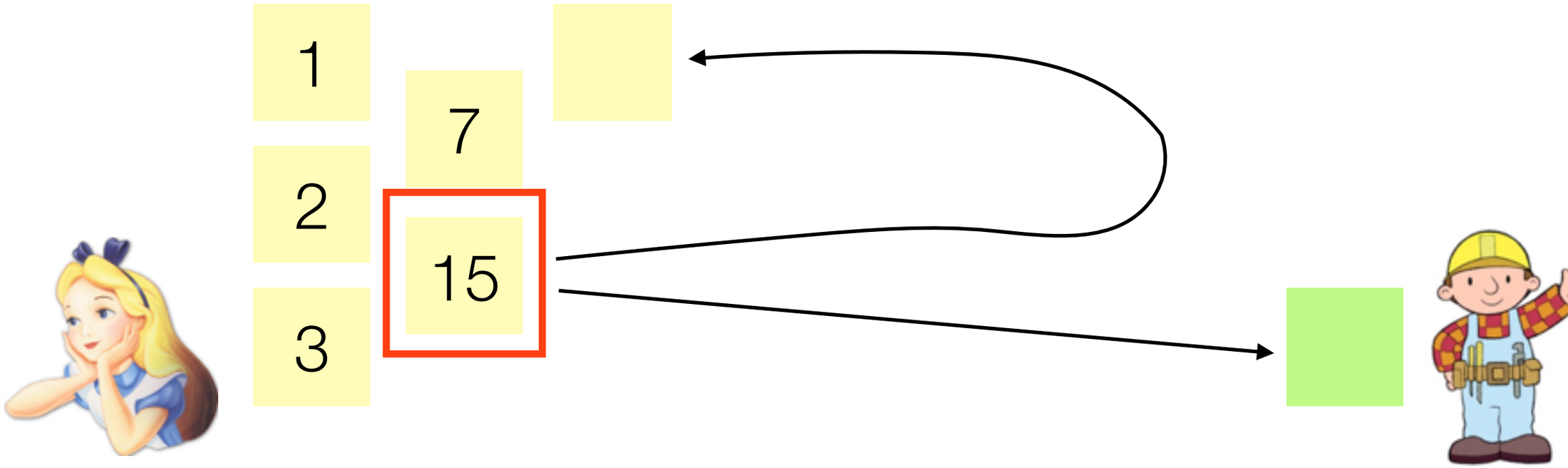
7

15

All bitcoins received in a transaction must be spent all at once

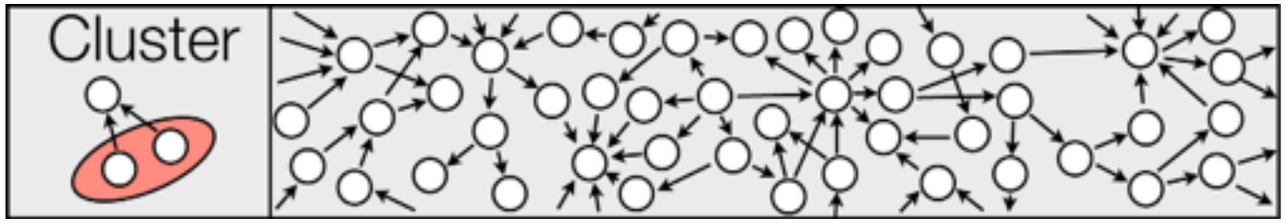
Change addresses

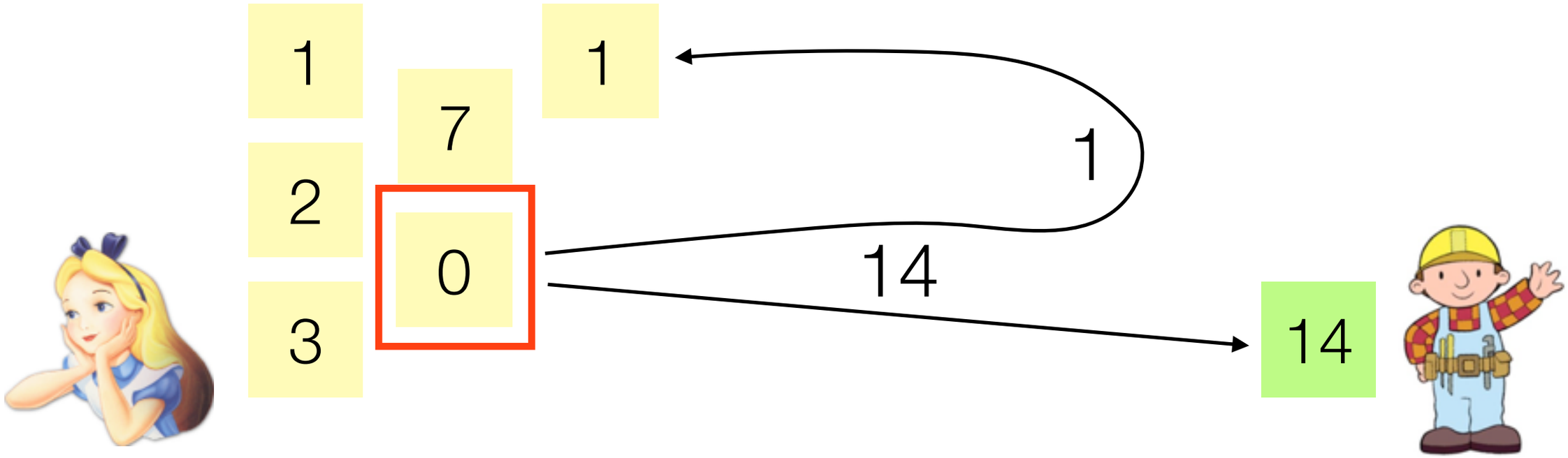




All bitcoins received in a transaction must be spent all at once

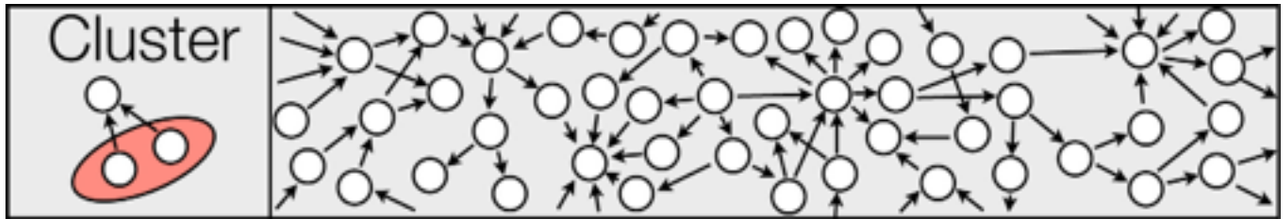
Change addresses

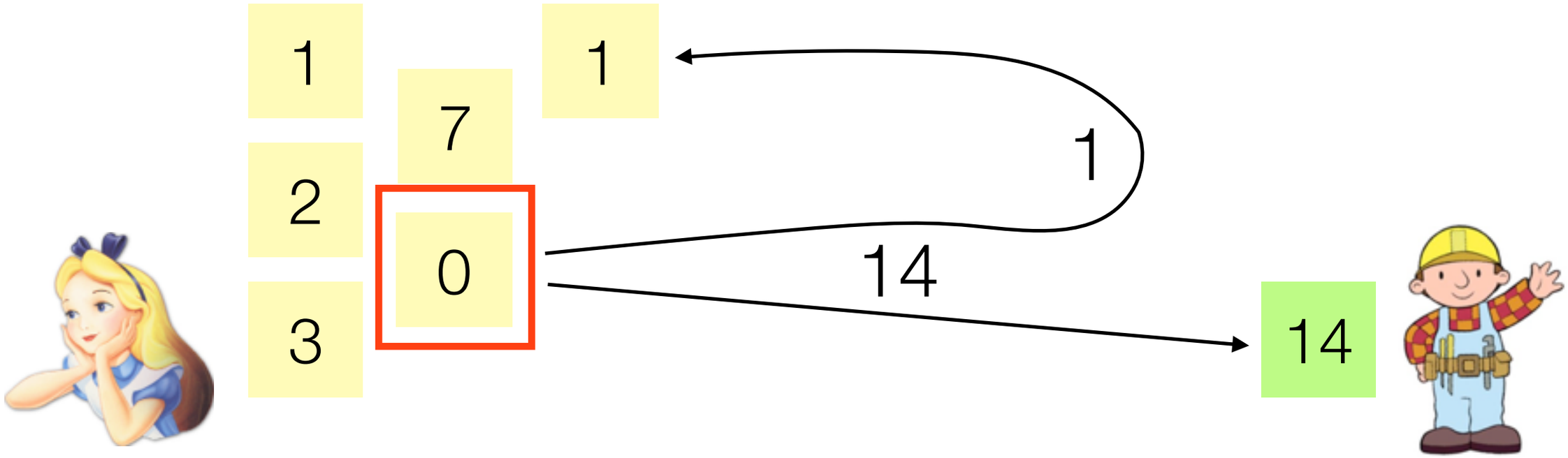




All bitcoins received in a transaction must be spent all at once

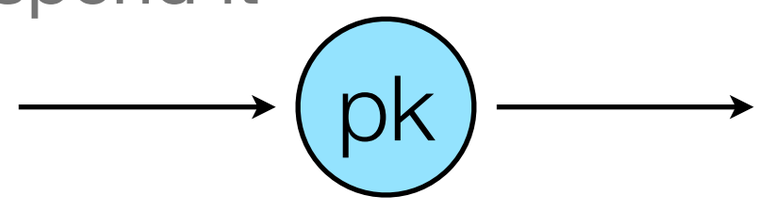
Change addresses



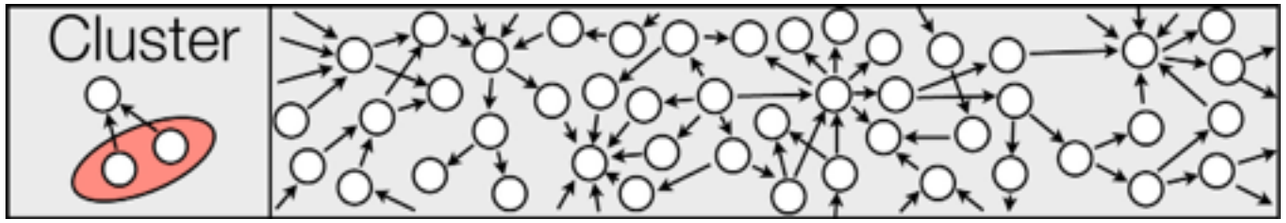


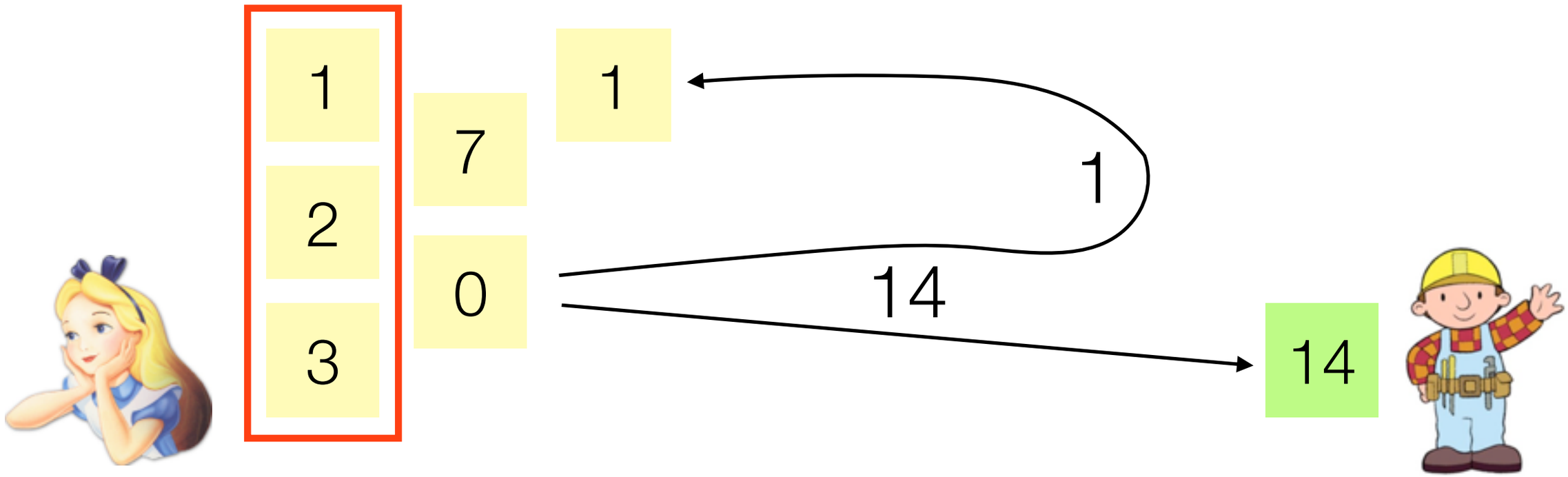
All bitcoins received in a transaction must be spent all at once

In the standard idiom, change addresses are used at most twice: to receive change and to spend it

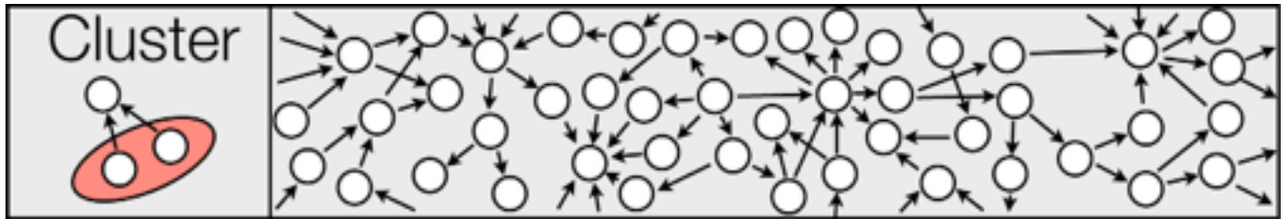


Clustering by change





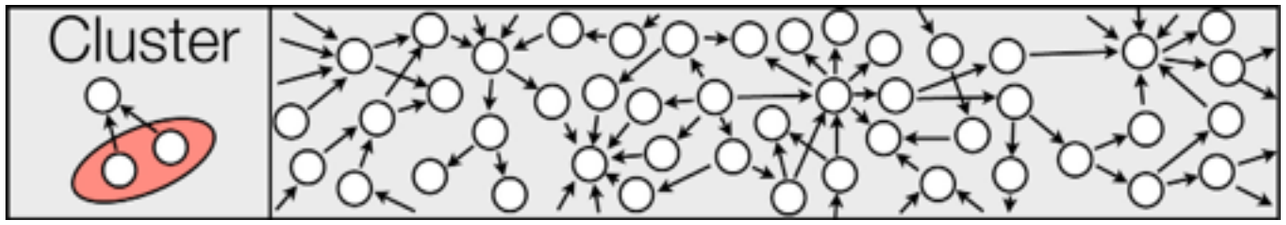
Clustering by change



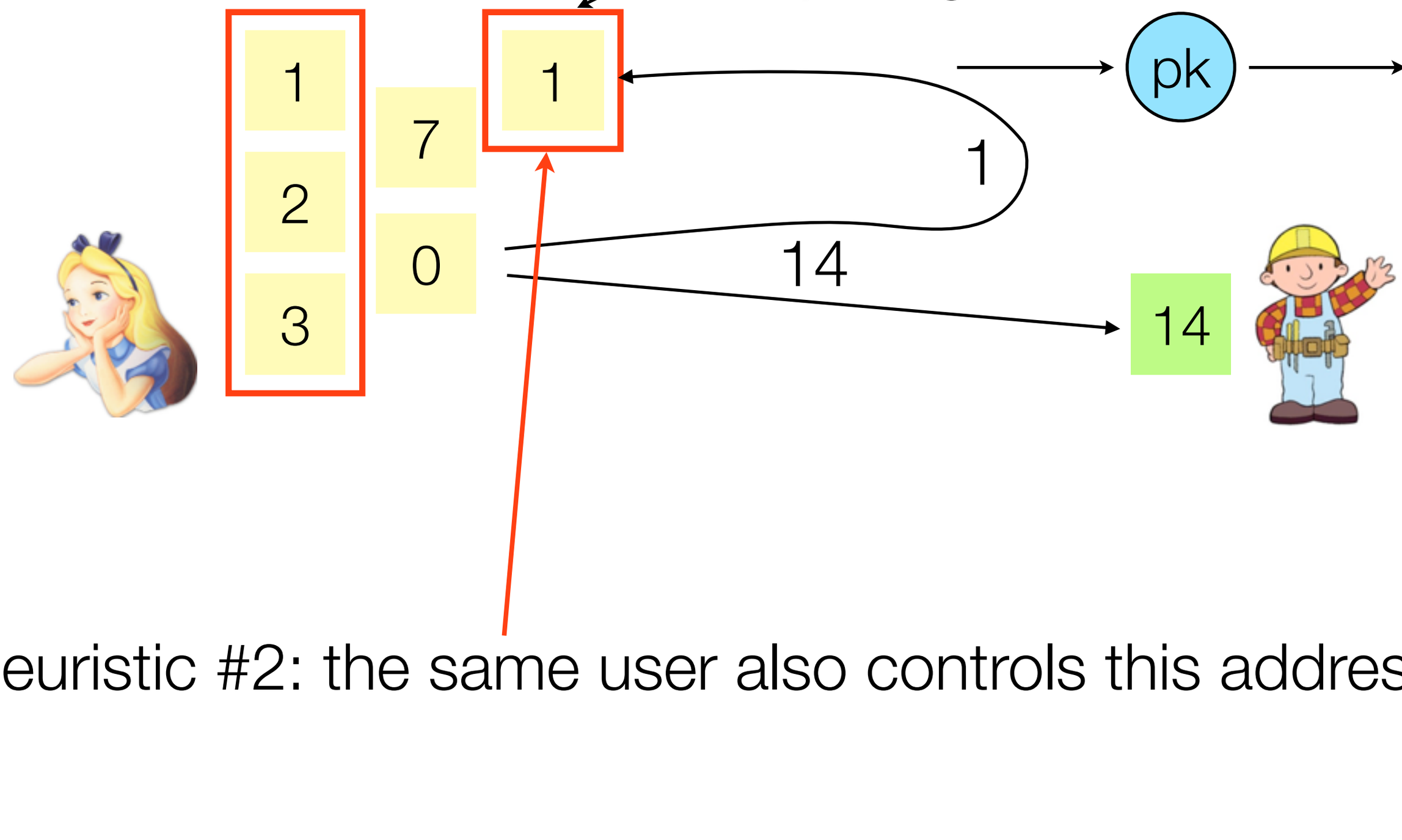
identify using one-time behavior

1
2
3
7
0
1
1
pk
14
14

Clustering by change

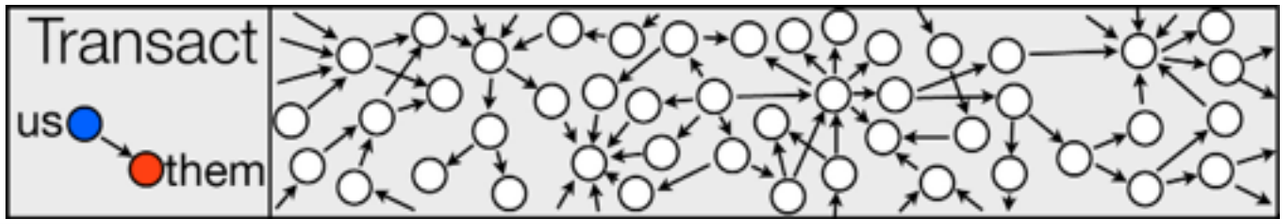


identify using one-time behavior

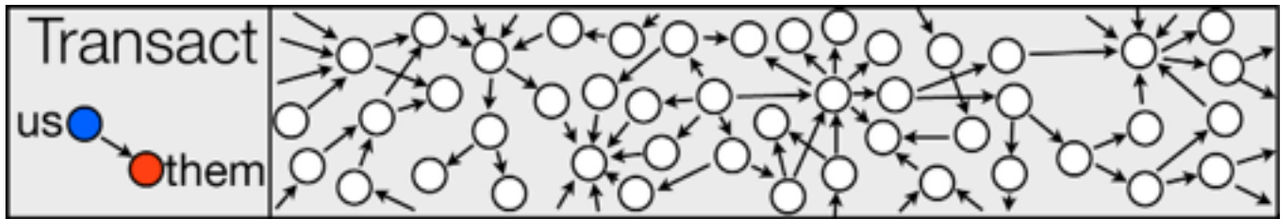


Heuristic #2: the same user also controls this address

Data collection

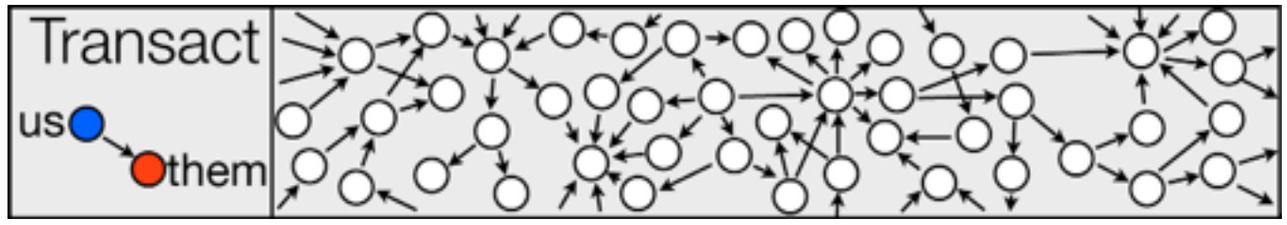


Data collection



Engaged in transactions with:

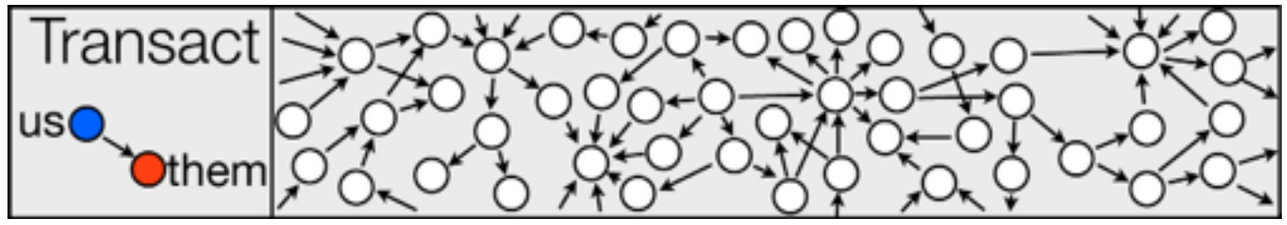
Data collection



Engaged in transactions with:

- Exchanges

Data collection

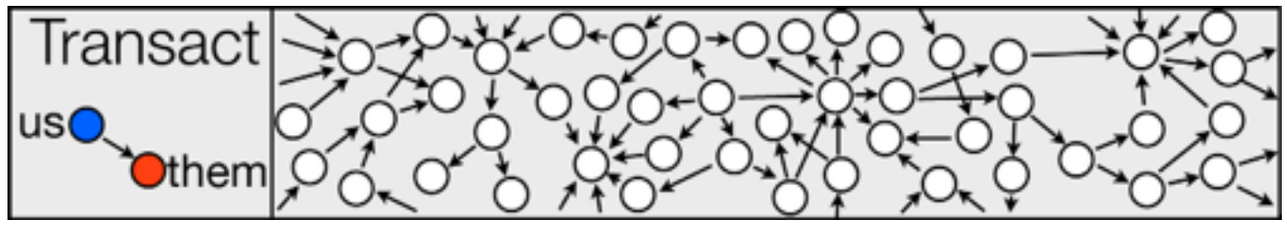


Engaged in transactions with:

- Exchanges

- Vendors

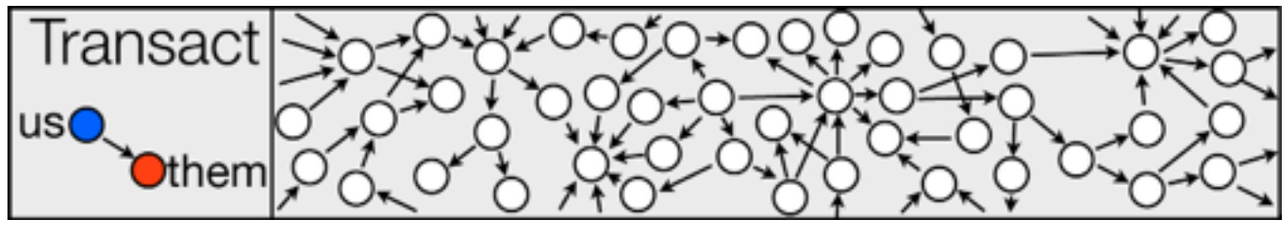
Data collection



Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools

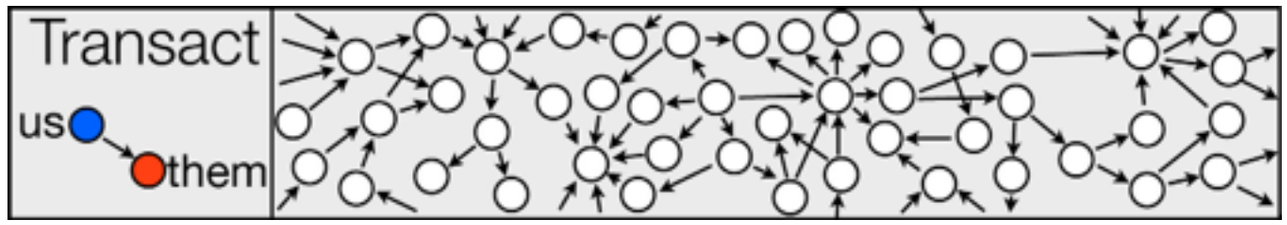
Data collection



Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools
- Gambling sites

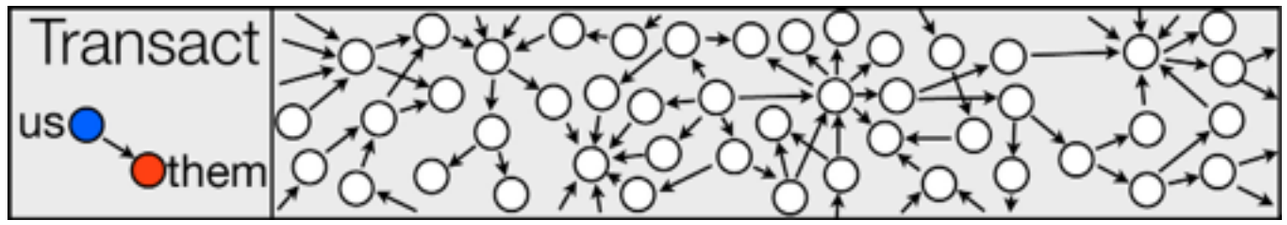
Data collection



Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools
- Gambling sites
- Wallet services

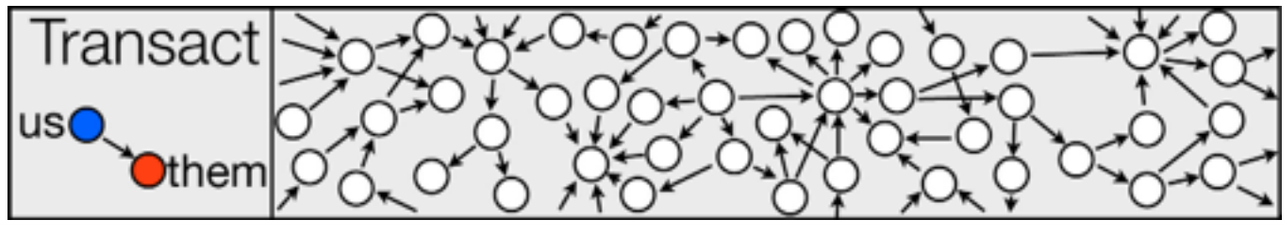
Data collection



Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools
- Gambling sites
- Wallet services
- Mix services

Data collection

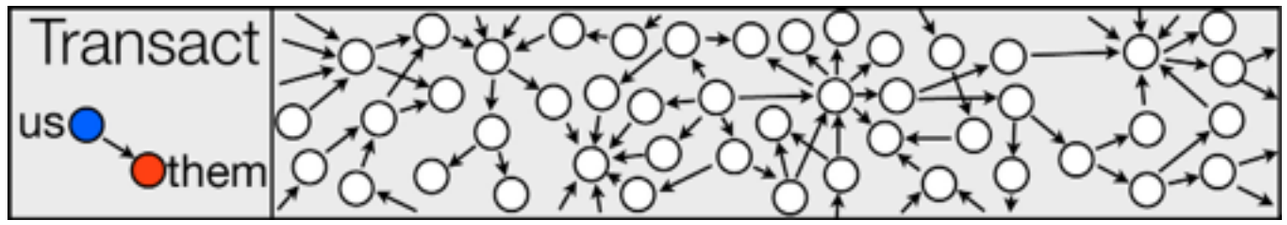


Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools
- Gambling sites
- Wallet services
- Mix services

Scraped published tags

Data collection



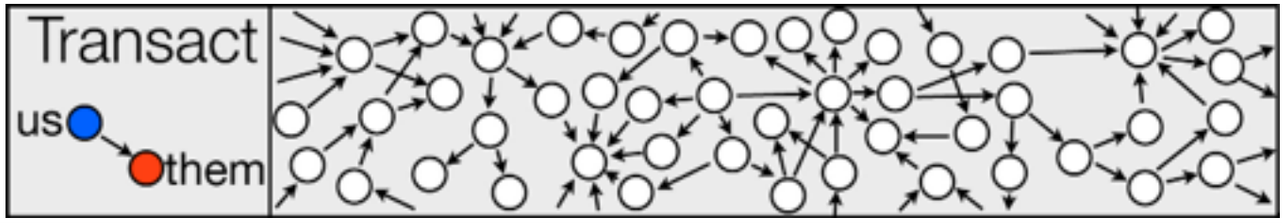
Engaged in transactions with:

- Exchanges
- Vendors
- Mining pools
- Gambling sites
- Wallet services
- Mix services

Scraped published tags

Found addresses discussed on forums

Exchanges



Bitcoin-24

BTC e

aurum exchange

Bitcoin-Central

CAMP BX
BITCOIN TRADING PLATFORM

BitInstant

bitcoin.de
Bitcoin-Marketplace - Made in Germany!

VirtEx
Canadian Virtual Exchange

BITCOIN NORDIC
Bitcoins easy. Bitcoins fast.

Bitcurex
The new age of currency.

ICBIT
Bitcoin Exchange

btcQuick

bitfloor

mercadobitcoin

FastCash4Bitcoins

BitMarket.eu
YOUR PLACE TO EXCHANGE BITCOINS

MT.GOX

LILION TRANSFER
Login | Registration

BITME

TRADING LTD.
THEROCK

NANAIMO GOLD
DIGITAL CURRENCY EXCHANGE

BITSTAMP

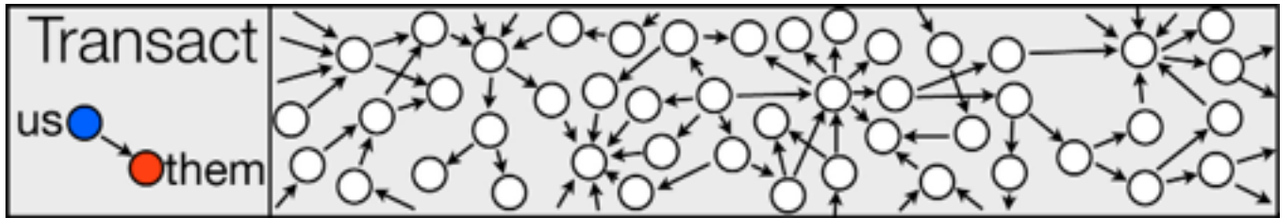
Vircurex

OKPAY payments made easy

Bitcoin China

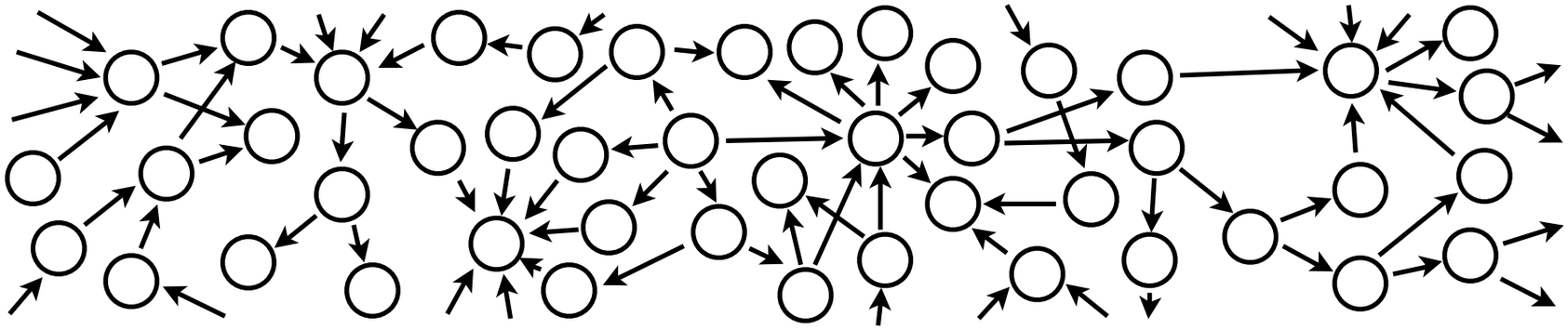
VirWoX
virtual world exchange

Vendors

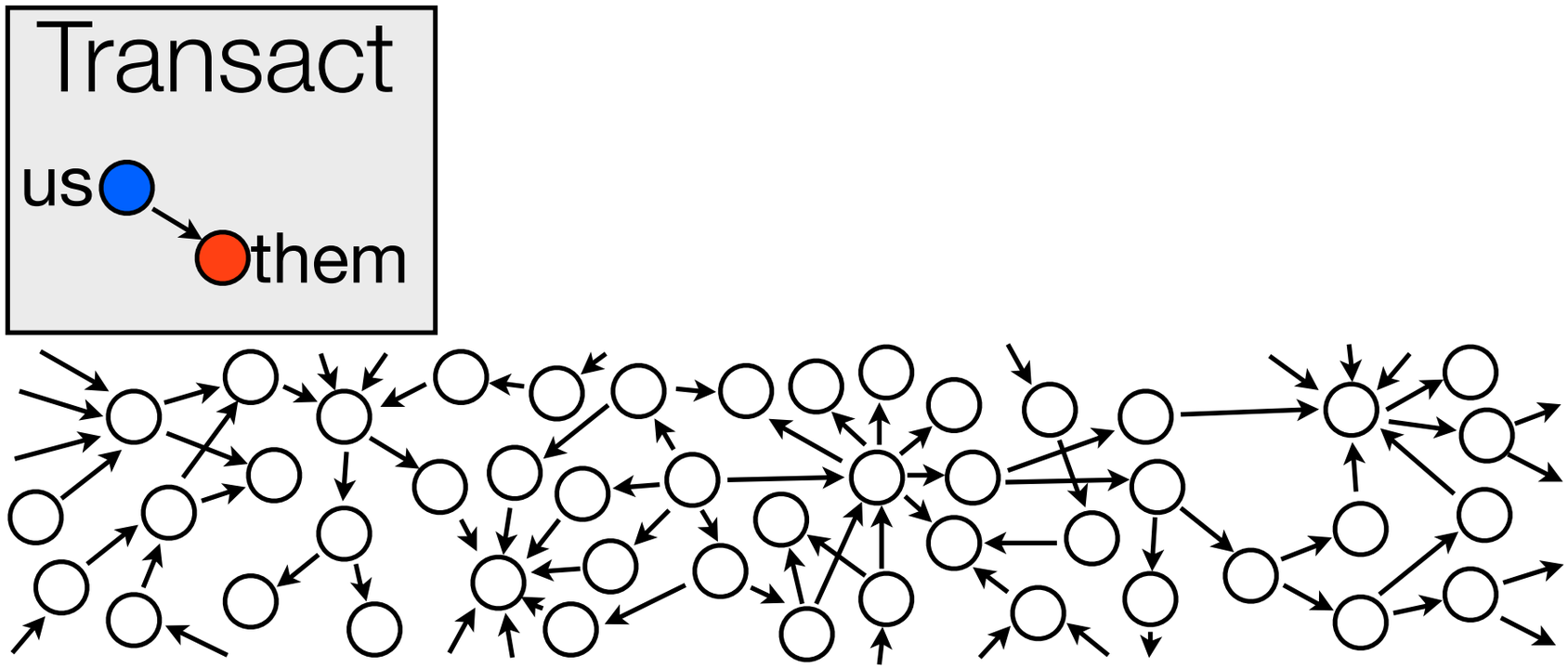

Transact
us
them


SUPER MARIO 3
House
ubuntu
CASIO
Bitcoin
organic colombian latrochela
Bakkav
bitcoin.org
ZYROSTARS
THE LITTLE BOOK OF ANSWERS

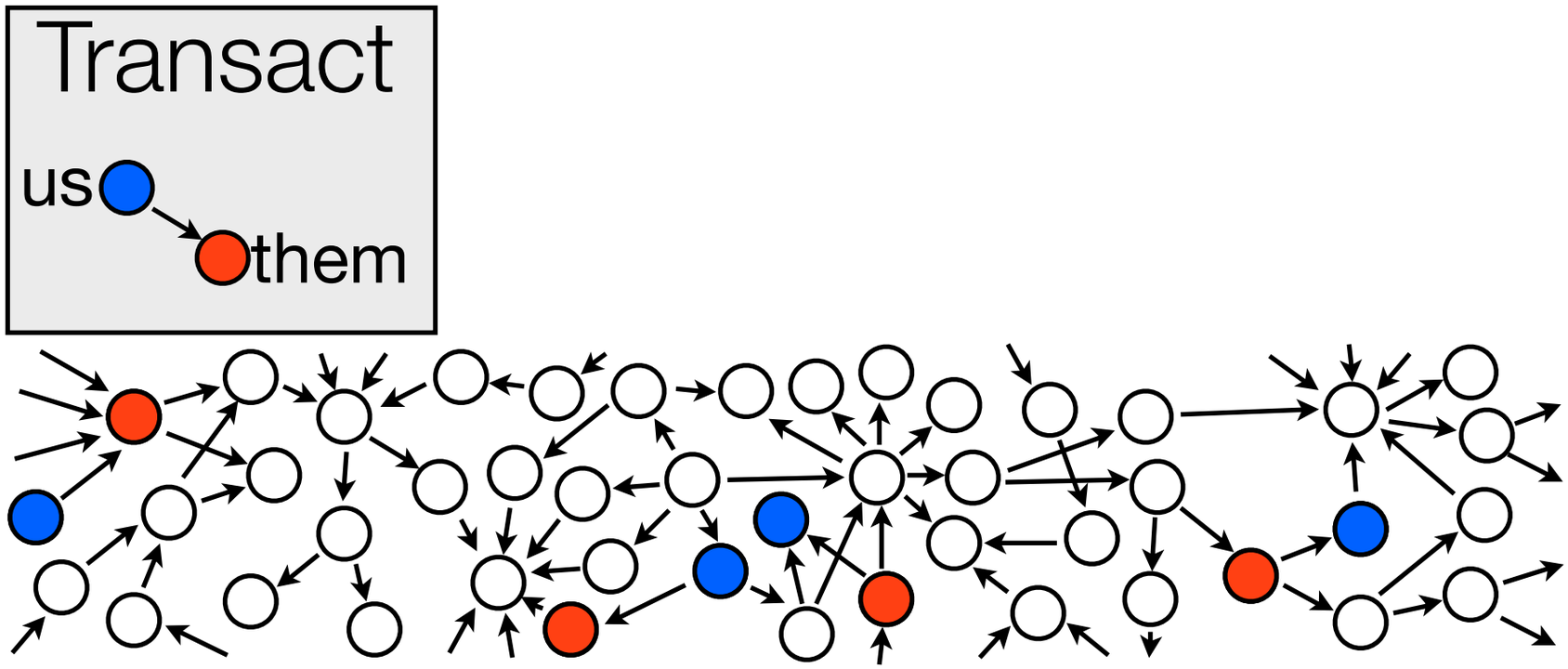
Putting it all together



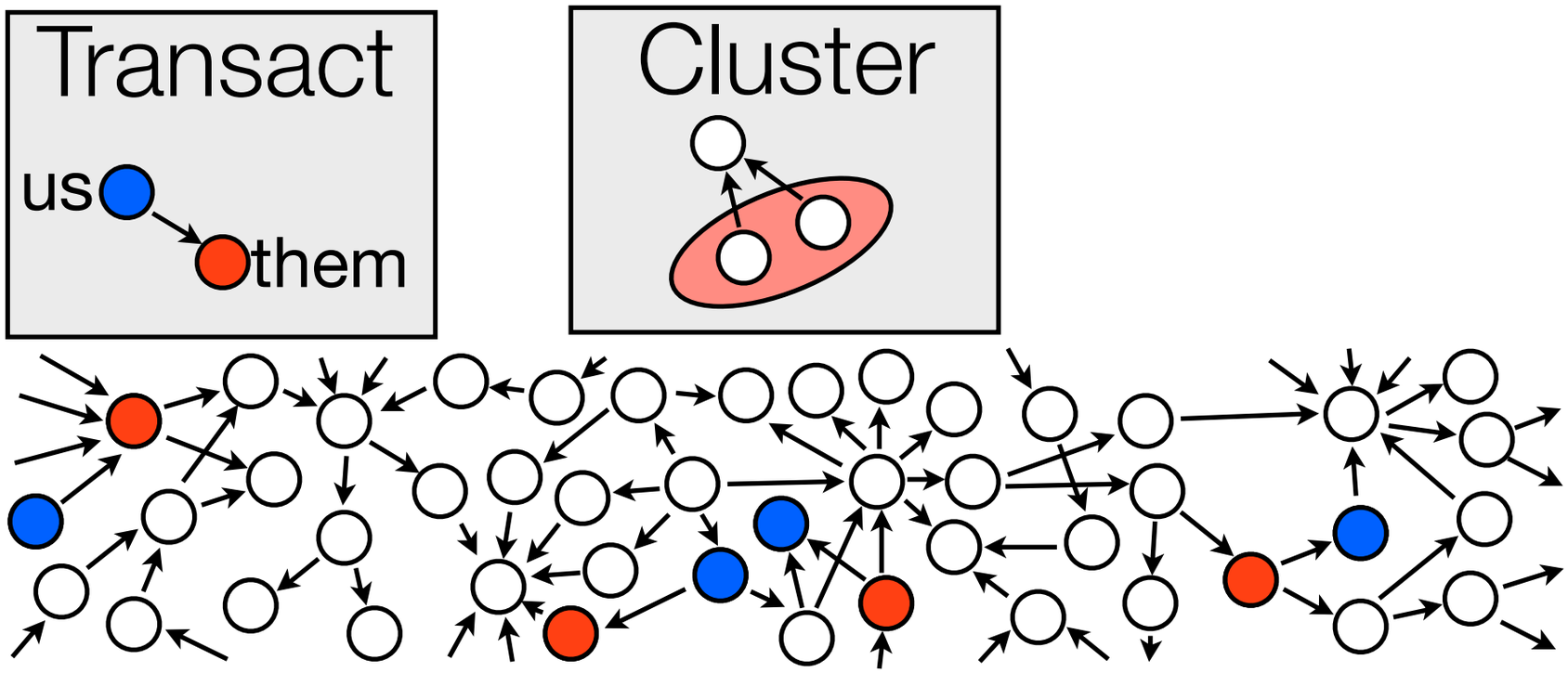
Putting it all together



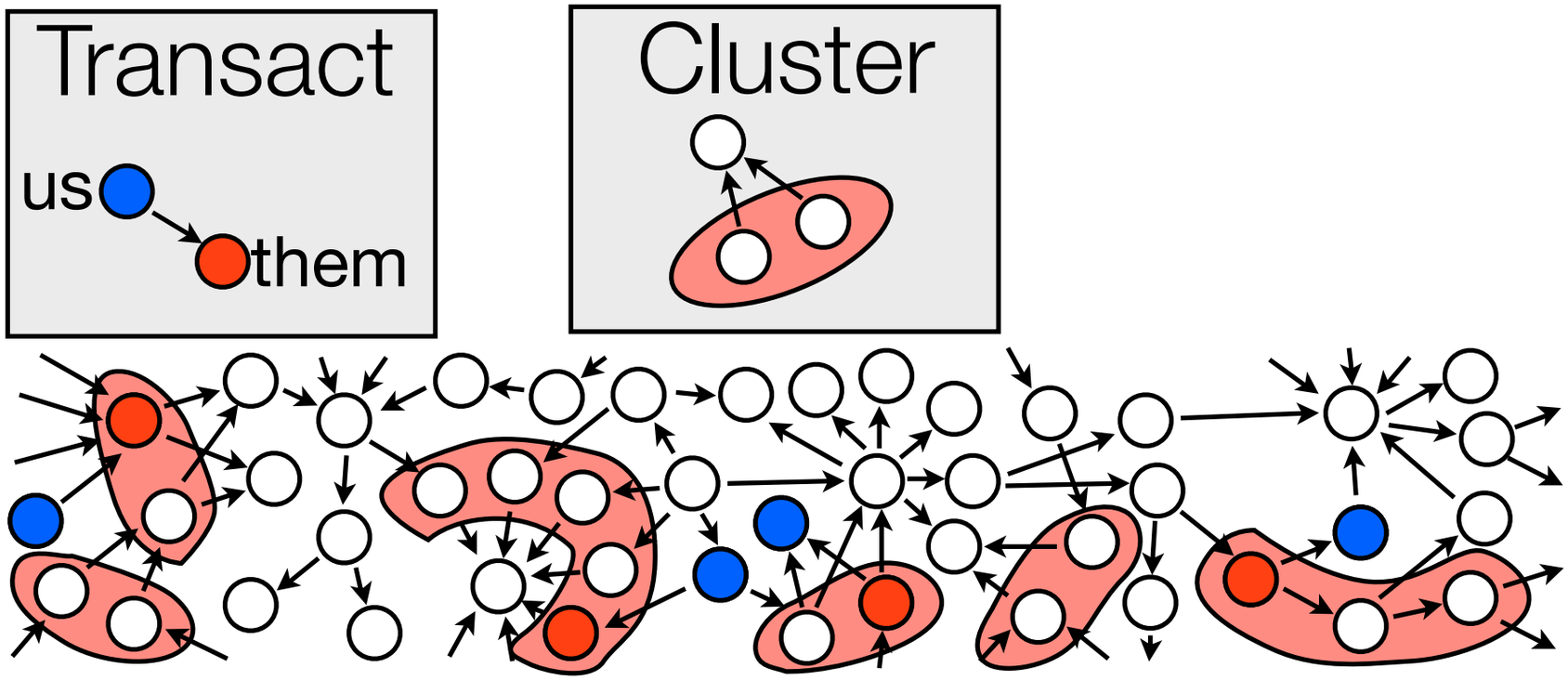
Putting it all together



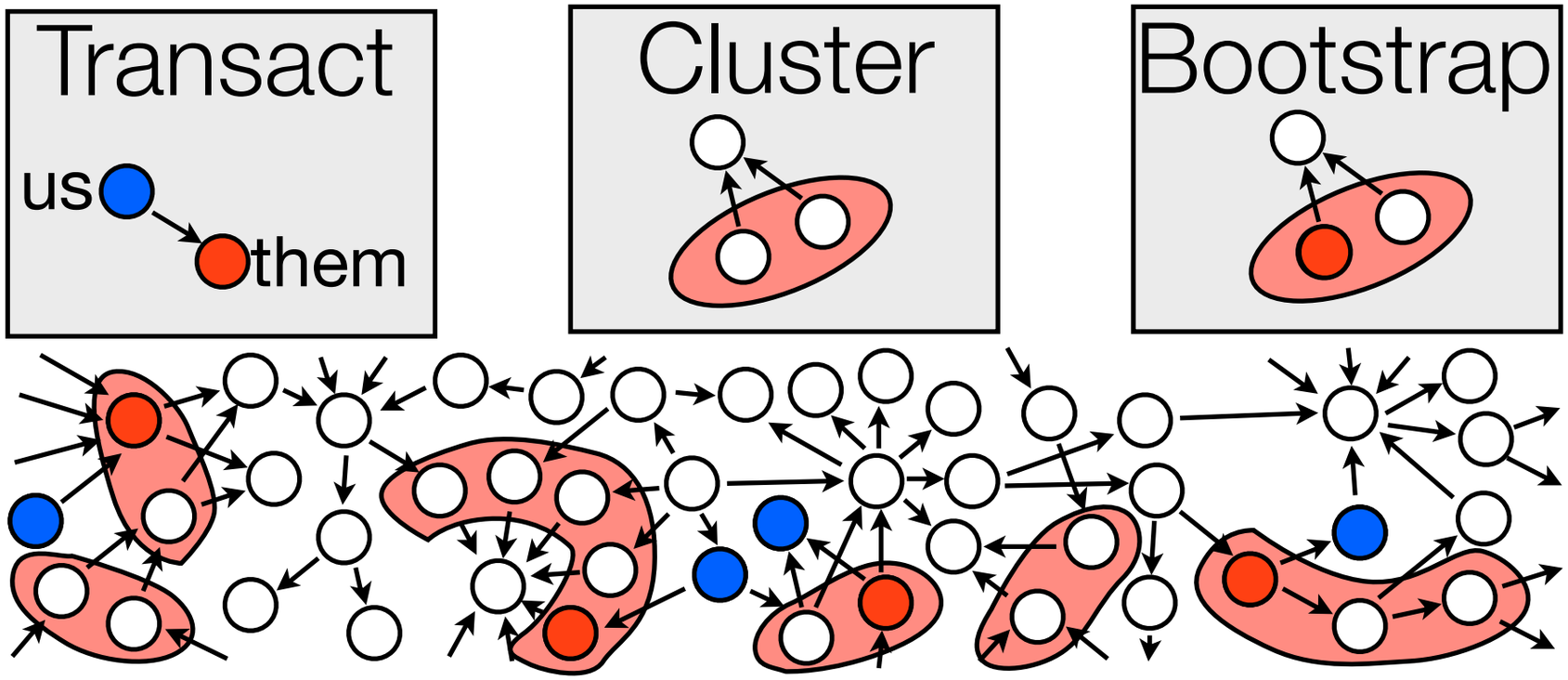
Putting it all together



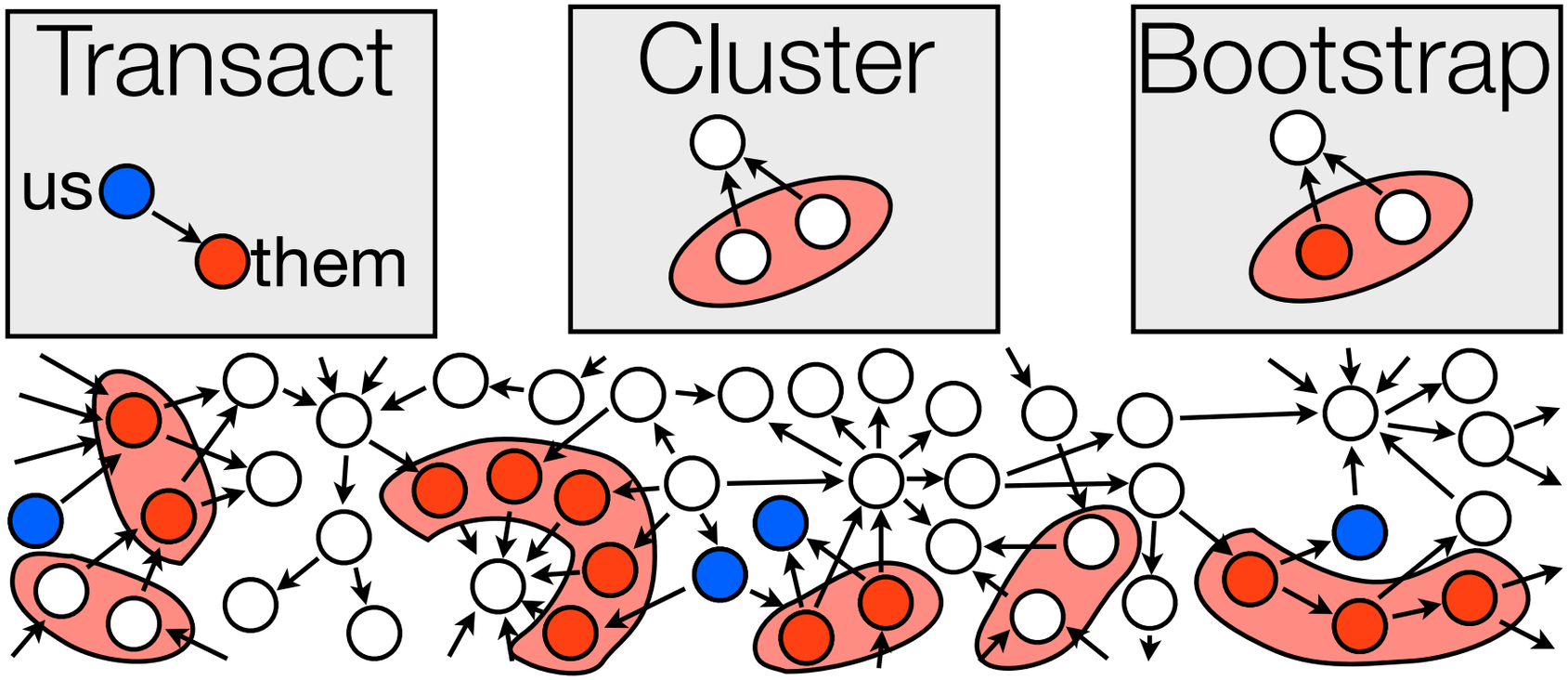
Putting it all together



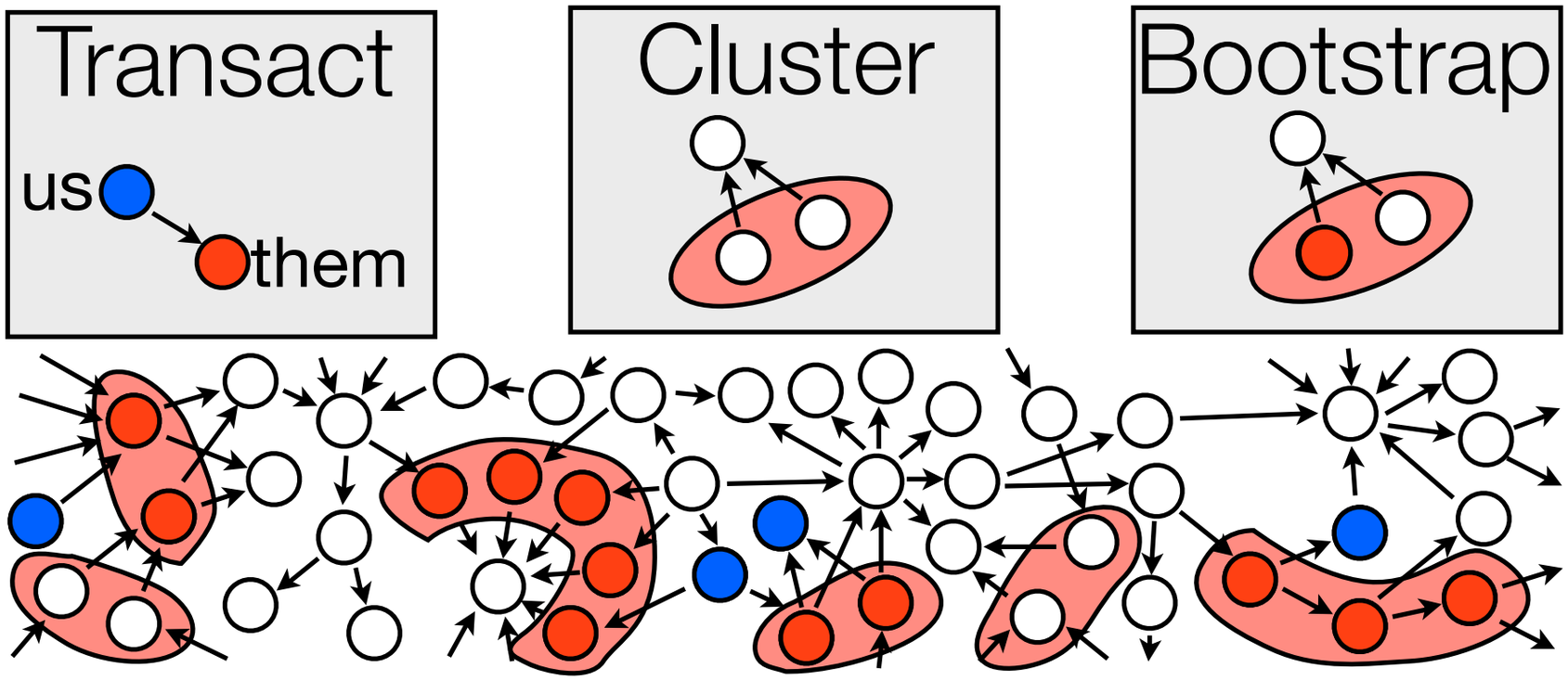
Putting it all together



Putting it all together



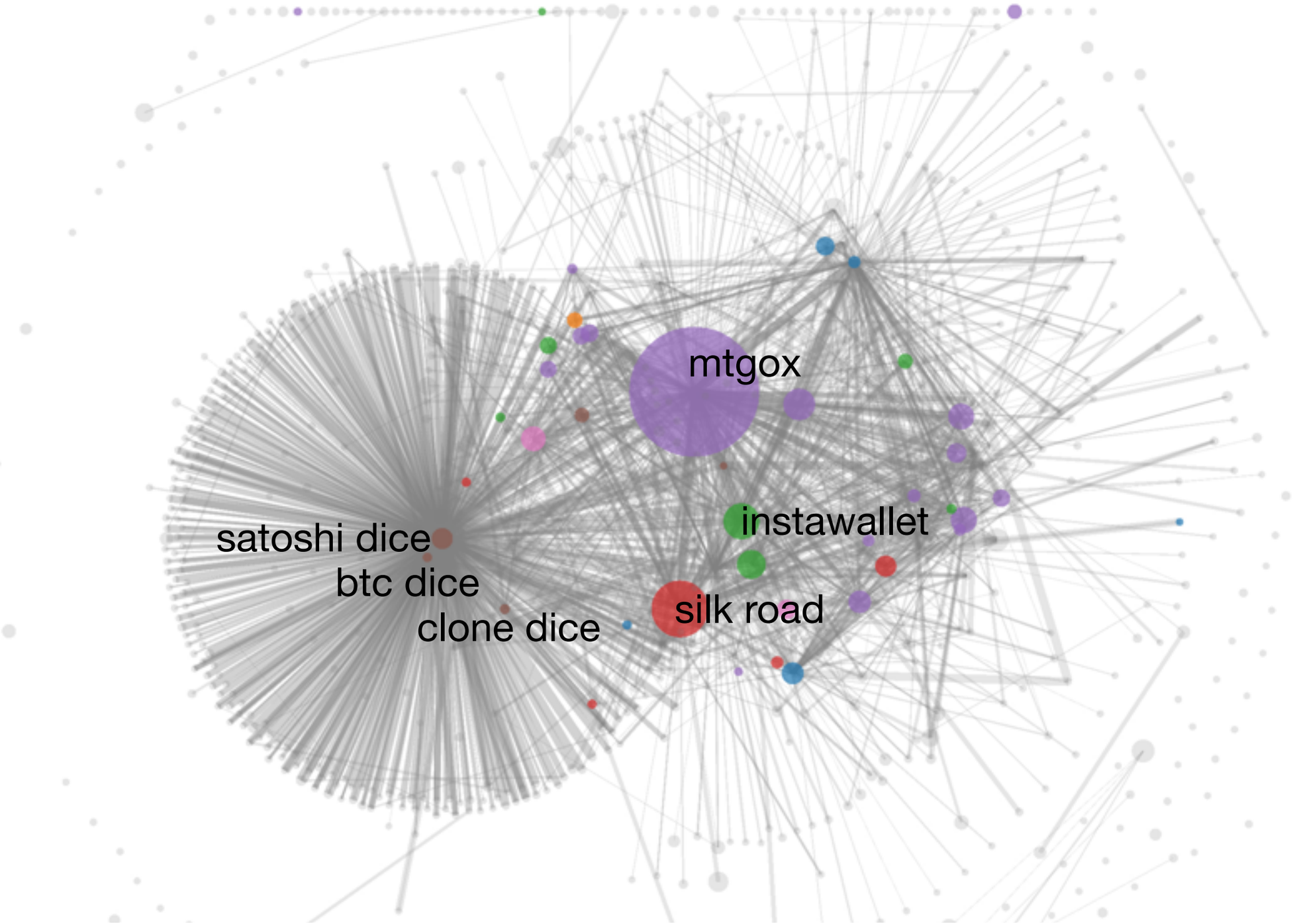
Putting it all together



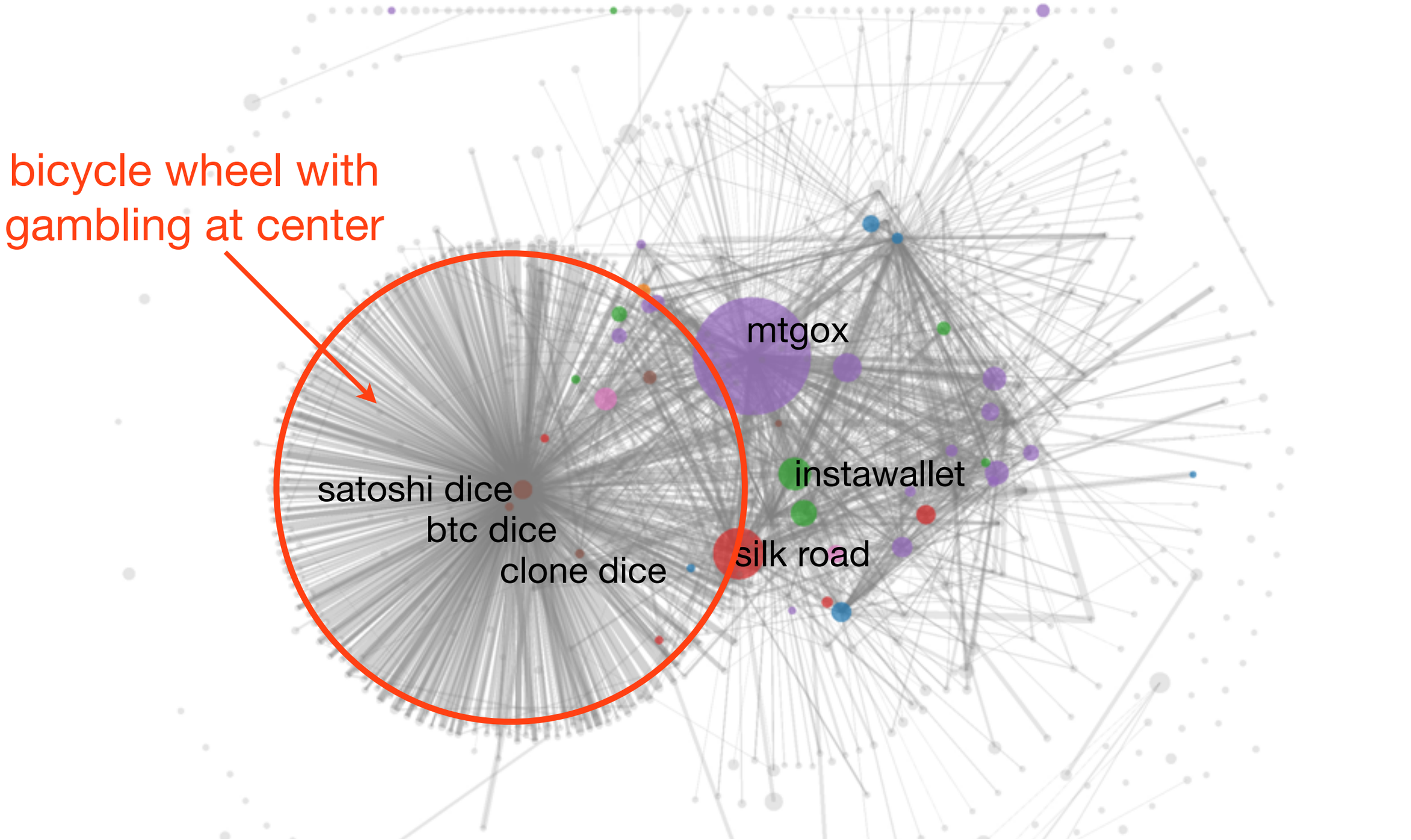
Interacted with 31 MtGox addresses, tagged 518,723!

Participated in 344 transactions and tagged 1.3M public keys

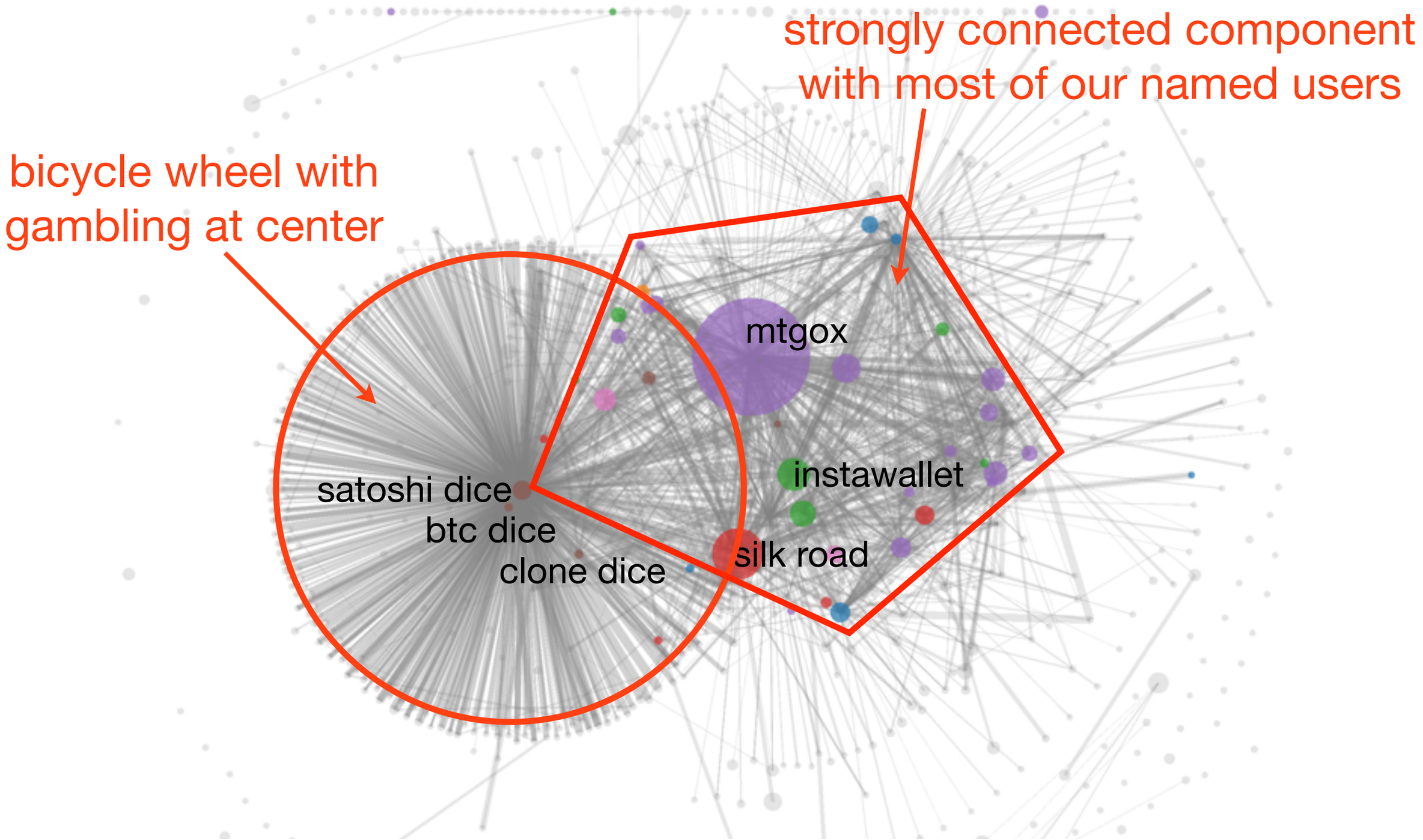
Clustering using Heuristic 2



Clustering using Heuristic 2



Clustering using Heuristic 2



Following bitcoins

Following bitcoins

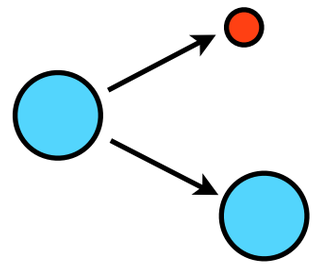
Can see when bitcoins meaningfully cross cluster boundaries

Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries

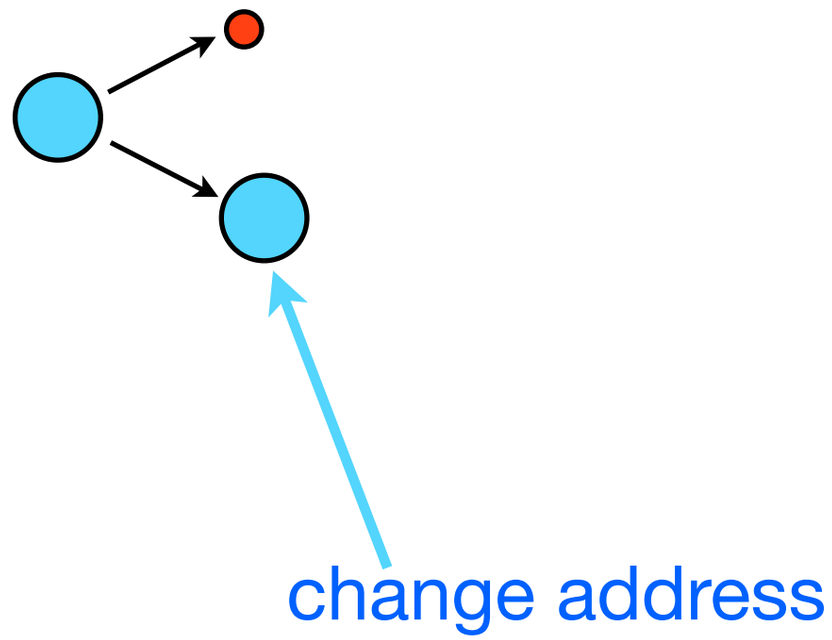
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries



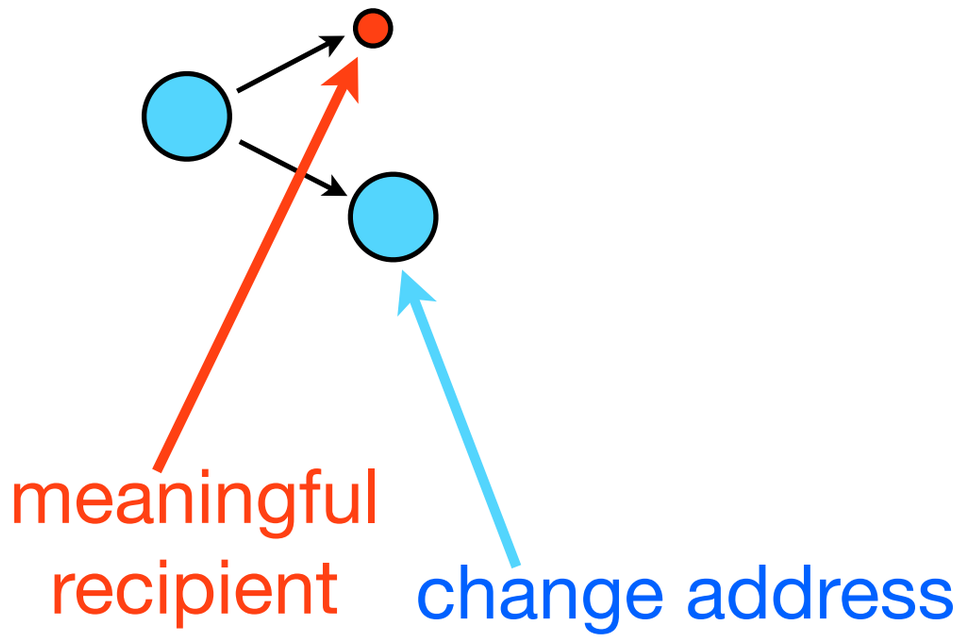
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries



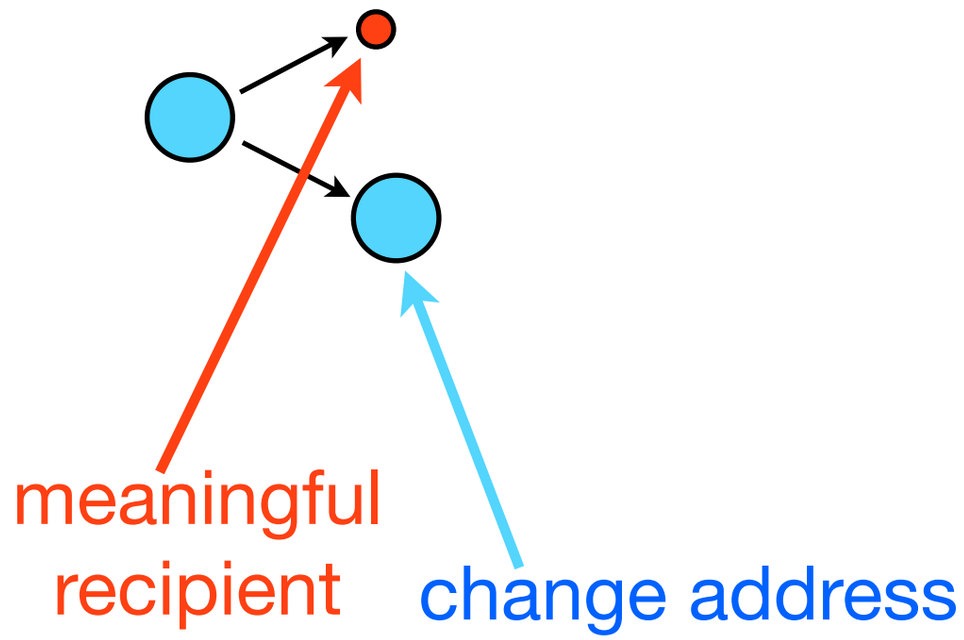
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries



Following bitcoins

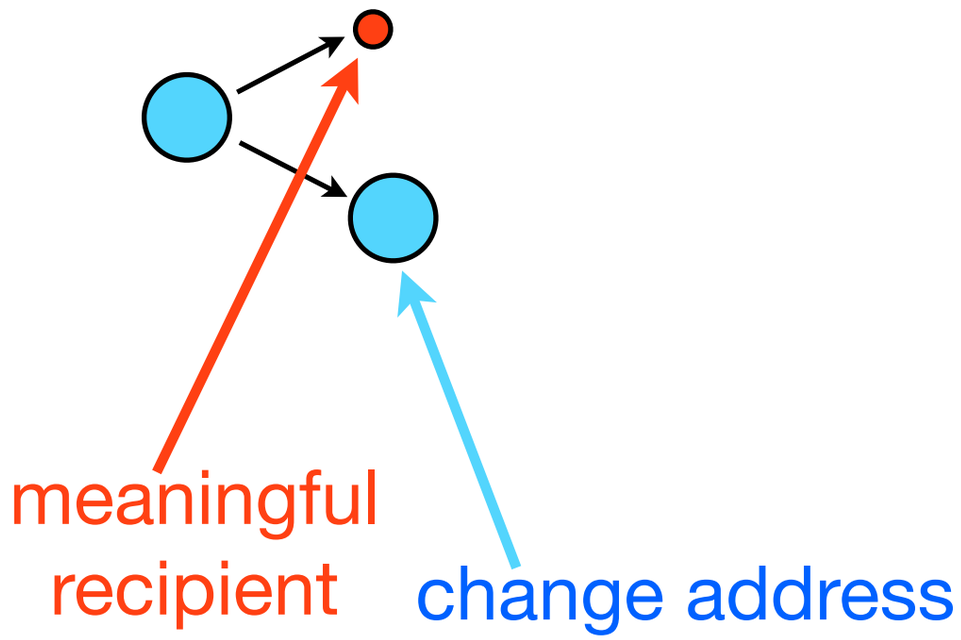
Can see when bitcoins meaningfully cross cluster boundaries



Identifying recipients potentially de-anonymizes user

Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries



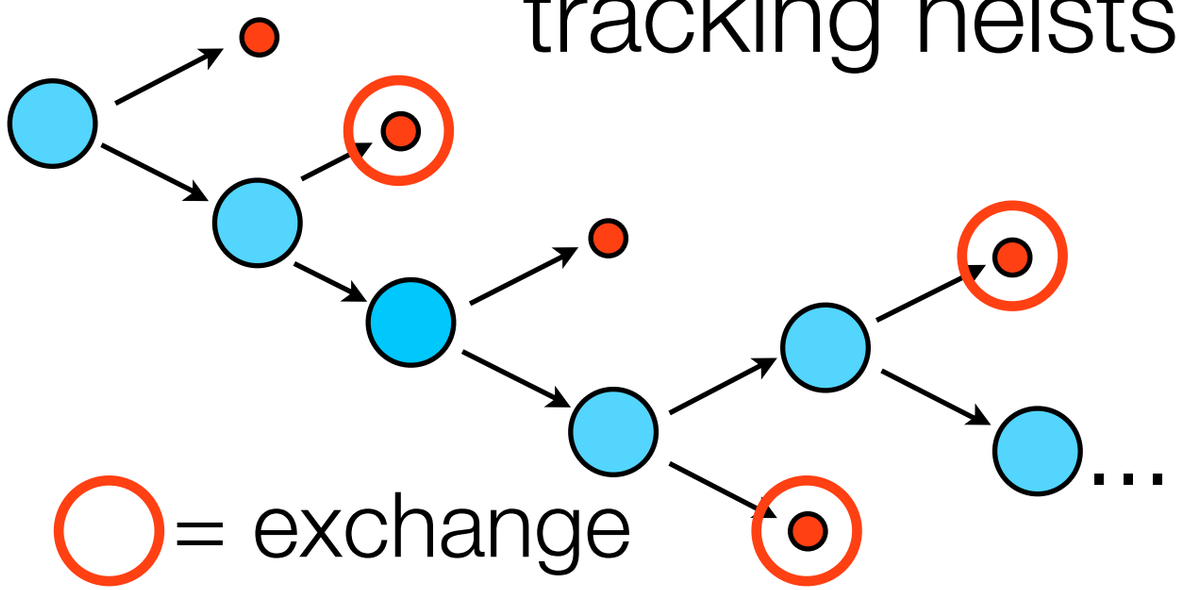
Identifying recipients potentially de-anonymizes user

Hypothesis: if you subpoena exchanges, you can identify users

Tracking technique

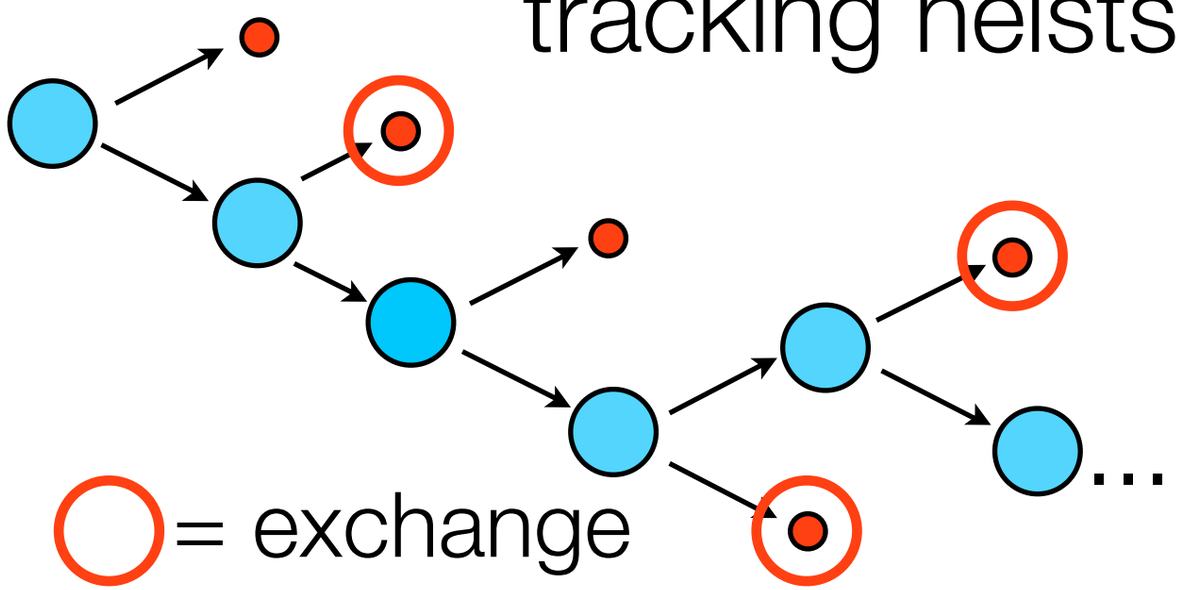
Tracking technique

tracking heists

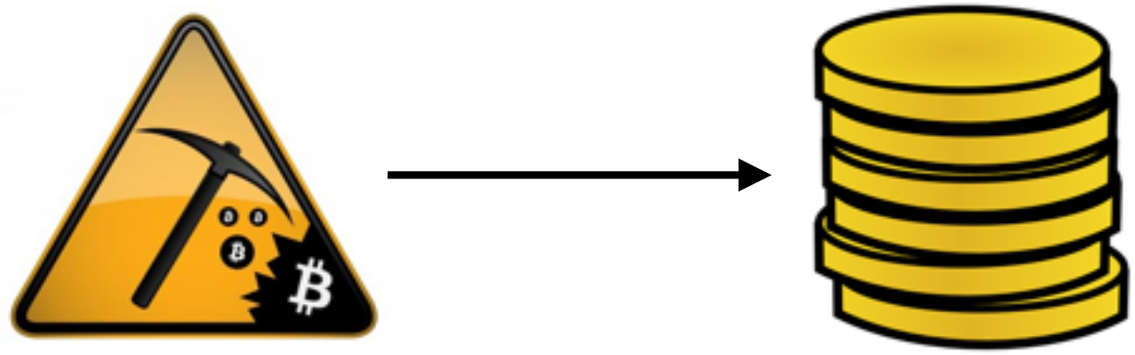


Tracking technique

tracking heists



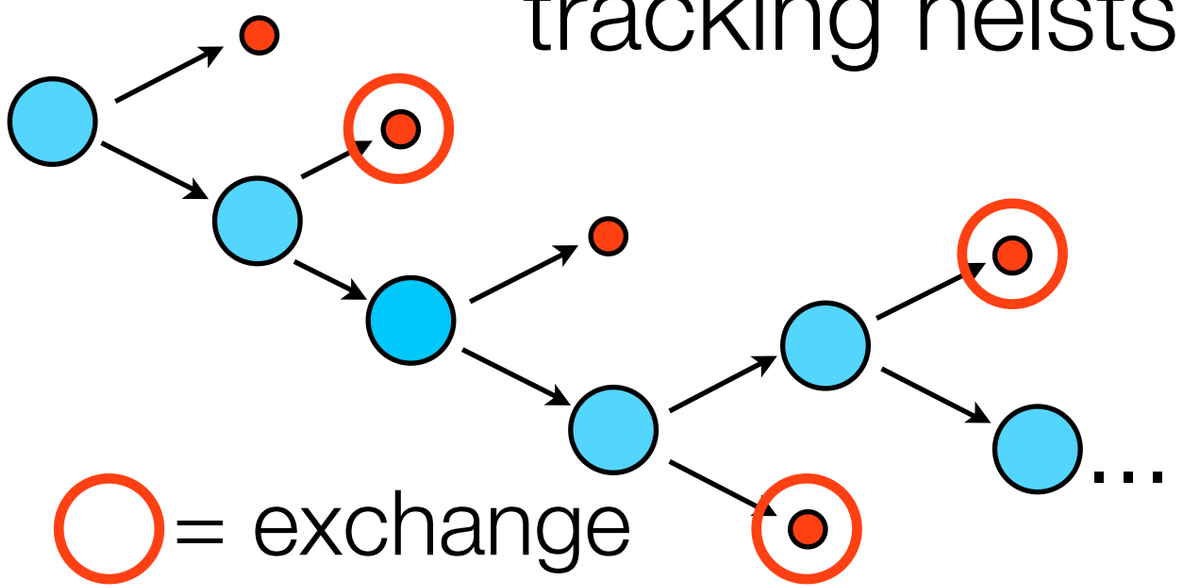
[HD**M**+ NDSS'14]

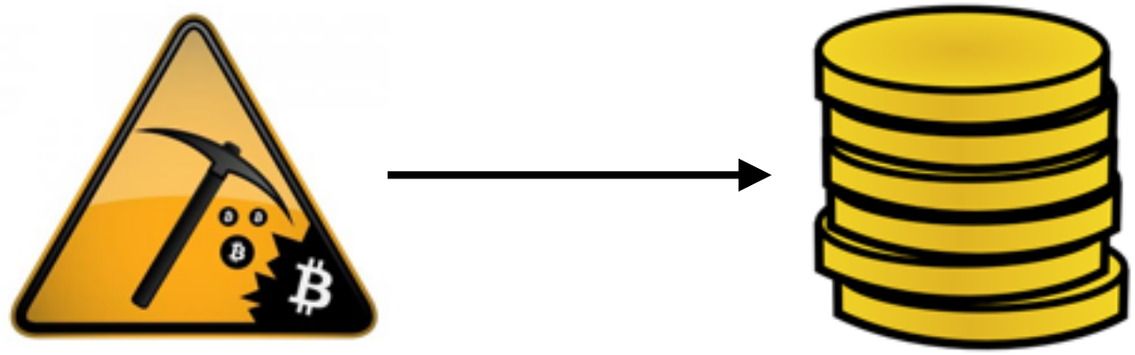


Tracking technique

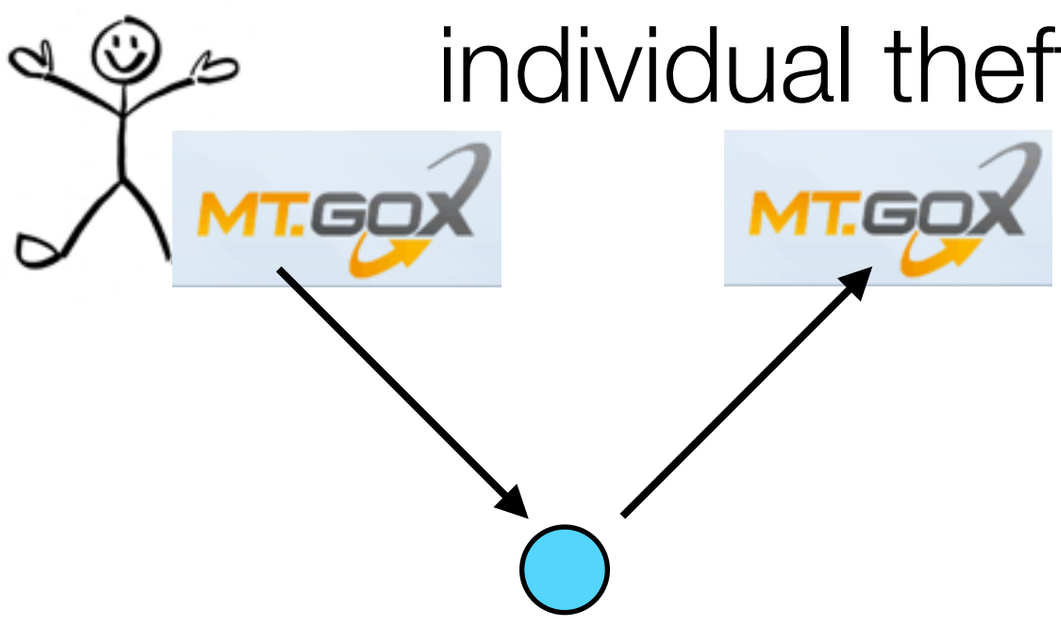
tracking heists

[HD**M**+ NDSS'14]





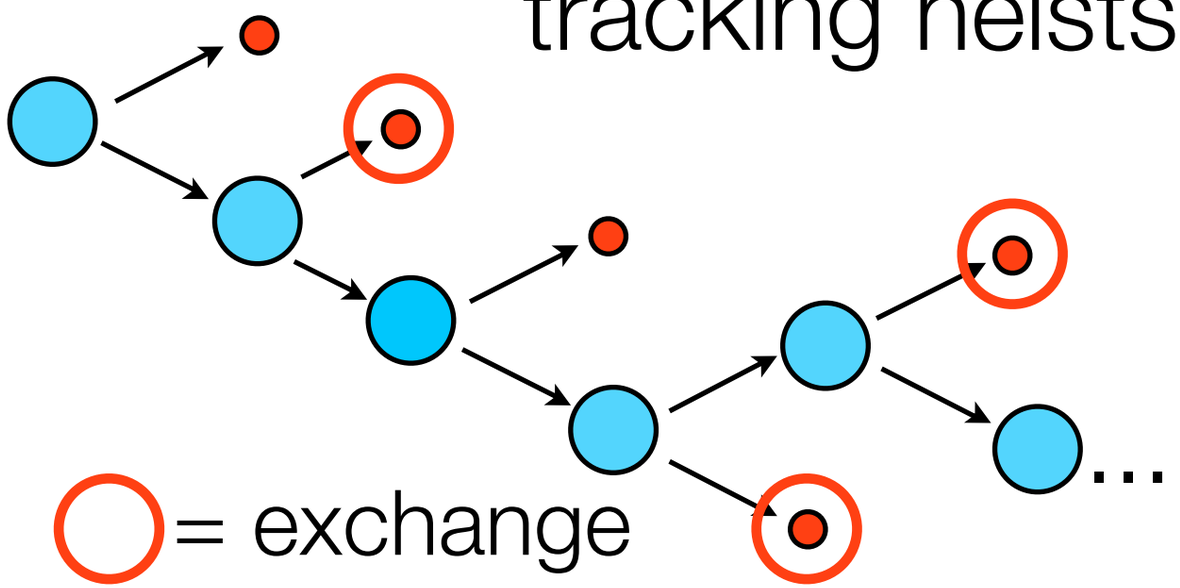
individual thefts

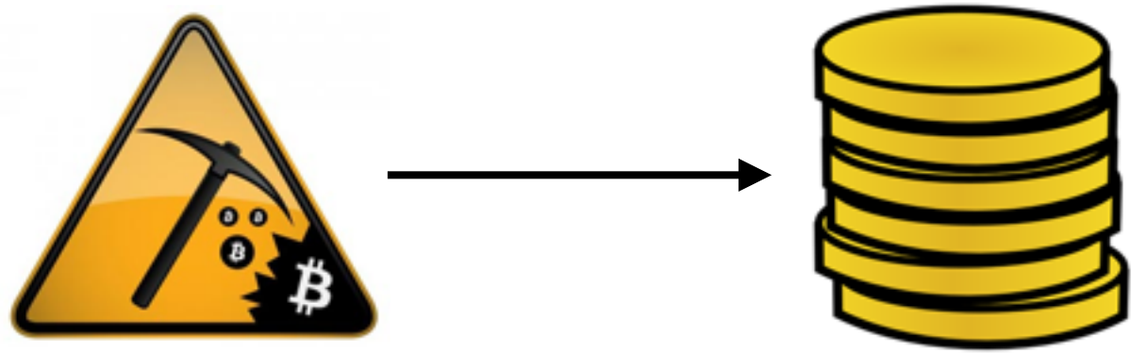


Tracking technique

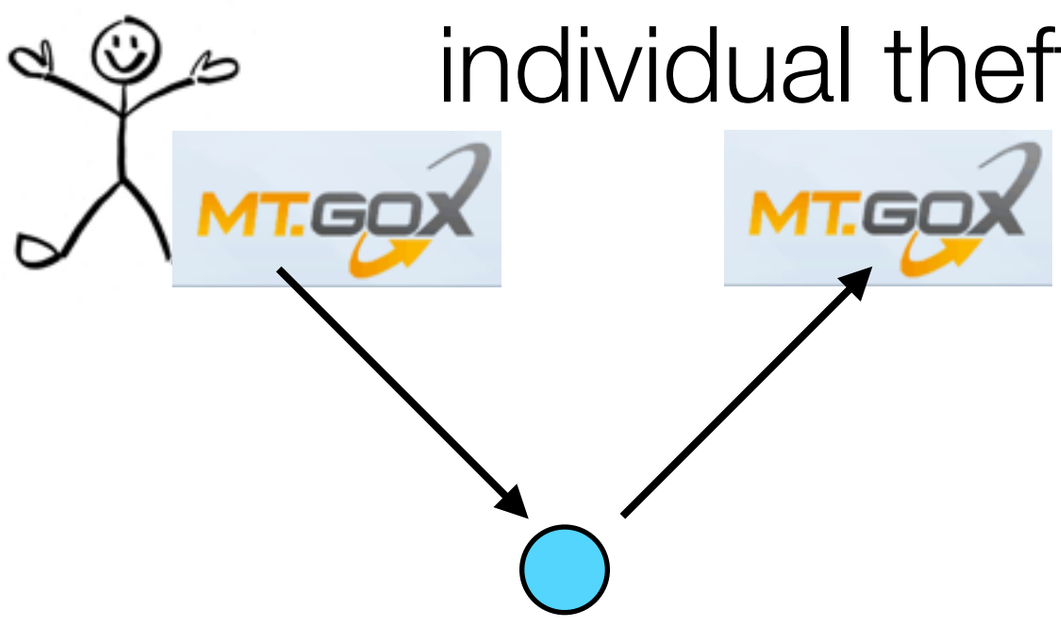
tracking heists

[HD**M**+ NDSS'14]





individual thefts



service interaction

Andy Greenberg, Forbes Staff

Covering the worlds of data security, privacy and hacker culture.

+ Follow (1,142)

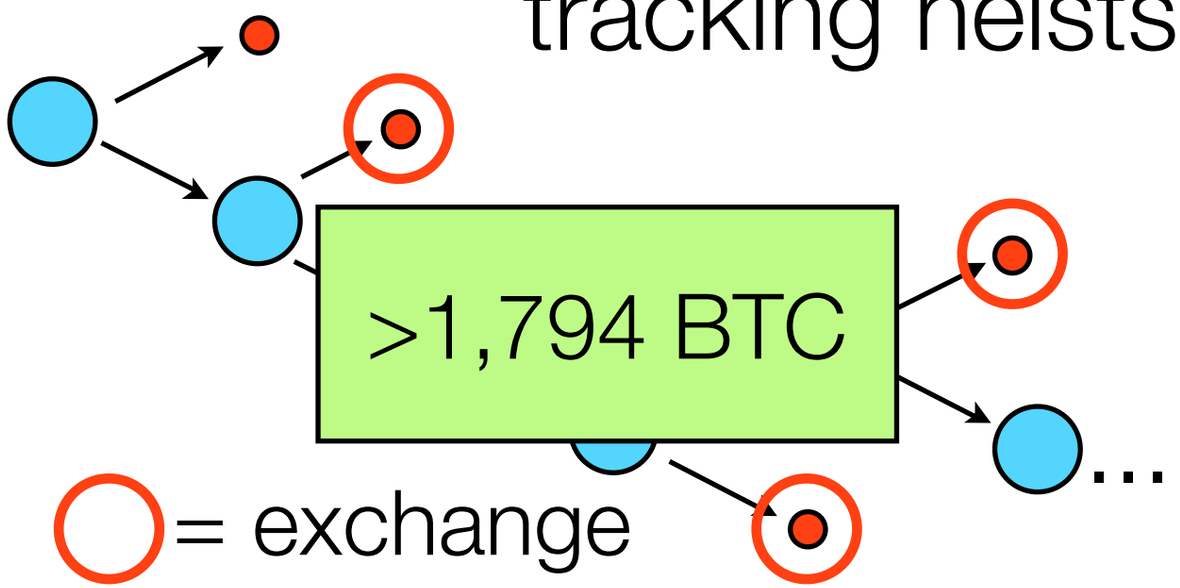
SECURITY | 9/05/2013 @ 10:36AM | 131,694 views

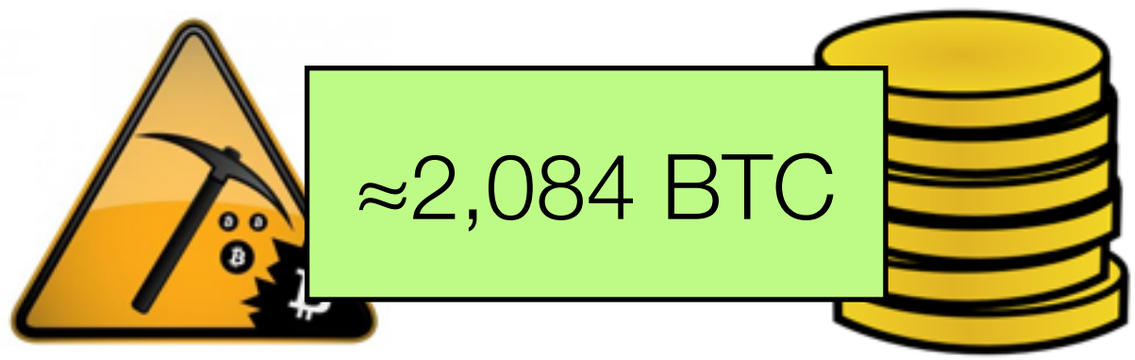
Follow The Bitcoins: How We Got Busted Buying Drugs On Silk Road's Black Market

Tracking technique

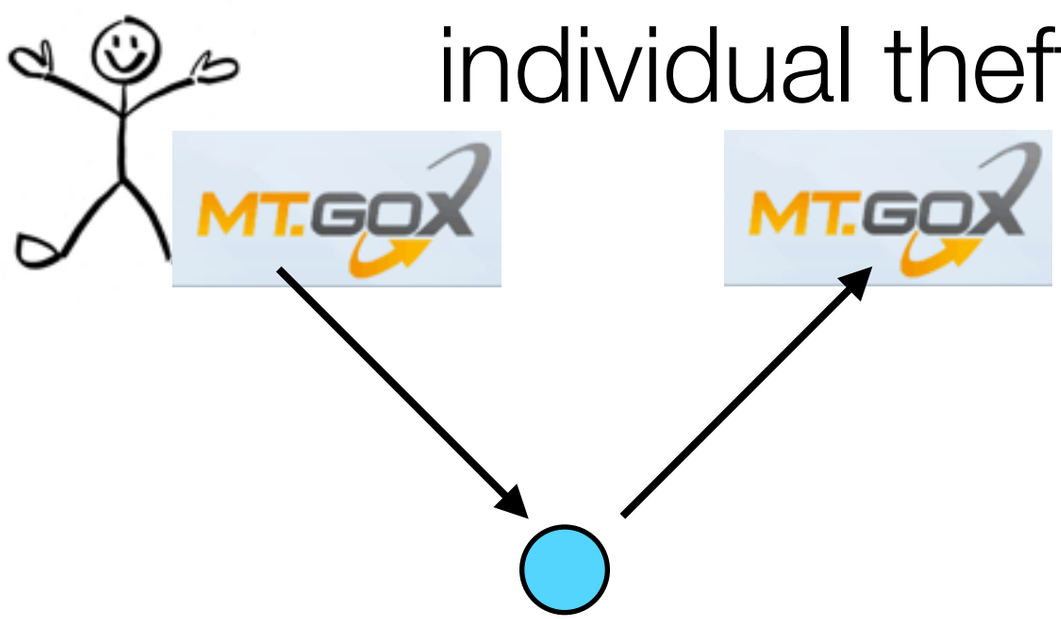
tracking heists

[HD**M**+ NDSS'14]





individual thefts



service interaction

Andy Greenberg, Forbes Staff

Covering the worlds of data security, privacy and hacker culture.

+ Follow (1,142)

SECURITY | 9/05/2013 @ 10:36AM | 131,694 views

Follow The Bitcoins: How We Got Busted Buying Drugs On Silk Road's Black Market

Tracking technique

tracking heists [HD**M**+ NDSS'14]

Takeaway

How much anonymity does Bitcoin really provide?

Takeaway

How much anonymity does Bitcoin really provide?

Our analysis provides a real-world way to track flows of bitcoins

Takeaway

How much anonymity does Bitcoin really provide?

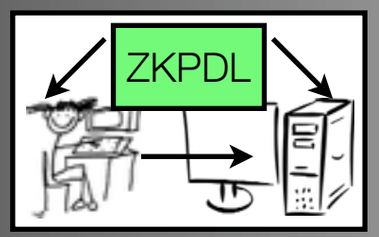
Our analysis provides a real-world way to track flows of bitcoins

Seems hard to launder significant quantities of money

My research

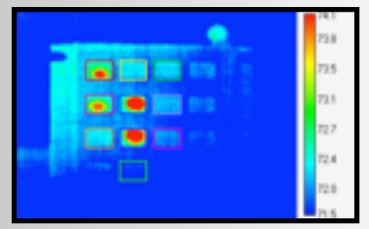

[**M**SF10,L**M**13]

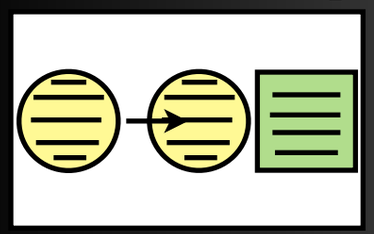

[B**M**T14]

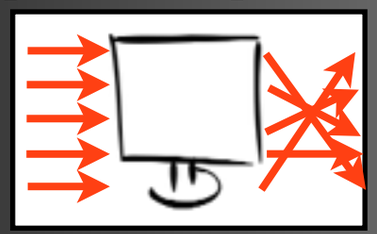


[**M**EKHL10]

[**M**MCS11]

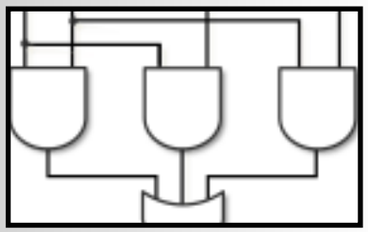

[M**M**S11]

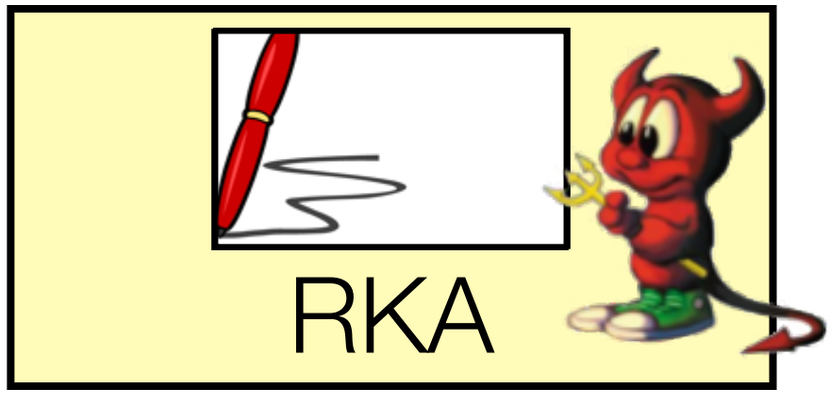

[C**M**14]


[CKL**M**12,13a,13b]

[C**M**Z13]


[**M**P+13,HD**M**+14]


[O**M**SK13]



Acknowledgements

Acknowledgements