goto; chicago

Practical Cryptography

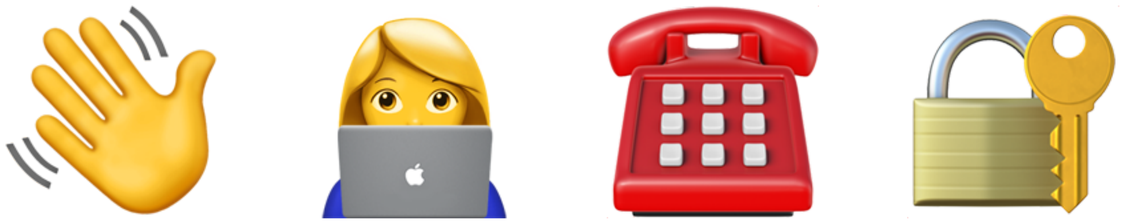
An Introduction to Public Key Cryptography

Kelley Robinson


CAUTION

Practical Cryptography

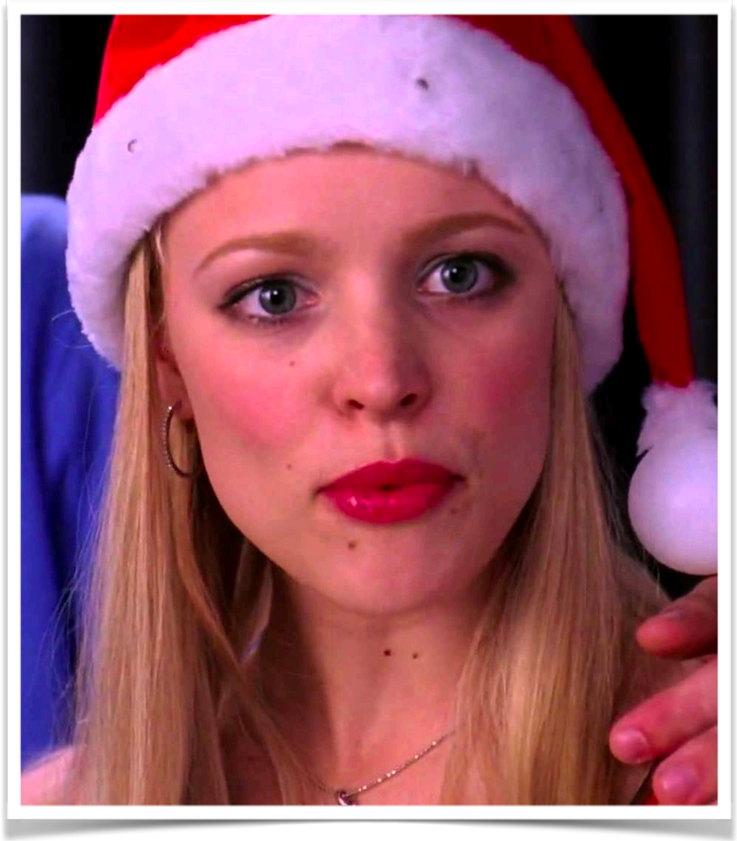
Introduction to Public Key Cryptography

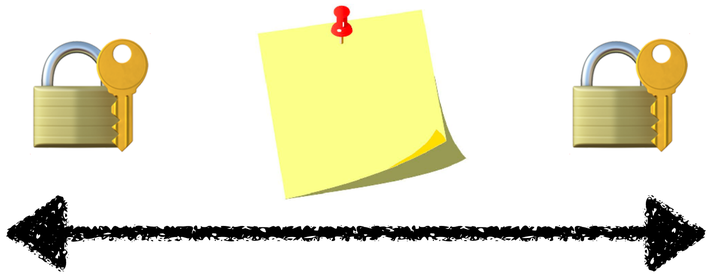


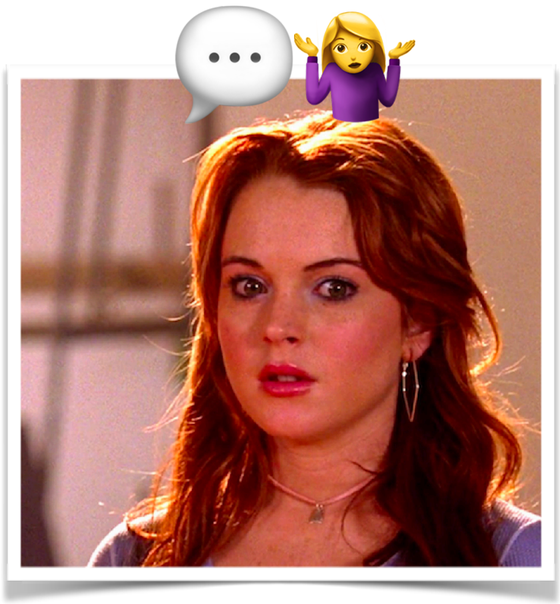
@kelleyrobinson

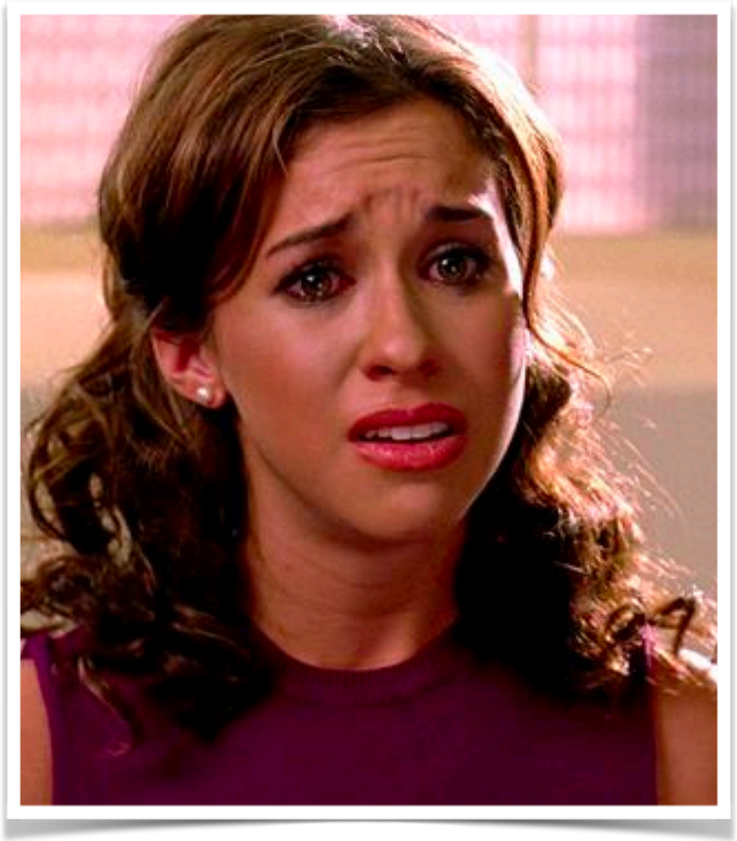
twilio AUTHY

Meet Alice and Bob

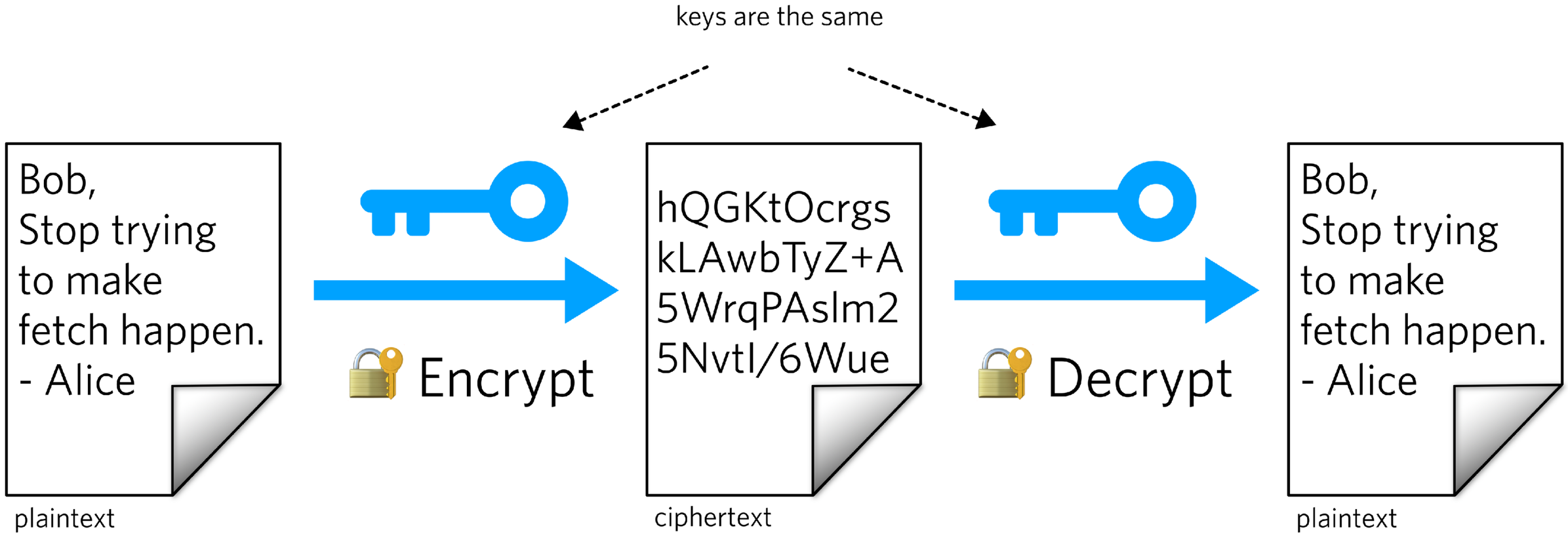








Symmetric Key Cryptography



What is Public Key Crypto?

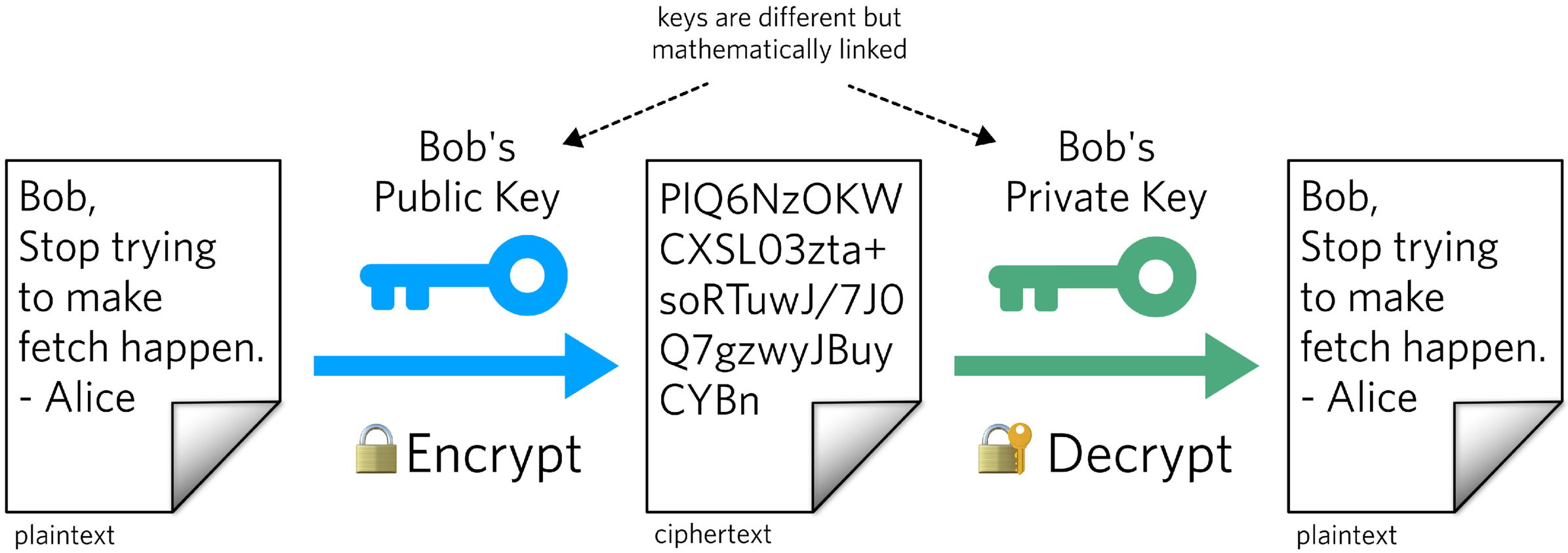
Each entity has 2️⃣ keys

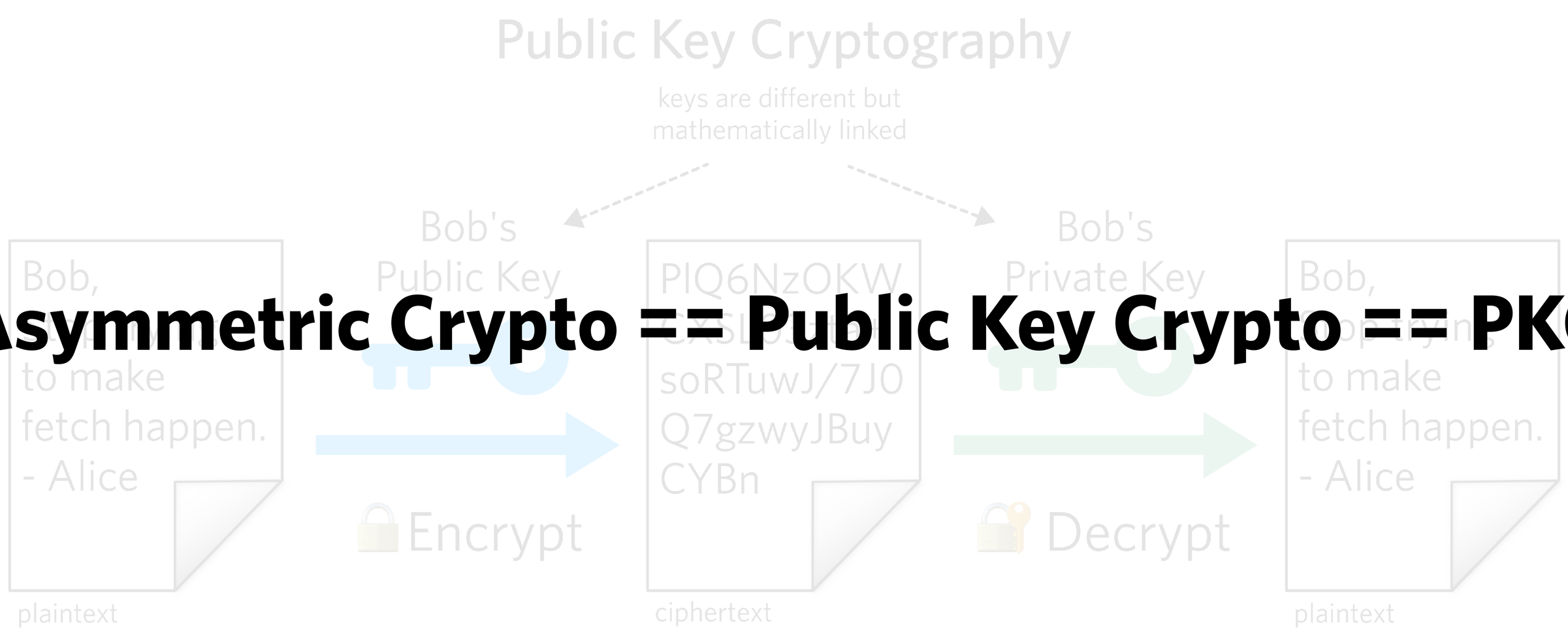
 Public Key - to be shared

 Private Key - to be kept secret

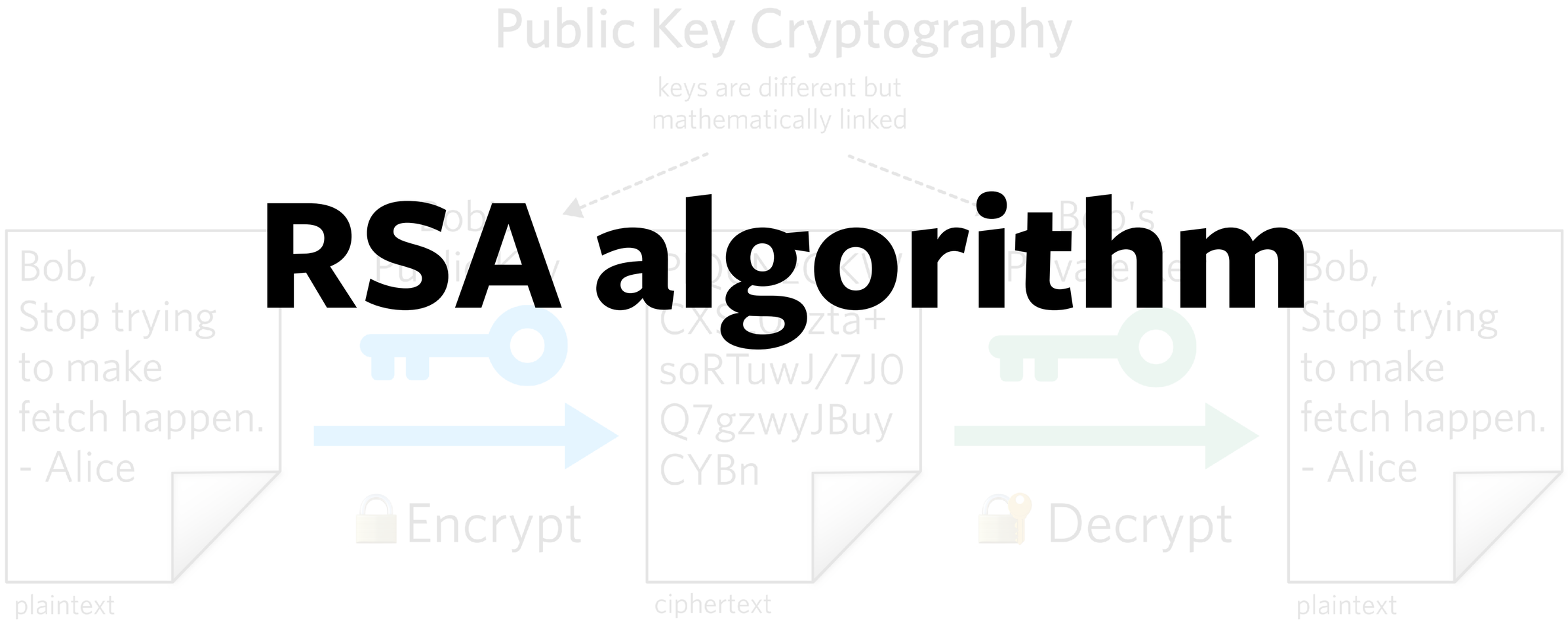
Public Key Cryptography

@kelleyrobinson





Asymmetric Crypto == Public Key Crypto == PKC



RSA algorithm

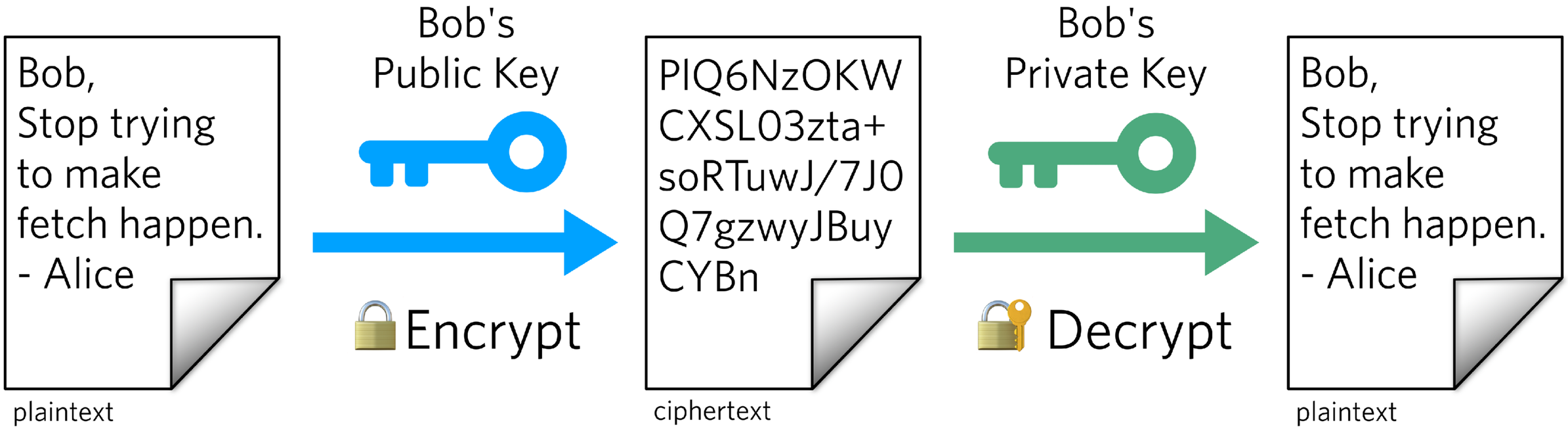
(there are other algorithms, we'll get to that)

What is Public Key Crypto?

Two major use cases (for RSA):

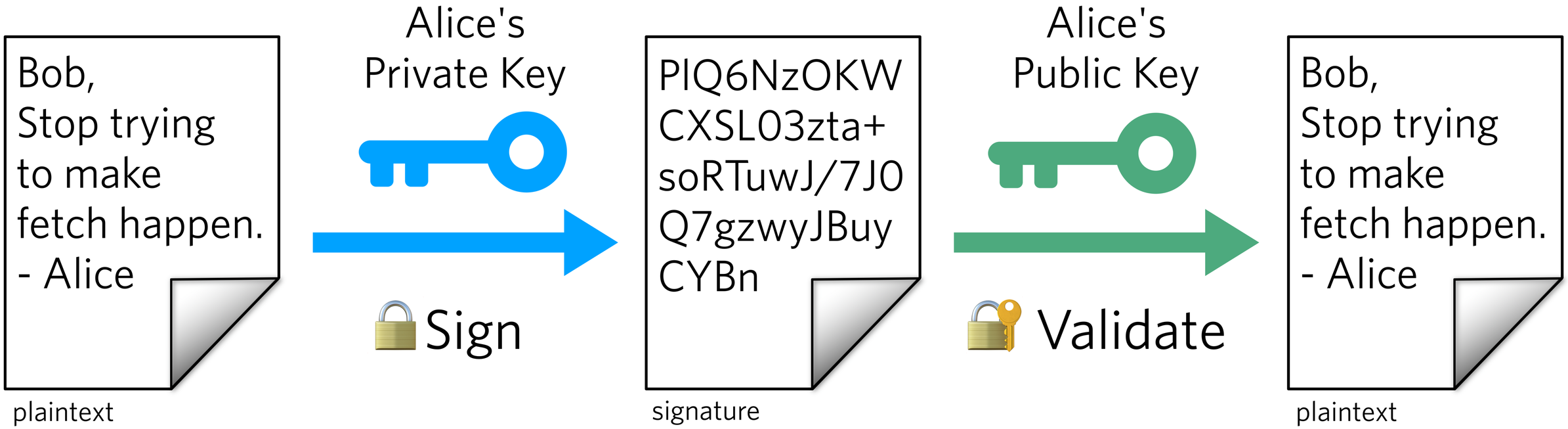
1. **Encrypt** with **Public Key**
2. **Sign** with **Private Key**

Encrypting with **Public Key**



...only Bob can decode the cyphertext

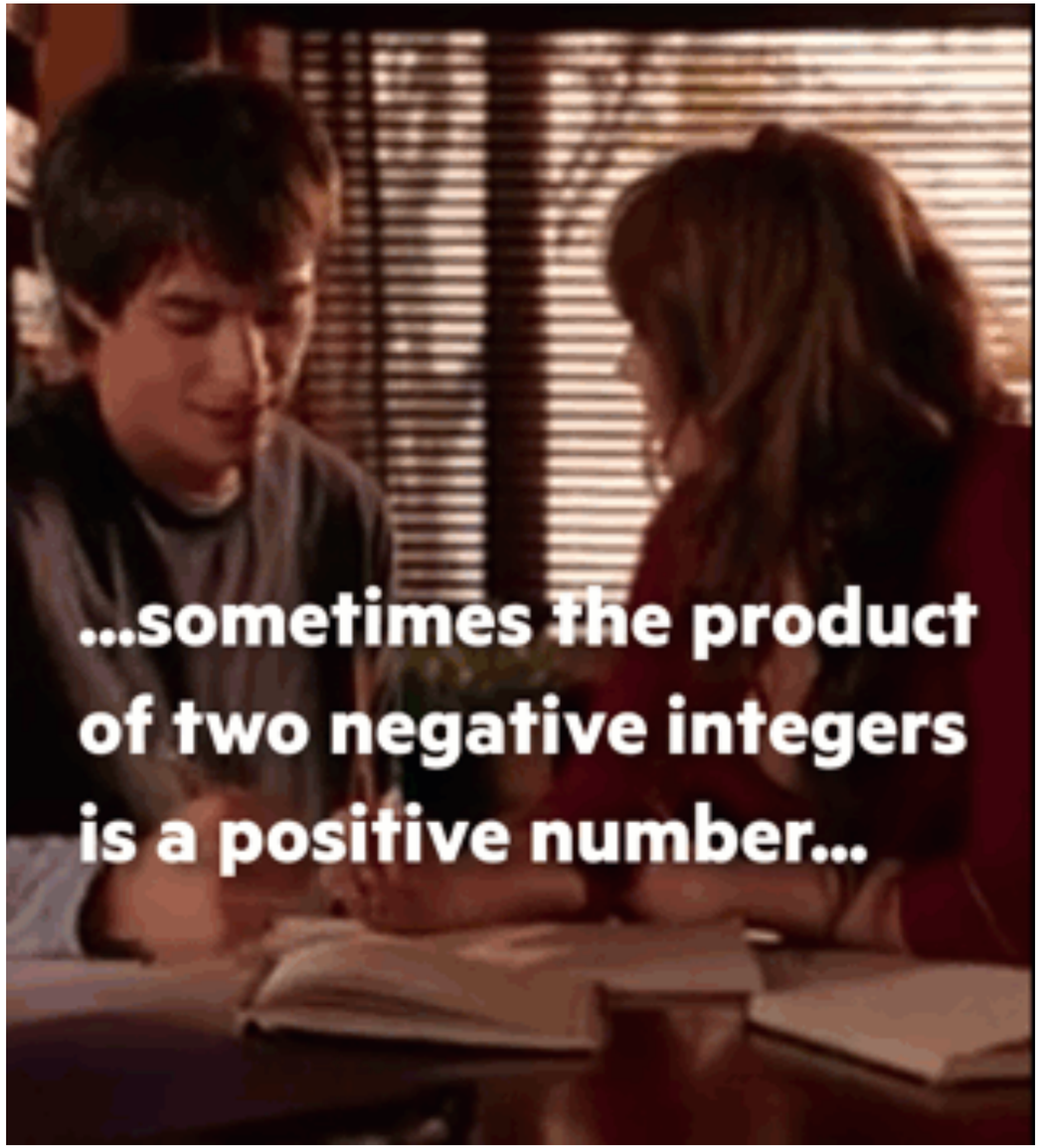
Signing with **Private Key**



...only Alice can be the author

How are keys generated?

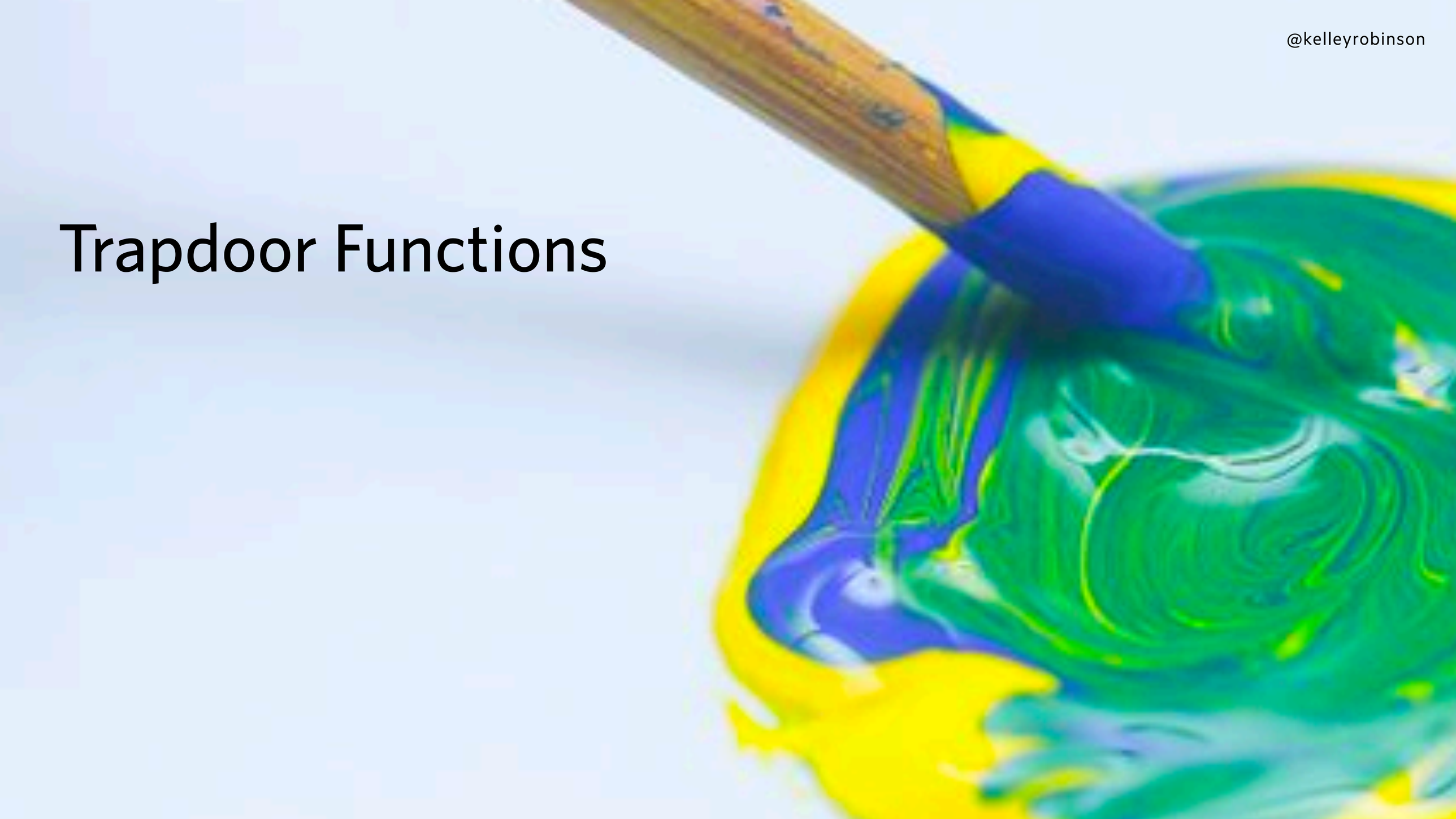
TL;DR: math



@kelleyrobinson

How are keys generated?

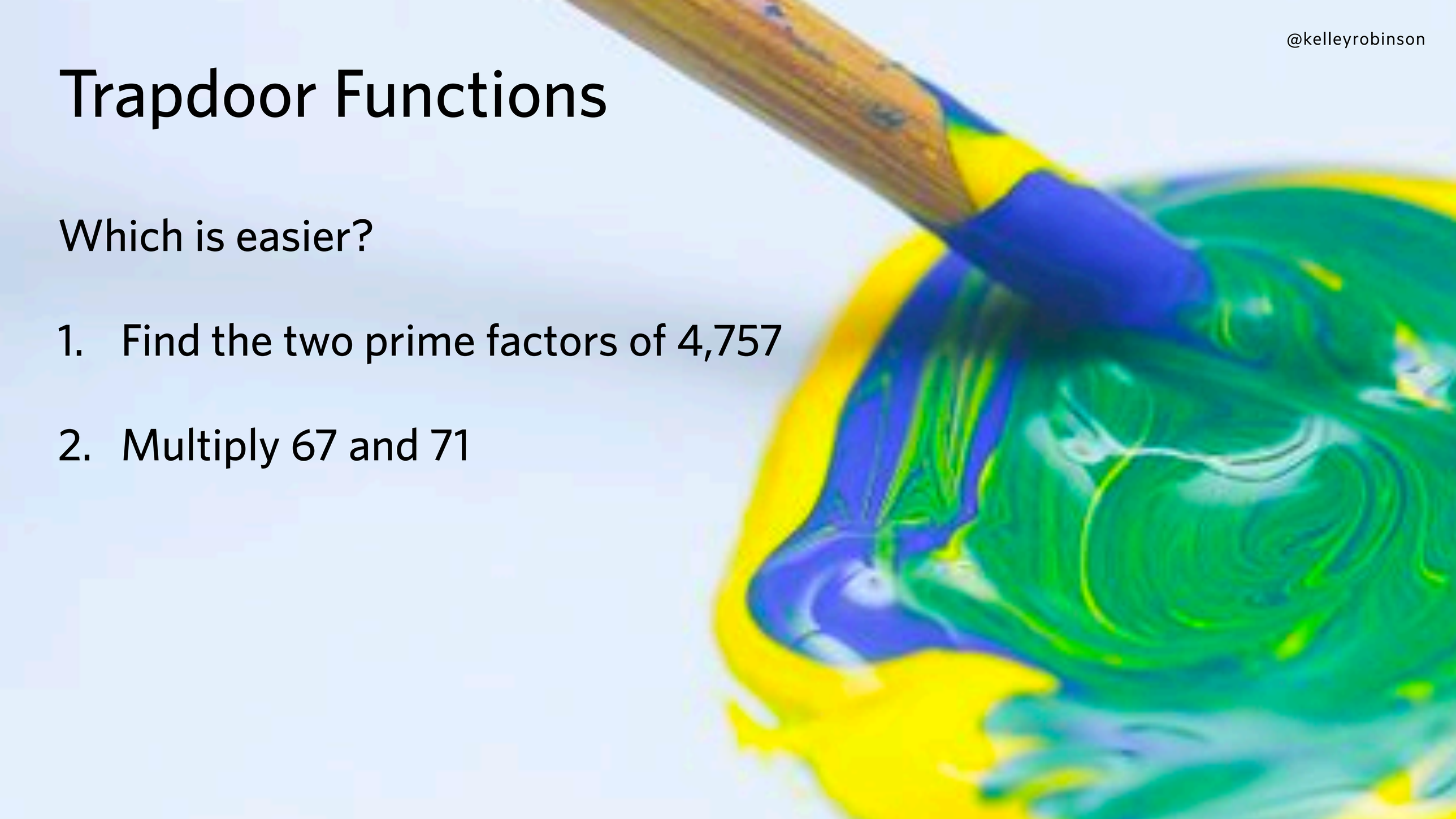
Trapdoor Functions



Trapdoor Functions

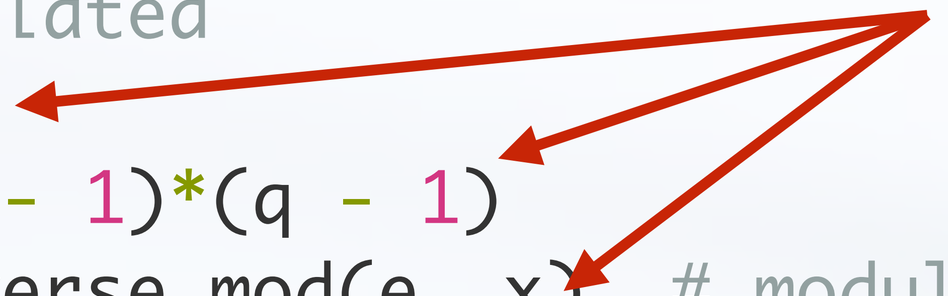
Which is easier?

1. Find the two prime factors of 4,757
2. Multiply 67 and 71



RSA Algorithm Example

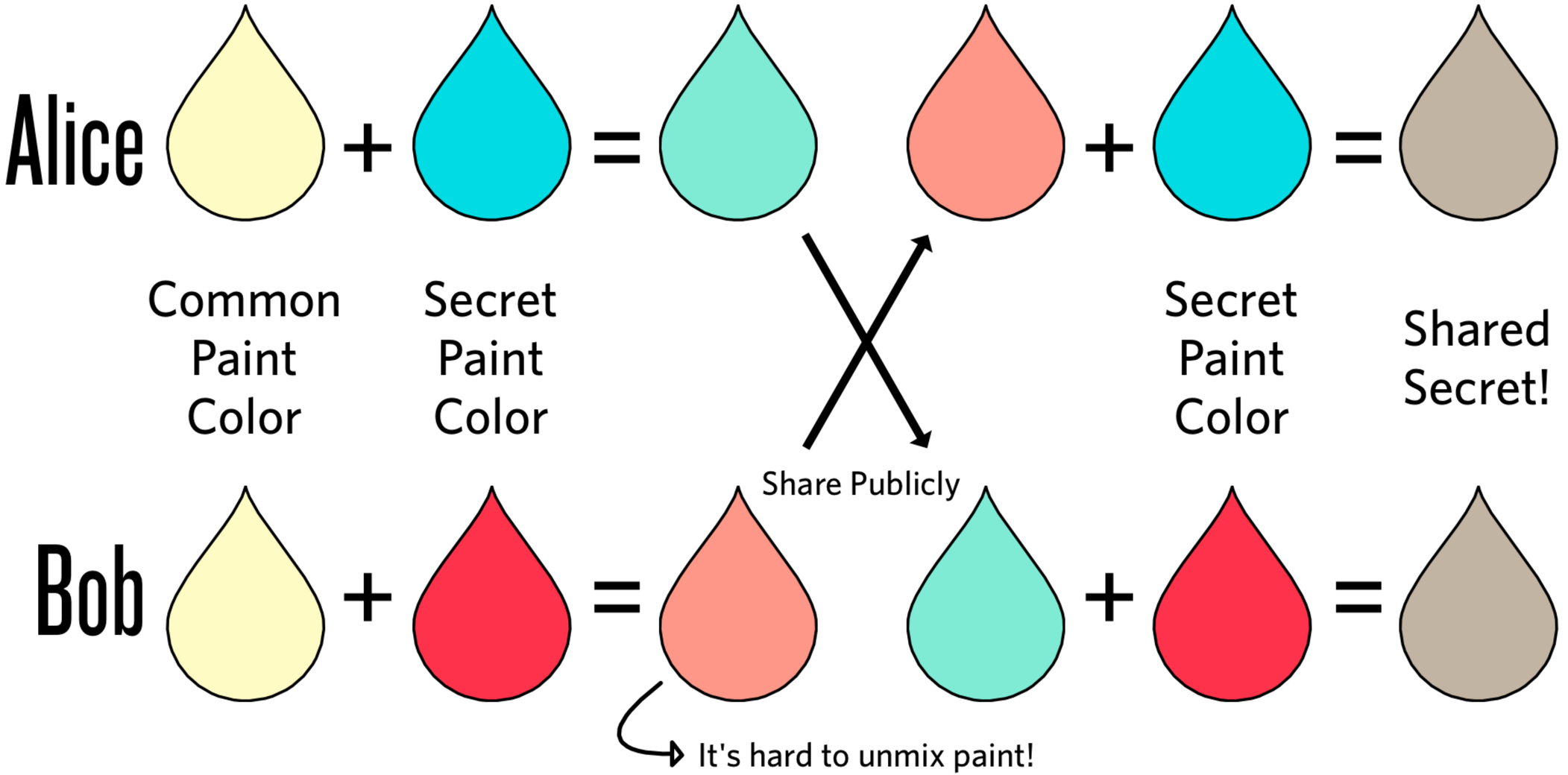
```
# Chosen inputs
p, q, e = 67, 71, 37

# Calculated
n = p*q
x = (p - 1)*(q - 1)
d = inverse_mod(e, x)  # modular multiplicative inverse

public_key = (e, n)
private_key = (d, n)

message = 123
encrypted = pow(message, e, n)
decrypted = pow(encrypted, d, n)  # == message 🤯
```



https://github.com/robinske/rsa-example

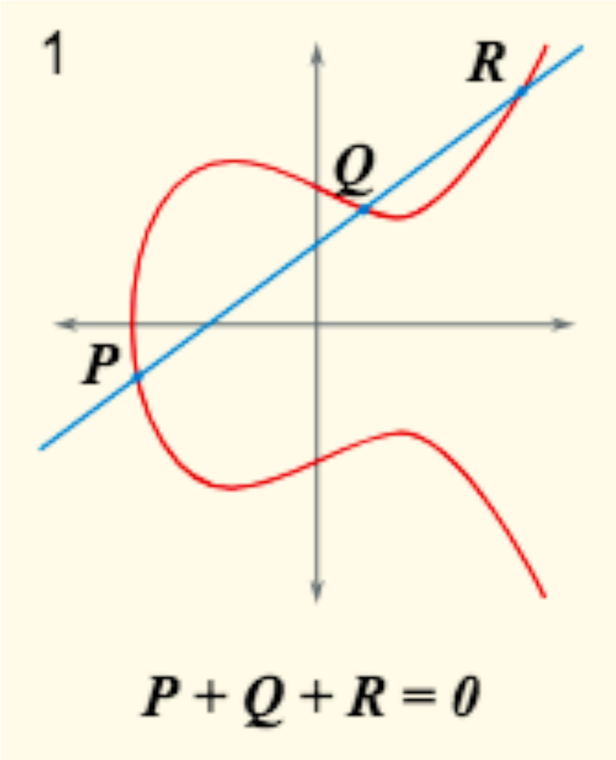
Other Common Algorithms

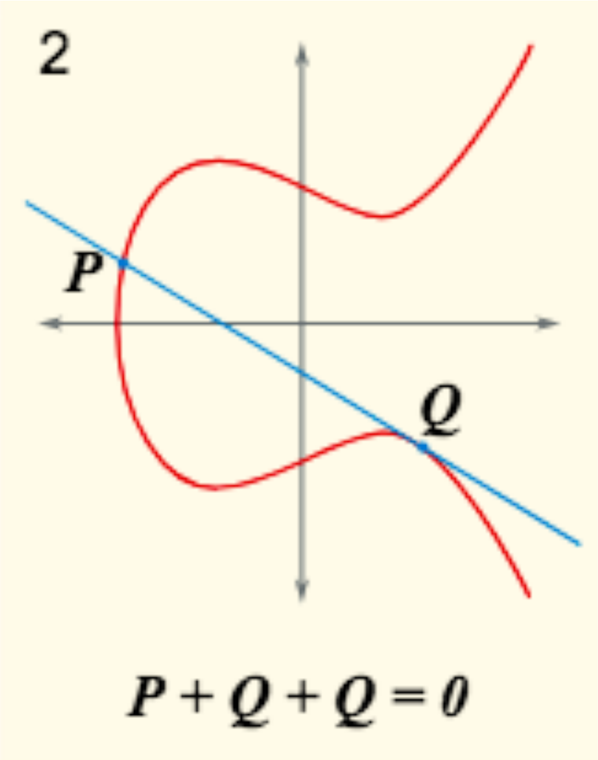
Diffie-Hellman Key Exchange

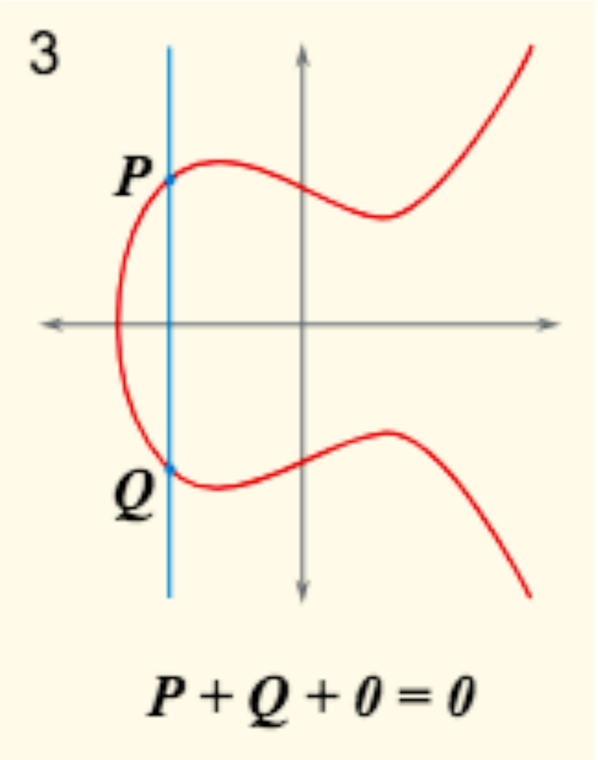


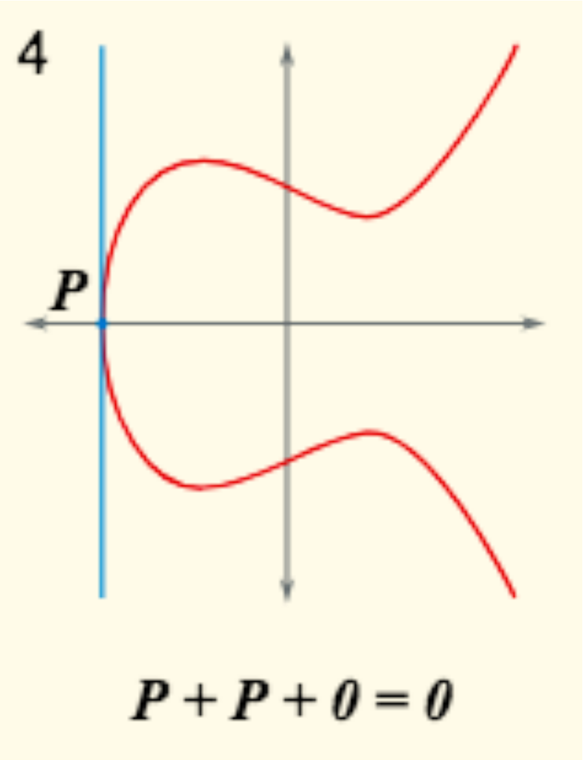
Elliptic-Curve Cryptography (ECC)

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



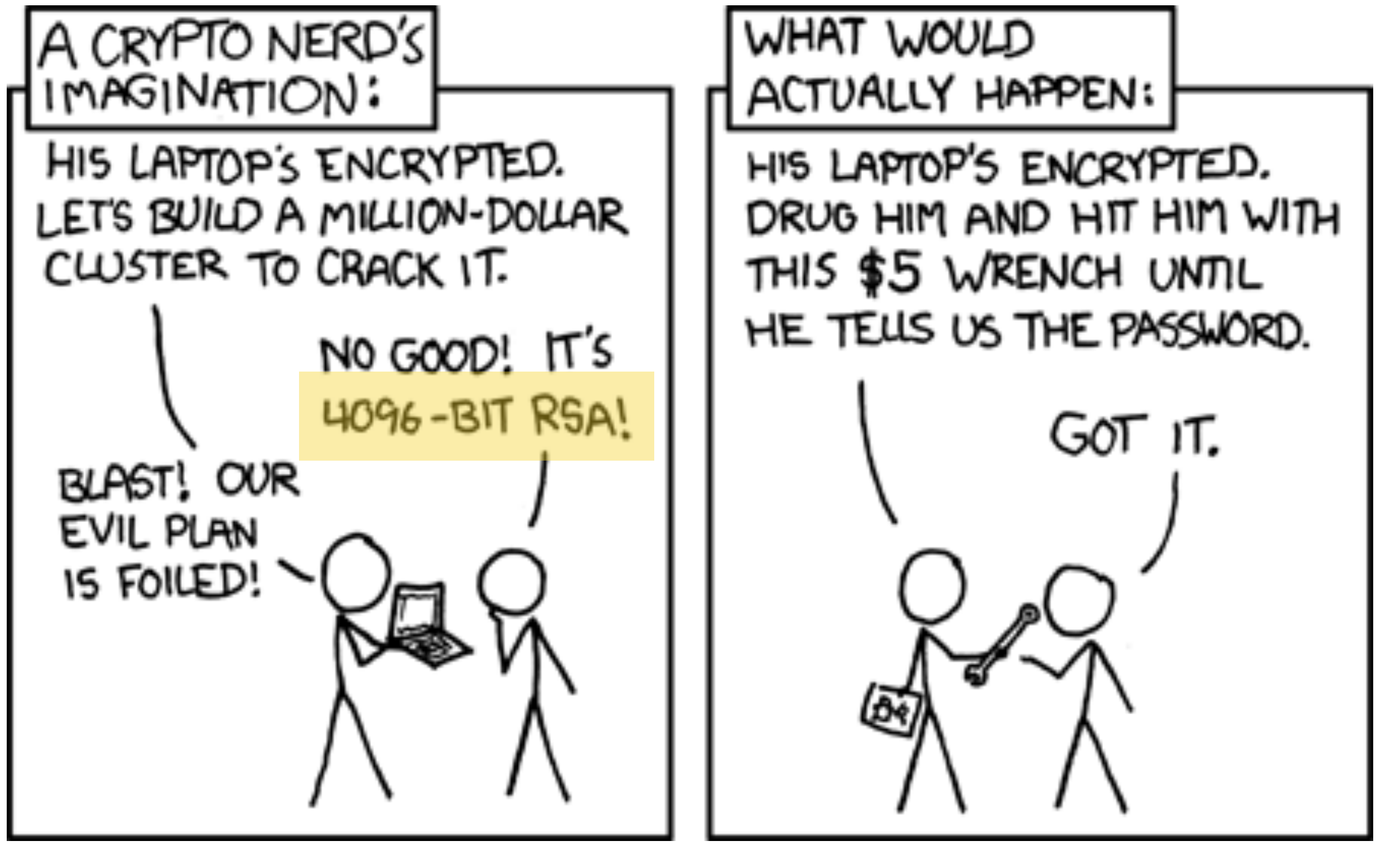






Elliptic Curve Addition (Image By SuperManu [GFDL or CC BY-SA 3.0], via Wikimedia Commons)

What is Key Size?



https://xkcd.com/538/

What is Key Size?

Impacts **security strength**, measured in bits of security

- **RSA key size of 2048 → 112 bits of security**
- **RSA key size of 3072 → 128 bits of security**
- **ECC key size of 256 → 128 bits of security**

What is Security Strength?

> “a number associated with the amount of work that is required to break a cryptographic algorithm or system.”

- NIST Recommendation for Key Management

PKC in Python

PKC in Python

```
# pip install cryptography
from cryptography.hazmat.primitives.asymmetric import rsa

private_key = rsa.generate_private_key(
    public_exponent=65537,
    key_size=2048,
    ...
)
```

PKC in Python

Docs » Primitives » Asymmetric algorithms » RSA

Edit on GitHub

Danger

This is a “Hazardous Materials” module. You should **ONLY** use it if you’re 100% absolutely sure that you know what you’re doing because this module is full of land mines, dragons, and dinosaurs with laser guns.

https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/

Encrypting

```
public_key = private_key.public_key()

ciphertext = public_key.encrypt(message, ...)

plaintext = private_key.decrypt(ciphertext, ...)

# returns True
message == plaintext
```

Signing

```
public_key = private_key.public_key()

signature = private_key.sign(message, ...)

# throws exception if not verified
public_key.verify(signature, message, ...)
```

Everyday Uses of Public Key Cryptography

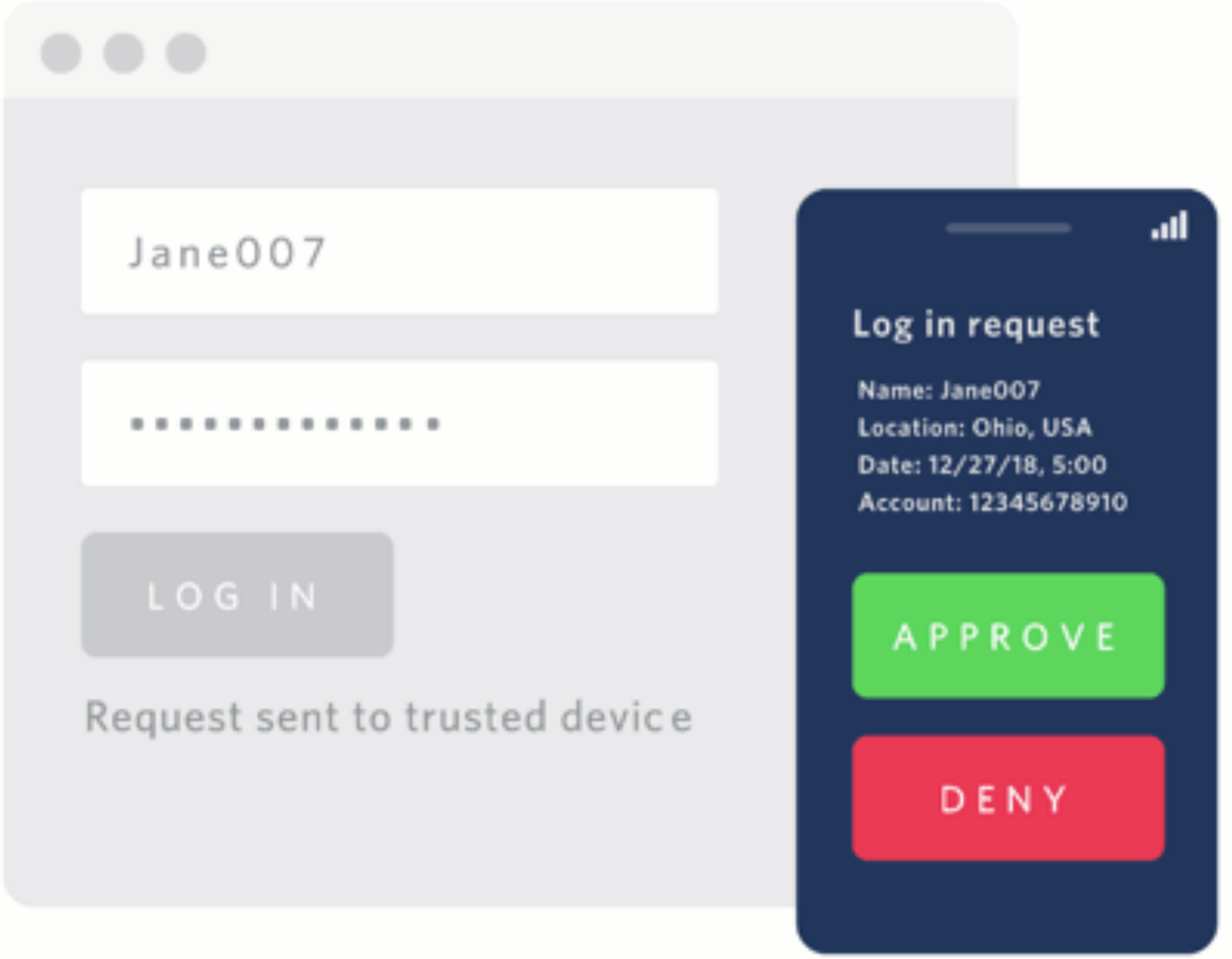
TLS (HTTPS)

Bitcoin

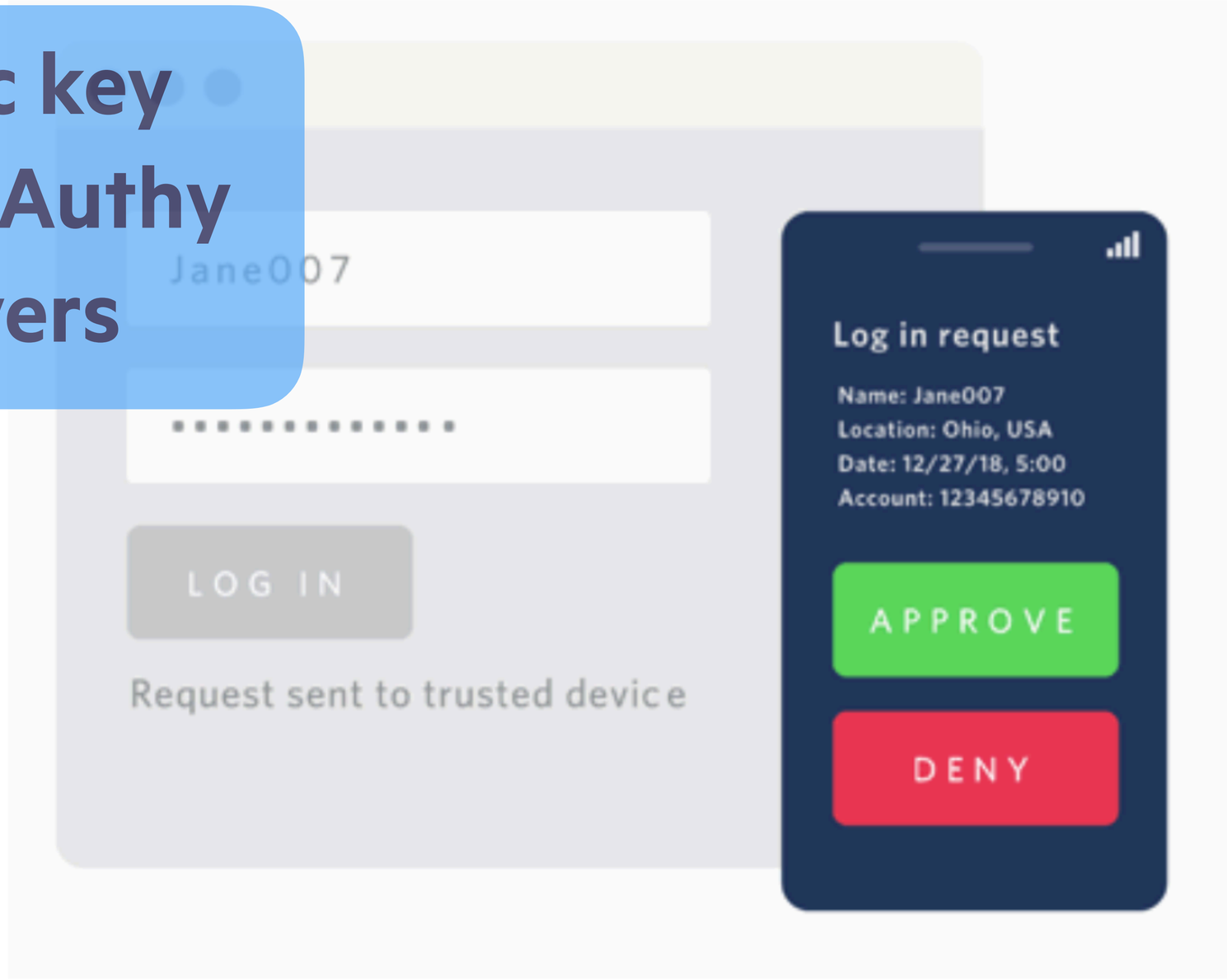
SSH

Authy! 

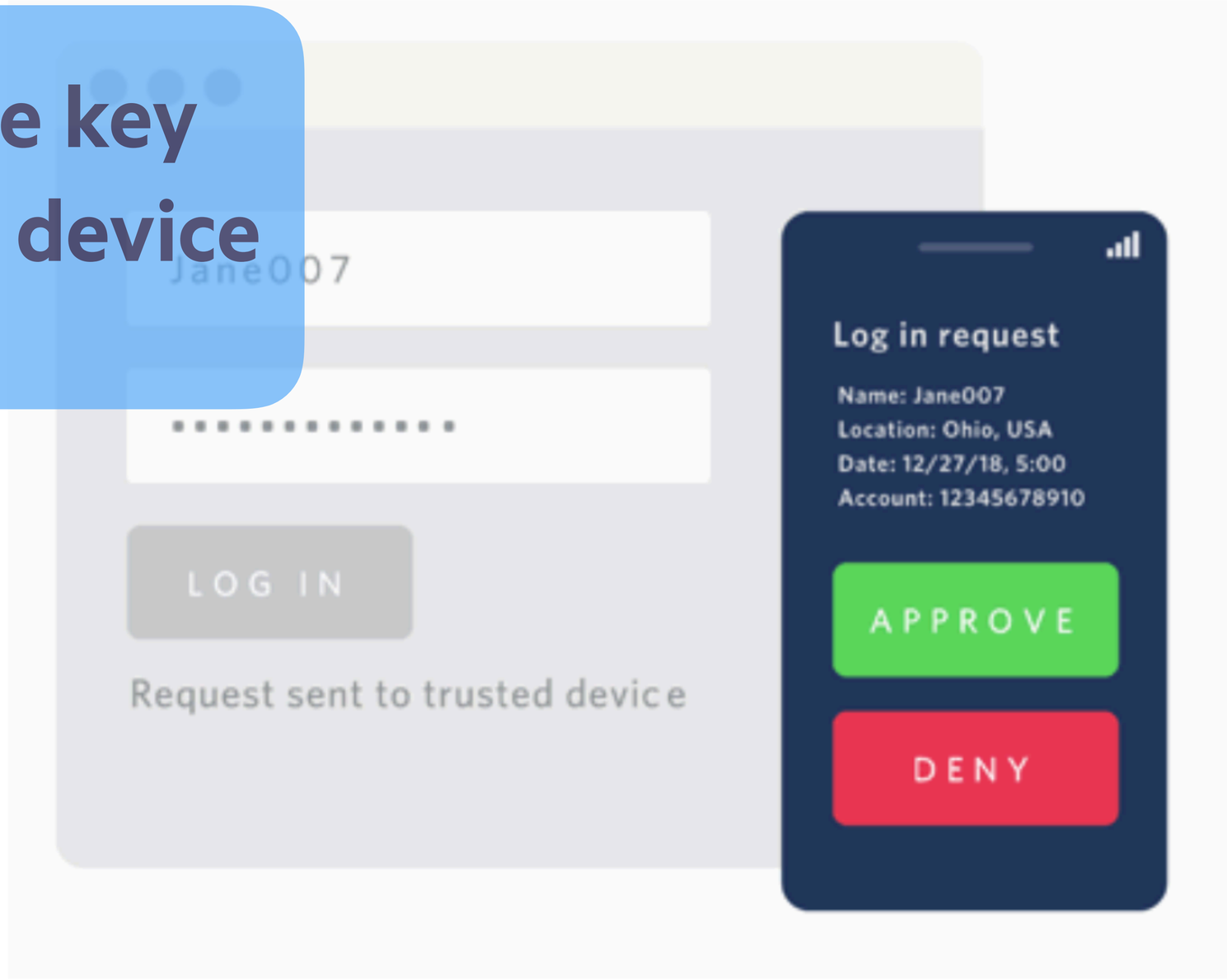
PGP and GPG


Jane007
LOG IN
Request sent to trusted device
Log in request
Name: Jane007
Location: Ohio, USA
Date: 12/27/18, 5:00
Account: 12345678910
APPROVE
DENY

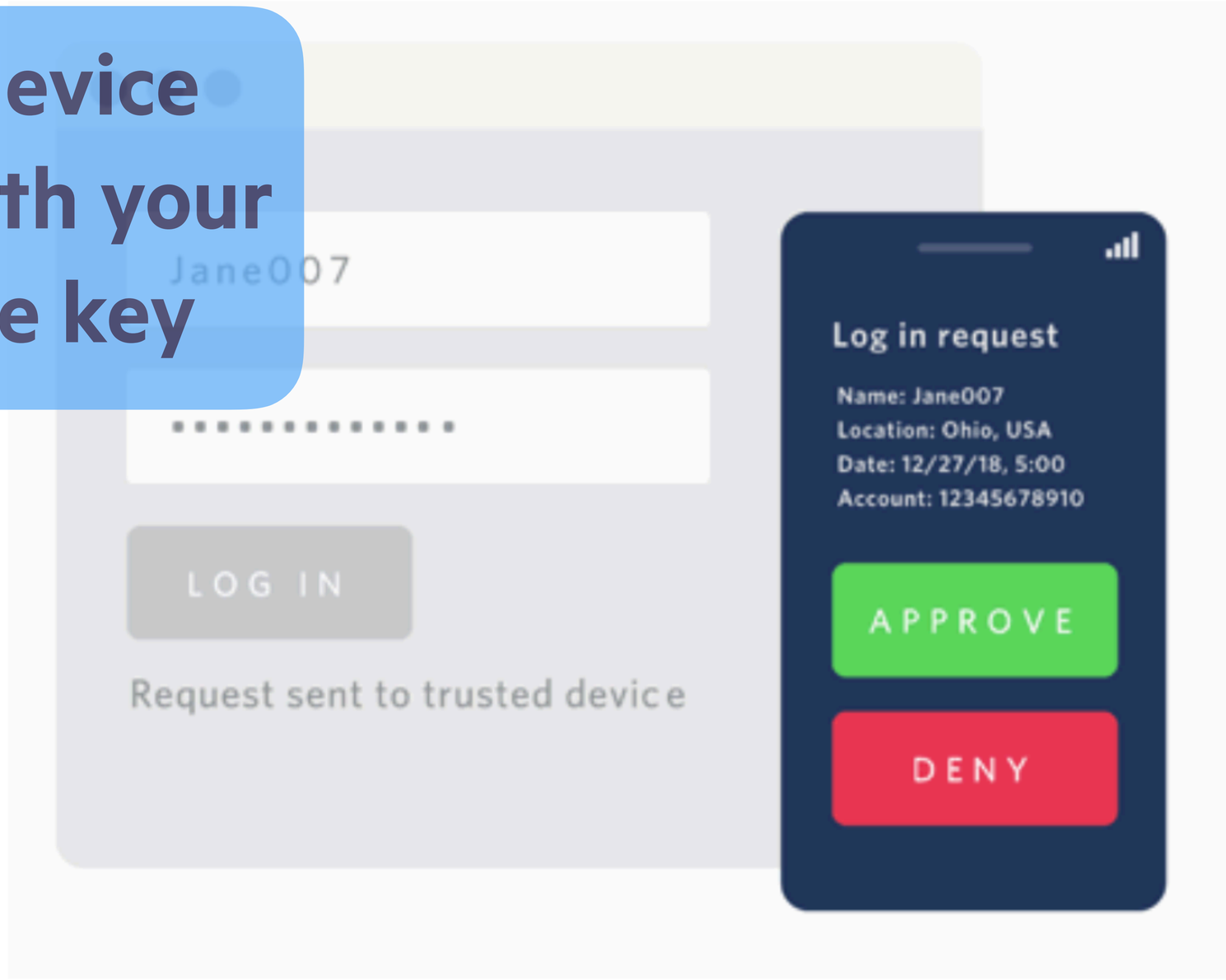
Public key on the Authy servers



Private key on your device



Your device signs with your private key



Generating a new SSH key

1. Open Terminal.

2. Paste the text below, substituting in your GitHub email address.

```
$ ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

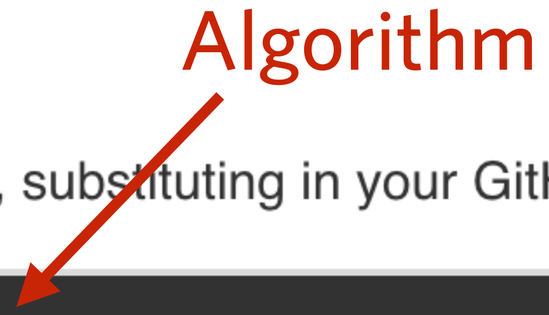
This creates a new ssh key, using the provided email as a label.

```
Generating public/private rsa key pair.
```

Generating a new SSH key

1. Open Terminal.

2. Paste the text below, substituting in your GitHub email address.




```
$ ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

This creates a new ssh key, using the provided email as a label.

```
Generating public/private rsa key pair.
```

Generating a new SSH key

1. Open Terminal.

2. Paste the text below, substituting in your GitHub email address.

This creates a new ssh key, using the provided email as a label.

```
Generating public/private rsa key pair.
```

==========
https://keybase.io/krobs

I hereby claim:

  * I am an admin of http://krobinson.me
  * I am krobs (https://keybase.io/krobs) on keybase.
  * I have a public key ASDqg7DNuP_tpNoFEYp6PTrS4nJHWizO7mI2ZG8rBBbnAgo

To do so, I am signing this object:


```
{
  "body": {
    "key": {
      "eldest_kid": "0120ea83b0cdb8ffeda4da05118a7a3d3ad2e272475a2cceee6236646f2b0416e7020a",
      "host": "keybase.io",
      "kid": "0120ea83b0cdb8ffeda4da05118a7a3d3ad2e272475a2cceee6236646f2b0416e7020a",
      "uid": "b280c9e0287609d97075335041ee6719",
      "username": "krobs"
    },
    "merkle_root": {
      "ctime": 1520889692,
      "hash": "9c6ae9b0faef1c12e983083312cd1616be7ee9643cb0fbe3151cbd659d0a4214a83012c3dad24e75df5b83d94e
      "hash_meta": "d14f786c3b3cfdbb7447cbb43727333d970a6fa3c78d3e800646ca24e0760bbc",
      "seqno": 2226488
    },
    "service": {
      "hostname": "krobinson.me",
      "protocol": "http:"
    },
```

```
    "type": "web_service_binding",
    "version": 1
  },
  "client": {
    "name": "keybase.io go client",
    "version": "1.0.45"
  },
  "ctime": 1520889726,
  "expire_in": 504576000,
  "prev": "7634d1138bb1c1de26e2e2913b67ca43092bd7bddabcbc100ab536872b81fe93",
  "seqno": 7,
  "tag": "signature"
}
```


which yields the signature:


```
hKRib2R5hqhkZXRhY2hlZMOpaGFzaF90eXBlCqNrZXnEIwEg6oOwzbj/7aTaBRGKej060uJyR1oszu5iNmRvKwQW5wIKp3BheWxvYWTFA
TIwZWE4M2IwY2RiOGZmZWRhNGRhMDUxMThhN2EzZDNhZDJlMjcyNDc1YTJjY2VlZTYyMzY2NDZmMmIwNDE2ZTcwMjBhIiwiaG9zdCI6Ir
ZmZWRhNGRhMDUxMThhN2EzZDNhZDJlMjcyNDc1YTJjY2VlZTYyMzY2NDZmMmIwNDE2ZTcwMjBhIiwidWlkIjoiYjI4MGM5ZTAyODc2MDI
yb2JzIn0sIm1lcmtsZV9yb290Ijp7ImN0aW1lIjoxNTIwODg5NjkyLCJoYXNoIjoiOWM2YWU5YjBmYWVmMWMxMmU5ODMwODMzMTJjZDE2
NGE4MzAxMmMzZGFkMjRlNzVkZjViODNkOTRlM2JmMzY0MzYxMTg3ODVhOTNlMmNlYzVkZTA3ZmZkMmZlMTBkMGUiLCJoYXNoX2lldGEiO
Dk3MGE2ZmEzYzc4ZDNlODAwNjQ2Y2EyNGUwNzYwYmJjIiwic2Vxbm8iOjIyMjY0ODh9LCJzZXJ2aWNlIjp7Imhvc3RuYW1lIjoia3JvYn
BlIjoid2ViX3NlcnZpY2VfYmluZGluZyIsInZlcnNpb24iOjF9LCJjbGllbnQiOnsibmFtZSI6ImtleWJhc2UuaW8gZ28gY2xpZW50Iiw
4OTcyNiwiZXhwaXJlX2luIjo1MDQ1NzYwMDAsInByZXYiOiI3NjM0ZDExMzhiYjFjMWRlMjZlMmUyOTEzYjY3Y2E0MzA5MmJkN2JkZGFi
InRhZyI6InNpZ25hdHVyZSJ9o3NpZ8RAwJoG6GA7p5VoC+zxVd8OimSUbowBjy9a7Qot4if3KAFEySTcPkORFuVVxxQhreKhW2JtFFlbh
mFsdWXEIKz6i24XAc0AO9vCJp3xecqfyMjPdp4P5G4FAQG2LGHWo3RhZ80CAqd2ZXJzaW9uAQ==
```

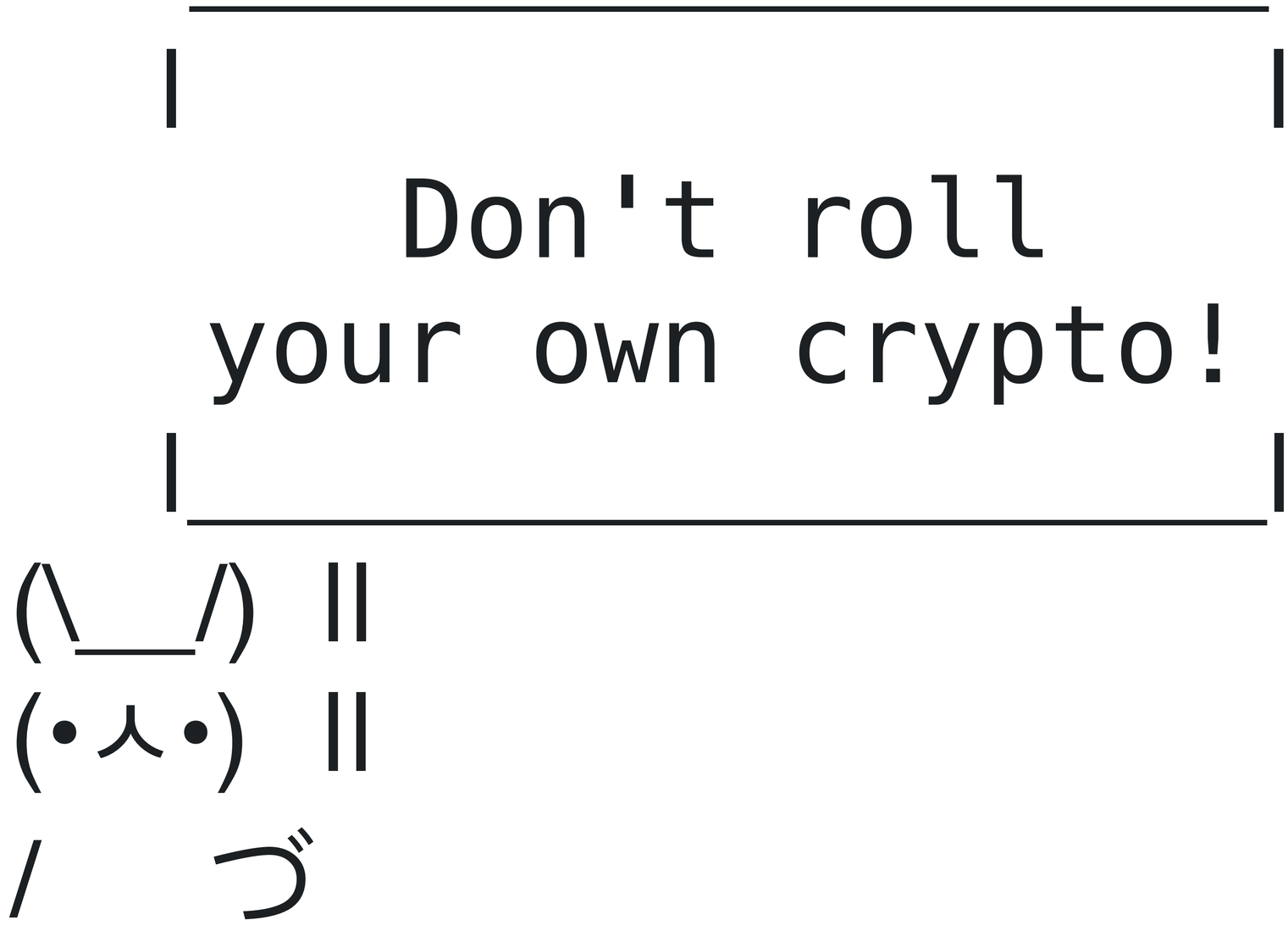

And finally, I am proving ownership of this host by posting or
appending to this document.

View my publicly-auditable identity here: https://keybase.io/krobs

HOW TO USE PGP TO VERIFY THAT AN EMAIL IS AUTHENTIC:

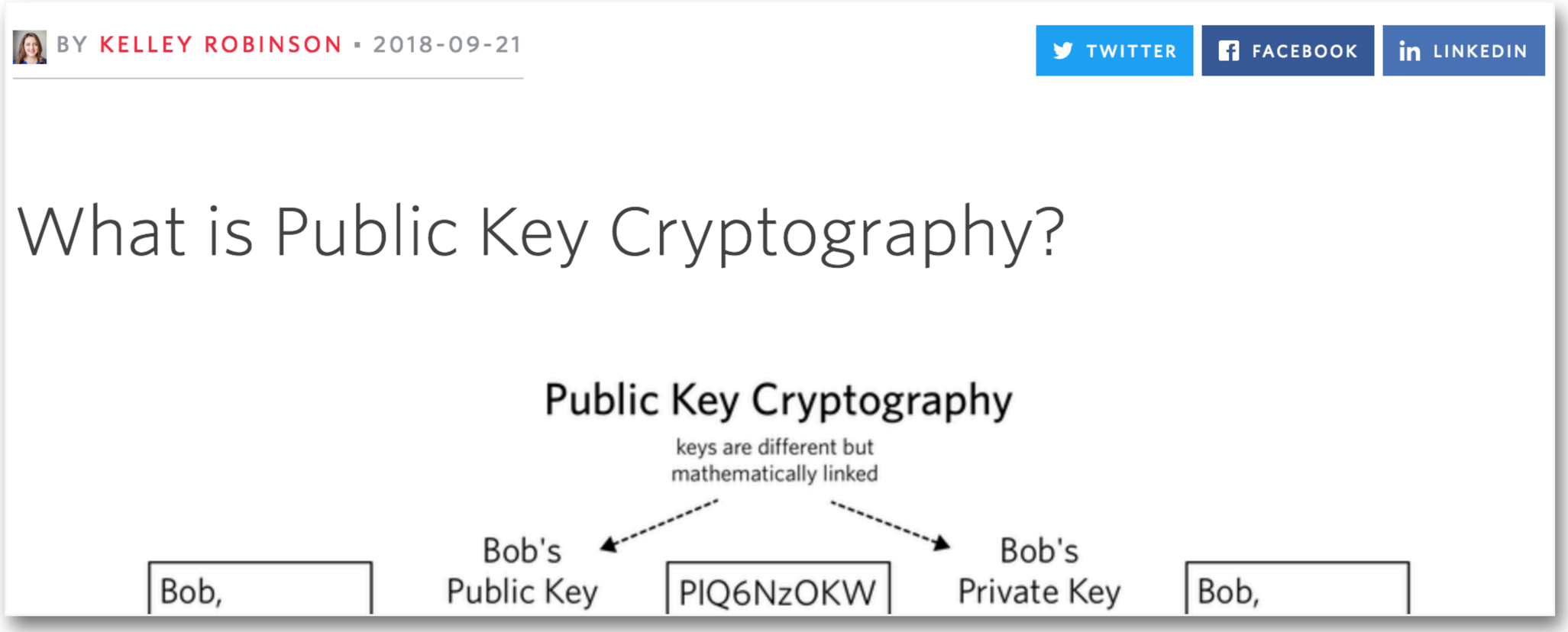
IF IT'S THERE, THE EMAIL IS PROBABLY FINE.

https://xkcd.com/1181/


Don't roll
your own crypto!
(__/) ||
(•ㅅ•) ||
/ づ

@kelleyrobinson

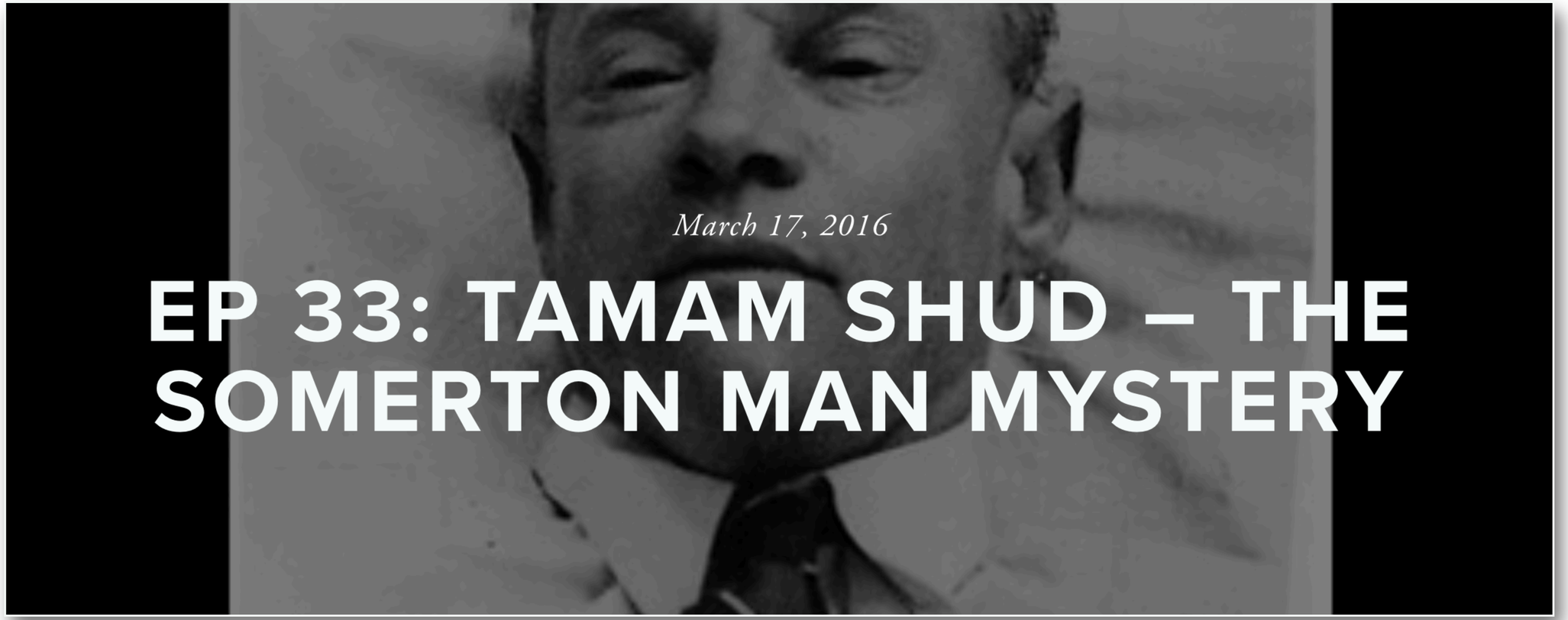
Further Reading



twilio.com/blog/what-is-public-key-cryptography

@kelleyrobinson

Further Listening



astonishinglegends.com

THANK YOU!

@kelleyrobinson

twilio

goto; chicago

Please

Remember to rate this session

Thank you!

follow us @gotochgo