Security and Privacy of Internet Applications (Module 5)

Basics of Cryptography

CRYPTOLOGY

Cryptology is a science that studies:

Cryptography & Cryptanalysis

CRYPTOLOGY

"Cryptography"

The point of cryptography is to send secure messages over an insecure medium (like the Internet)

Turning plaintext (an ordinary readable message) into ciphertext (secret messages that are "hard" to read)

CRYPTOLOGY

"Cryptanalysis"

Cryptanalysis studies cryptographic systems to look for weaknesses or leaks of information

Breaking secret messages

Recovering the plaintext from the ciphertext

THE SCOPE

The goal of the cryptography unit in this course is to show you what cryptographic tools exist, and information about using these tools in a secure manner

We won't be showing you details of how the tools work

THE MAIN CAST

Honest communicating parties

adversary

BUILDING BLOCKS

There are three main components of Cryptography:

Confidentiality Integrity and Authenticity

STRONG CRYPTOSYSTEMS

"Confidentiality"

- Preventing Eve from reading Alice's messages

BUILDING BLOCKS

There are three main components of Cryptography:

Confidentiality Integrity and Authenticity

STRONG CRYPTOSYSTEMS

"Integrity"

Preventing Eve from modifying Alice's messages without being detected

BUILDING BLOCKS

There are three main components of Cryptography:

Confidentiality Integrity and Authenticity

STRONG CRYPTOSYSTEMS

"Authenticity"

- Preventing Eve from impersonating Alice

STRONG CRYPTOSYSTEMS

What are the ideal properties of a Strong Cryptosystem?

Never Have a SECRET Encryption Algorithm

KIRCHHOFF'S PRINCIPLE

A Crypto System should be secure, even if everything about the system, except the key is public knowledge.

SHANNON'S MAXIM

One ought to design systems under the assumption that the enemy will immediately gain full familiarity with them.

SHANNON'S MAXIM

So don't use secret encryption methods (security by obscurity)

Have public algorithms that use a secret key as input

It's easy to change the key: it's usually just a smallish number

STRONG CRYPTOSYSTEMS

Eve may know the Algorithm Some part of plain text

plain text cipher text pairs has access to

encryption/decryption oracle

STRONG CRYPTOSYSTEMS

"Algorithm"

- The cryptographic algorithm is always public.
  it can AES, RSA etc.

STRONG CRYPTOSYSTEMS

Eve may know the Algorithm Some part of plain text

plain text cipher text pairs has access to

encryption/decryption oracle

STRONG CRYPTOSYSTEMS

“ Some part of plain text ”

Should be resistant to known plain text attack

Pattern Matching

STRONG CRYPTOSYSTEMS

Eve may know the Algorithm Some part of plain text

plain text cipher text pairs has access to

encryption/decryption oracle

STRONG CRYPTOSYSTEMS

“plain text cipher text pairs”

Should be resistant to known plain text attack
Should be resistant to known cipher text attack

STRONG CRYPTOSYSTEMS

Eve may know the Algorithm Some part of plain text

plain text cipher text pairs has access to

encryption/decryption oracle

STRONG CRYPTOSYSTEMS

"encryption/decryption oracle"

- Should be resistant to chosen plain text attack
- Should be resistant to chosen cipher text attack

STRONG CRYPTOSYSTEMS

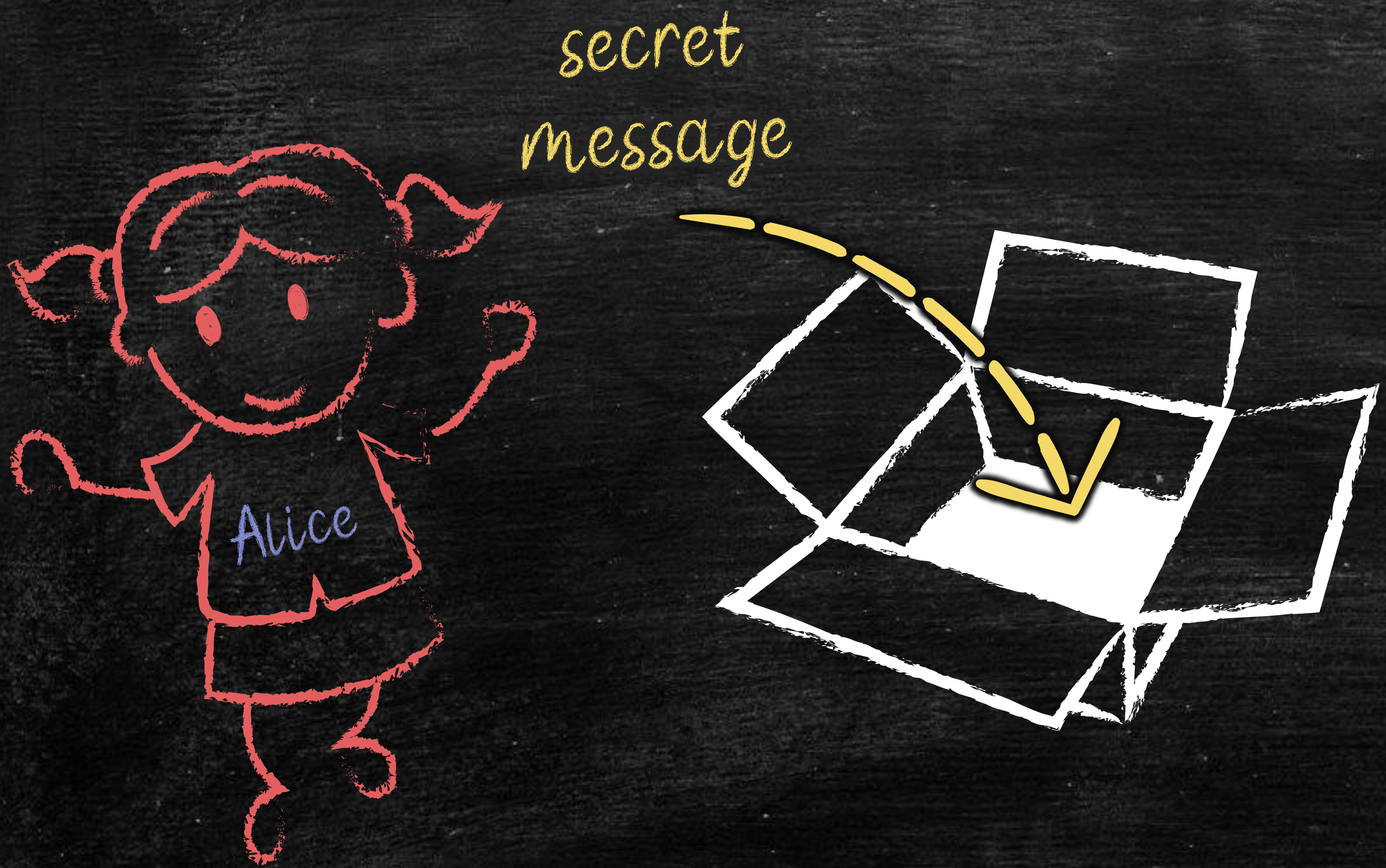
Eve may know the Algorithm Some part of plain text

plain text cipher text pairs has access to encryption/decryption oracle

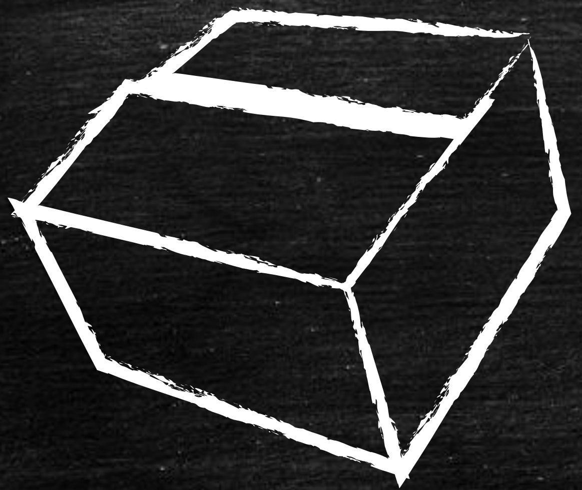
However,

Eve may not know the Key!!

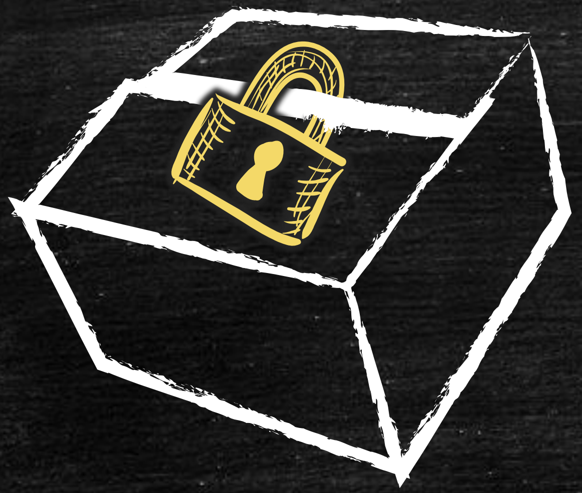
ENCRYPTION



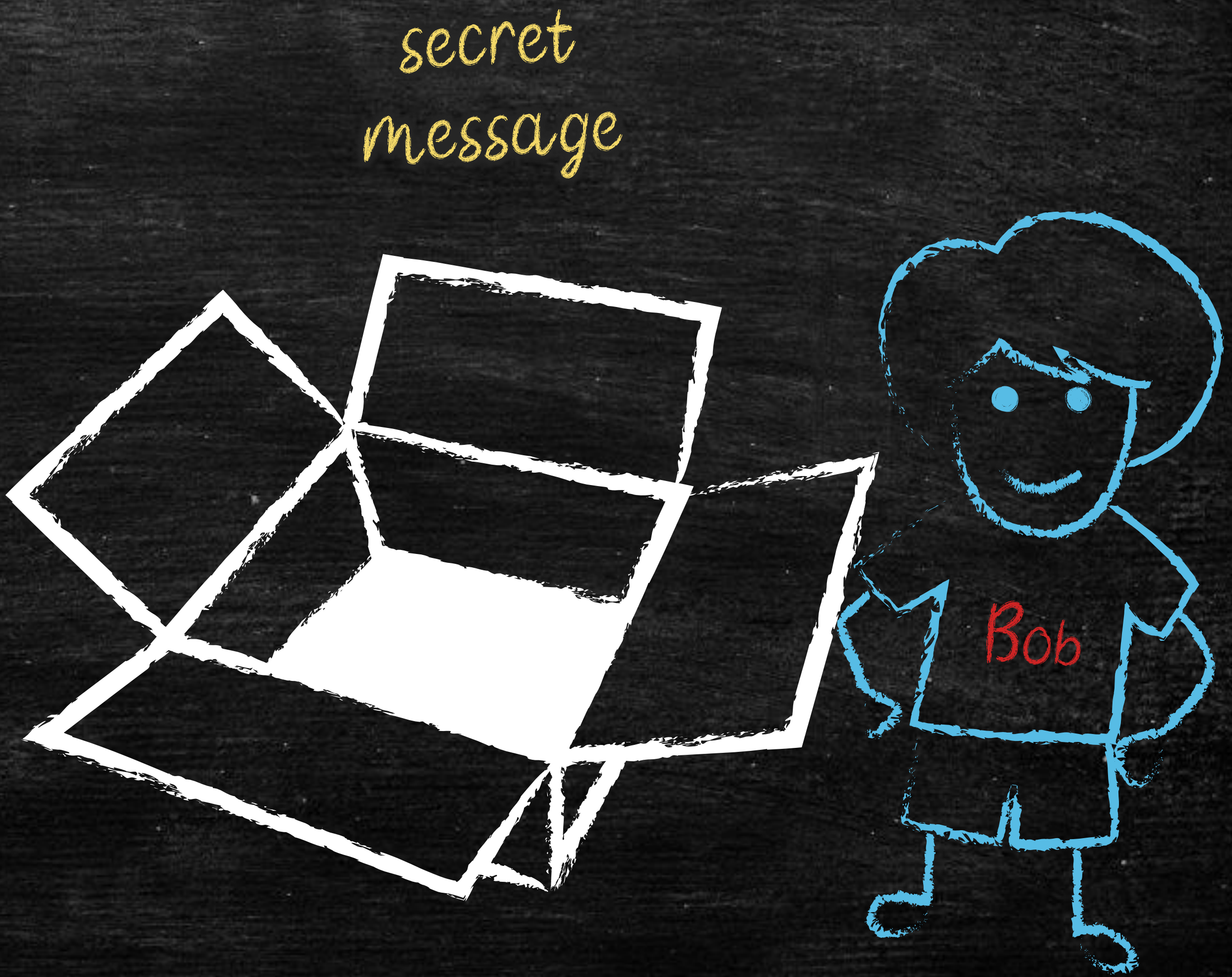
ENCRYPTION



ENCRYPTION



ENCRYPTION



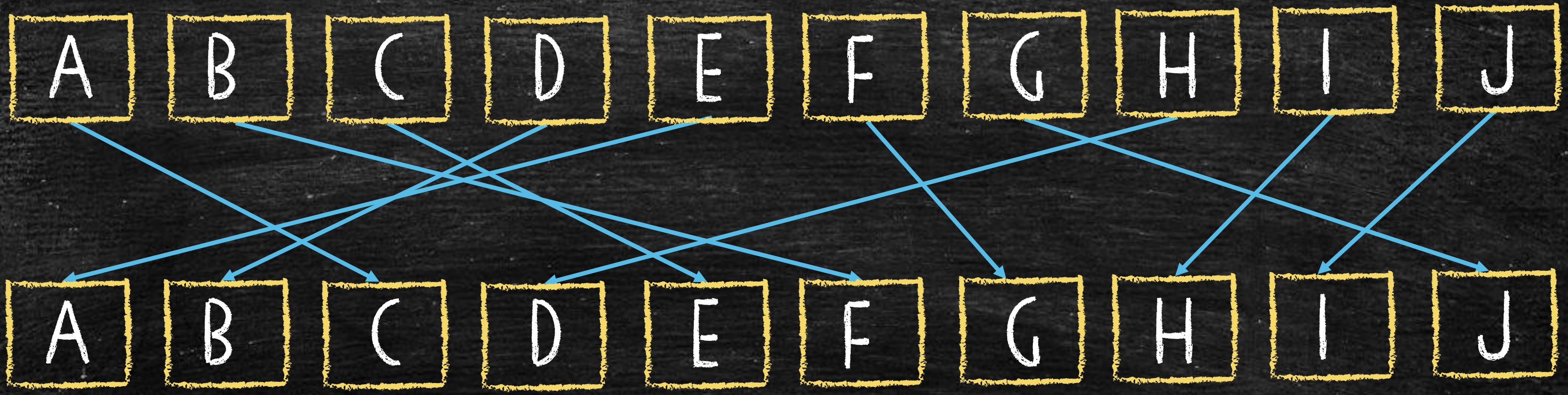
Encryption is the digital analog of the preceding scenario

An encryption scheme has three algorithms:

1. Gen creates keys
2. Enc locks messages under a given key
3. Dec unlocks messages using associated the key

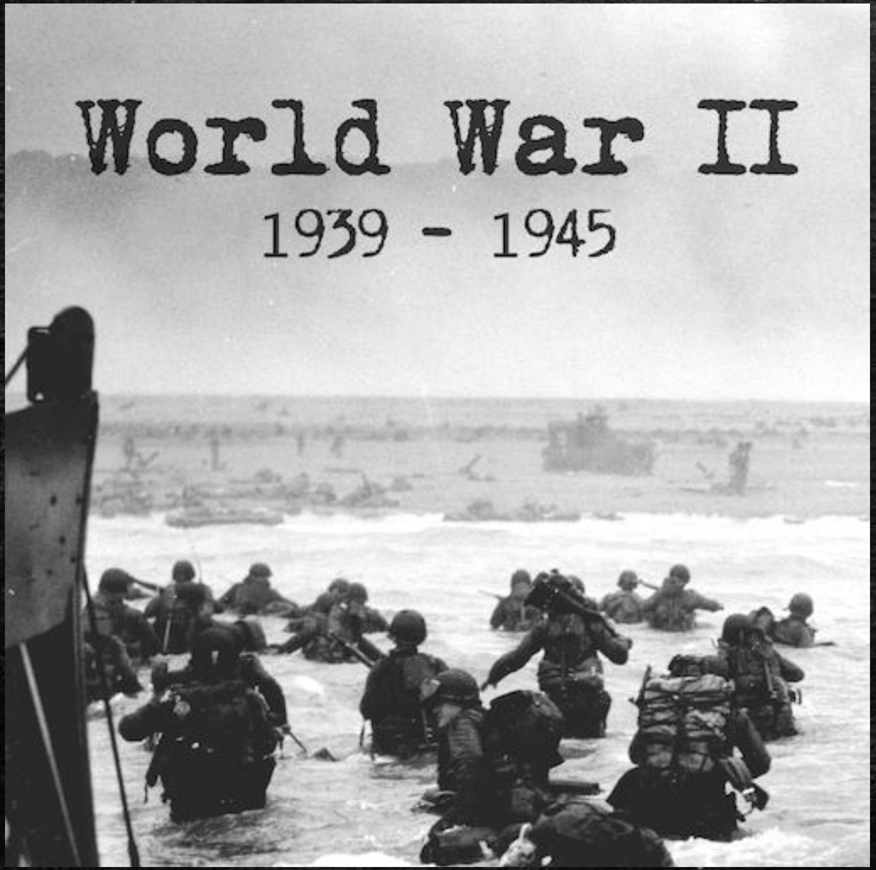
Let us turn the Clock Back

OLD CIPHERS



Caeser Cipher

CRYPTOGRAPHY IN WORLD WAR II



Caeser Cipher

But … A very sophisticated one

THE ENIGMA MACHINE

Caeser Cipher

But … A very sophisticated one

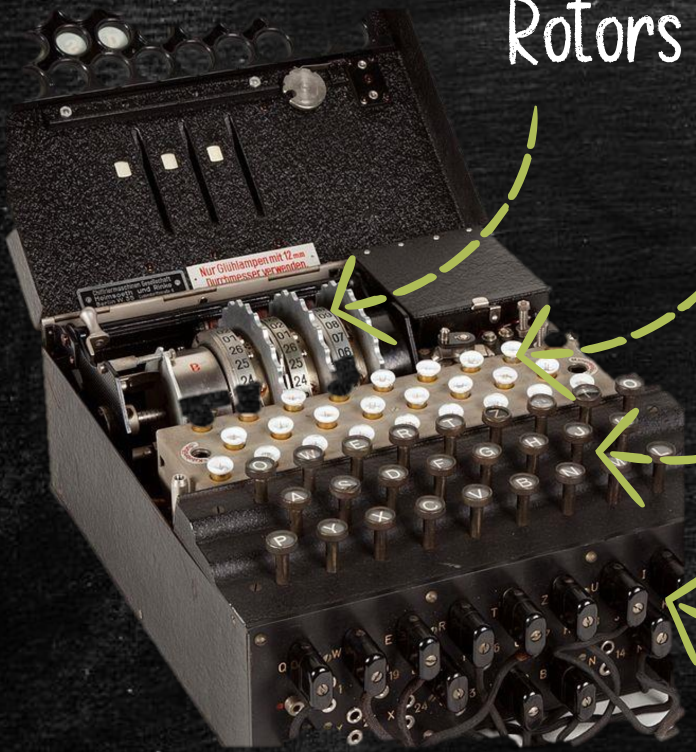
See this amazing lecture on the Enigma Machine

THE ENIGMA MACHINE

Caeser Cipher

But … A very sophisticated one

THE ENIGMA MACHINE



Caeser Cipher

But ... A very sophisticated one

THE ENIGMA MACHINE

1.07×10^{23} different ways to encrypt
(comparable with a 77-bit key)

26×26×26=17,576 (Number of possible starting positions)

1.07×10^{23} different ways to encrypt
(comparable with a 77-bit key)

THE ENIGMA MACHINE

1.07×10^{23} different ways to encrypt (comparable with a 77-bit key)

But this is not good by modern standards. Why??

"FLAW" IN THE ENIGMA

A major flaw in the Enigma was that a letter did not map to itself

This allowed for some Cryptanalysis by:

Marian Rejewski

Also Read

"FLAW" IN THE ENIGMA

- How can you exploit something like this?

BREAKING THE ENIGMA CODE

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

W E T T E R

Check out this paper: https://web.archive.org/web/20060720040135/http://members.fortunecity.com/jpeschel/gillog1.htm

BREAKING THE ENIGMA CODE

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

W E T T E R

Check out this paper

BREAKING THE ENIGMA CODE

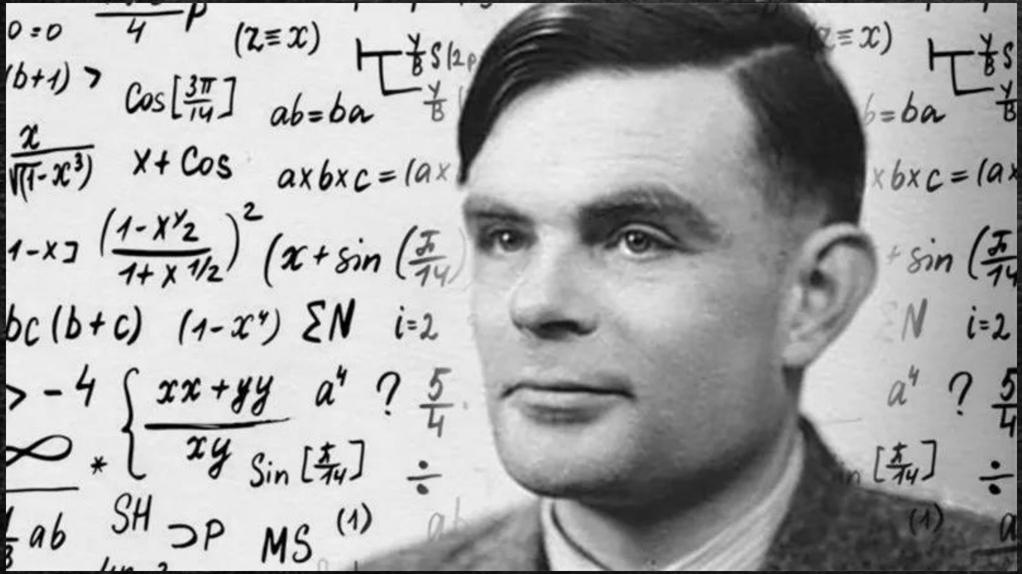
W J E Q L D U Y B N H J X P K Z V C F G R A

W E T T E R

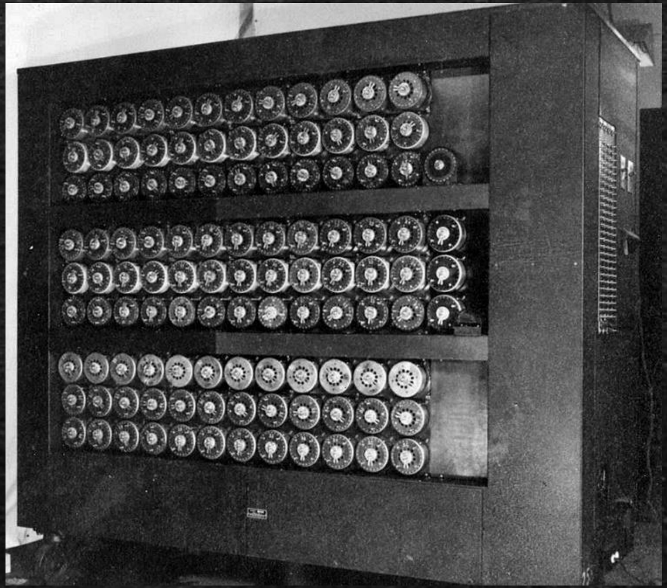
This is a possible location for the word "Wetter"

Check out this paper: https://web.archive.org/web/20060720040135/http://members.fortunecity.com/jpeschel/gillog1.htm

BREAKING THE ENIGMA CODE



Alan Turing



Bombe Machine

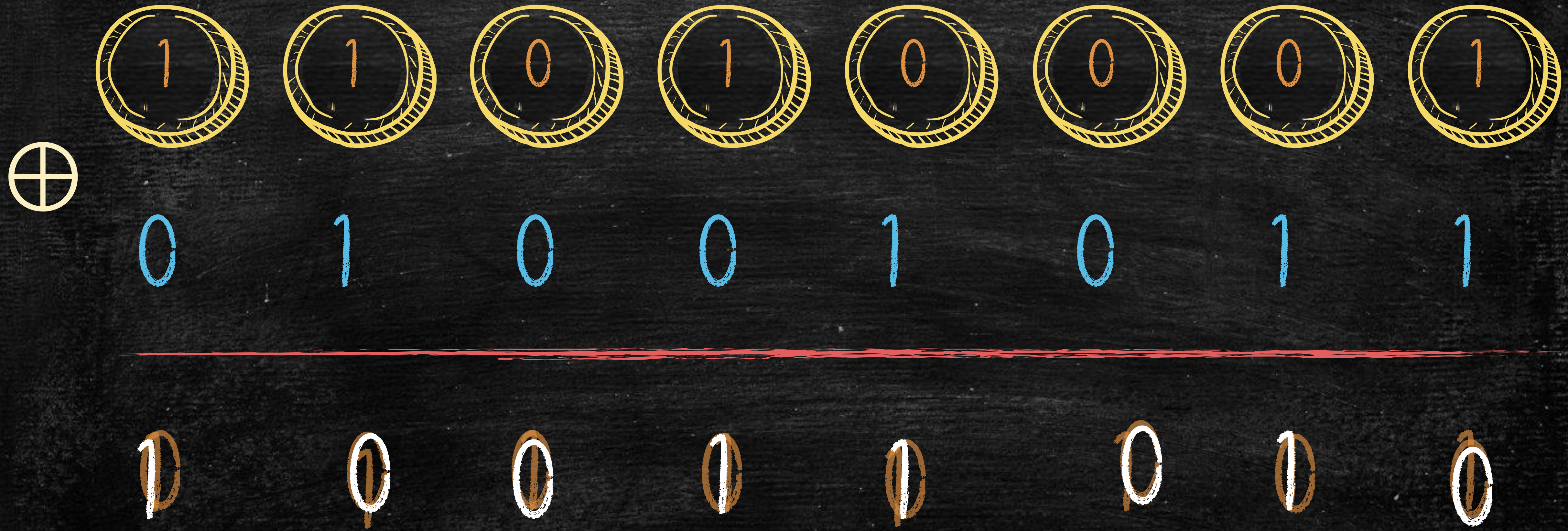
Bombe Machine Broke the Enigma Code in 20 minutes

Modern Cryptography

Secret Key Cryptography

GILBERT VERNAM (1890-1960)

- Engineer at AT&T Bell Labs
- "Invented" stream ciphers and the one-time pad (OTP) in 1919
- U.S. Patent 1,310,719
  - Actually, the patent was for a machine that encrypts a plaintext by *mechanically* XORing it with a secret key


1 1 0 1 0 0 0 1
⊕
0 1 0 0 1 0 1 1
1 0 0 1 1 0 1 0

VERNAM'S ONE-TIME PAD

$Gen(1^n)$ generates keys (referred to as pads)

Input: an integer ⇒ length of key

Output: a random n-bit string, $k \in \{0, 1\}^n$

truly random

VERNAM'S ONE-TIME PAD

Enc(k; m) encrypts m with pad k

Input: a pad k and message m of the exact same length (i.e., m, $k \in \{0, 1\}^n$)

Output: the bitwise XOR, $c \in \{0, 1\}^n$, of m and k (i.e., $c = m \oplus k$)

VERNAM'S ONE-TIME PAD

Dec(k, c) decrypts c with pad k

Input: a pad k and ciphertext c of the exact same length

(i.e., c k$\in\{0, 1\}^n$)

Output: the bitwise XOR, $m'\in\{0, 1\}^n$, of c and k

(i.e., $m' = c \oplus k$)

VERNAM'S ONE-TIME PAD

Provides Information Theoretic Security

No matter how computationally strong the adversary is OTP cannot be broken.

ONE TIME PADS

- Why does *try every key* not work here?
  - Because, given a ciphertext C for every possible message M, there exists a K that could have generated that cipher text?
- Does it provide integrity?
  - Nope! An adversary can flip the bits of the cipher text.

PROBLEM WITH ONE TIME PADS

- If your boss stores your salary (in binary) encrypted using One Time Pad what can you do with the cipher text?

- You can XOR a "1000000 …. ". This flips the most significant bit, which is most likely a zero!

PROBLEM WITH ONE TIME PADS

- 😟 The key must be truly random.
- 😟 Another problem is that key should be size of the message.
- 😟 The key must not be used more than once.
  Two Time Pads do not work.

Issues with Two-Time Pads

- What happens if you use the same key to encrypt two messages?
- Messages are not purely random.

$C_1 = M_1 \oplus K, \quad C_2 = M_2 \oplus K$

$C_1 \oplus C_2 = (M_1 \oplus K) \oplus (M_2 \oplus K) = (M_1 \oplus M_2)$

COMPUTATIONAL SECURITY

In contrast to One-Time Pad's perfect or Information Theoretic security, most cryptosystems have computational security.

This means that it is certain they can be broken by enough work by Eve

We want: Enough == NOT practical

40-BIT CRYPTOGRAPHY

This was the US legal export limit for a long time
(cryptosystems were classified as munitions unit the late 90's)

$2^{40} \sim 10^{12}$ possible keys

Computer	$\sim 10^{7}$ key per second	18 hours
Lab	$\sim 10^{9}$ key per second	11 hours
Bitcoin Network	$\sim 10^{20}$ key per second	4.2 ns

56-BIT CRYPTOGRAPHY

This was the US Government Standard (DES) for a long time

$2^{56} \sim 7.2 \times 10^{26}$ possible keys

Computer	$\sim 10^{7}$ key per second	134 years
Lab	$\sim 10^{9}$ key per second	16 months
Bitcoin Network	$\sim 10^{20}$ key per second	0.22 ms

128-BIT CRYPTOGRAPHY

This is the modern Standard

$2^{128} \sim 10^{38}$ possible keys

Computer	$\sim 10^{7}$ key per second	$6.3 \times \sim 10^{23}$ years
Lab	$\sim 10^{9}$ key per second	$6.3 \times \sim 10^{21}$ years
Bitcoin Network	$\sim 10^{20}$ key per second	$4.1 \times \sim 10^{10}$ years

128-BIT CRYPTO CAN'T BE BROKEN?

What about Quantum Computers?

They will not really help

What about Moore's Law?

If we believe Moore's Law after 132 years we'll have computers that break 128-bit Crypto in 18 hours

TYPES OF SECRET-KEY CRYPTOSYSTEMS

There are two main types of secret-key cryptosystems:

Stream Ciphers & Block Ciphers

TYPES OF SECRET-KEY CRYPTOSYSTEMS

Stream Ciphers

A stream cipher operates one bit at a time. Basically, take the One-Time Pad, but use a pseudorandom keystream instead of a truly random one.

TYPES OF SECRET-KEY CRYPTOSYSTEMS

Stream Ciphers

Can be very fast! And can allow us to send a lot of data securely

We saw the issue with re-using a key (two-time pad)

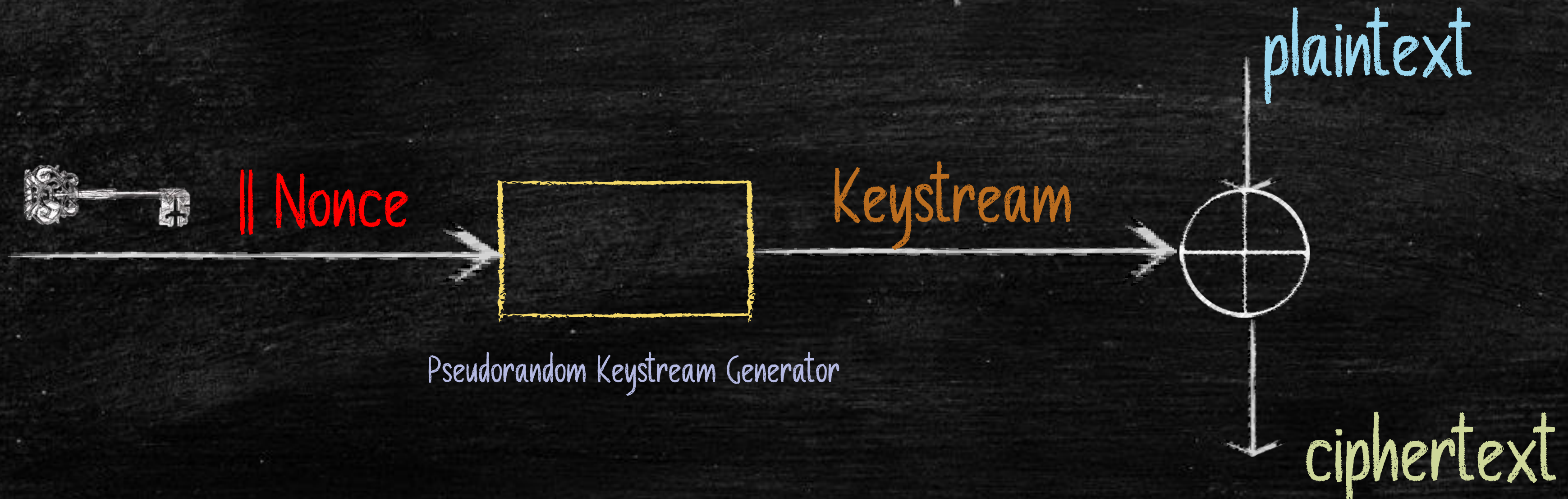
WEP, PPTP are great examples of how NOT use stream ciphers.

Concatenate the key with nonce.

TYPES OF SECRET-KEY CRYPTOSYSTEMS

Stream Ciphers

- RC4 was the most common stream cipher (now deprecated)
- ChaCha increasingly popular, and SNOW3G in mobile phones



TYPES OF SECRET-KEY CRYPTOSYSTEMS

Stream Ciphers

? What happens in a stream cipher if you flip just one bit of the plain text?

⚠ The corresponding bit of the cipher text is flipped.
"Bit-flipping attacks"

? Have we already seen a bit-flipping attack in the class?

Yes, You increased your salary!

TYPES OF SECRET-KEY CRYPTOSYSTEMS

There are two main types of secret-key cryptosystems:

Stream Ciphers & Block Ciphers

TYPES OF SECRET-KEY CRYPTOSYSTEMS

Block Ciphers

Operate on the messages one block at a time.
Blocks are usually 64 or 128 bits long.

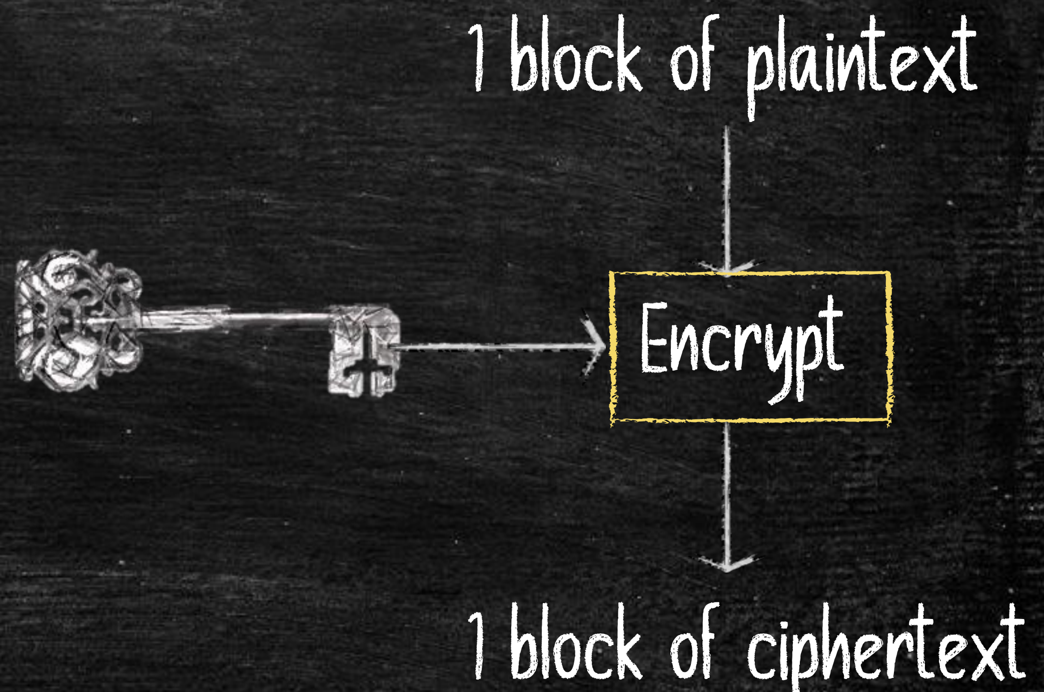
Example: AES is a block cipher everyone should use today.

(unless you have a really good reason)

BLOCK CIPHERS

Block Ciphers

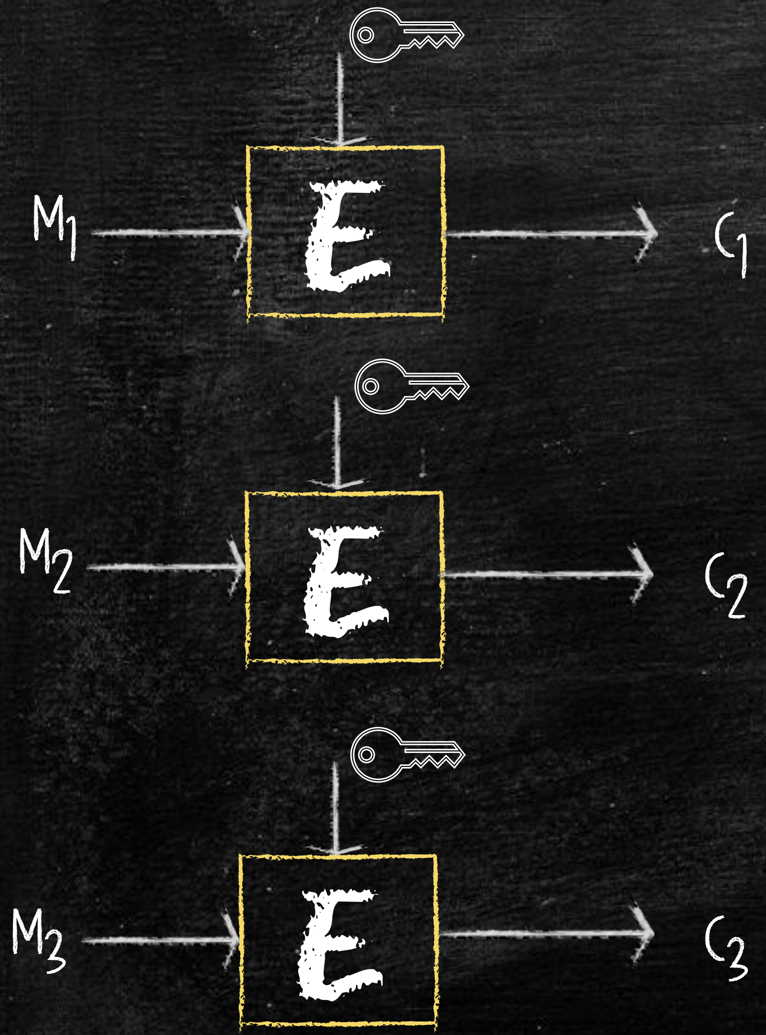
Operate on the messages one block at a time.
Blocks are usually 64 or 128 bits long.



If plaintext is smaller than one block: padding

If plaintext is larger than one block: The choice of what to do with multiple blocks is called the mode of operation of the block cipher

ELECTRONIC CODE BOOK (ECB) MODE

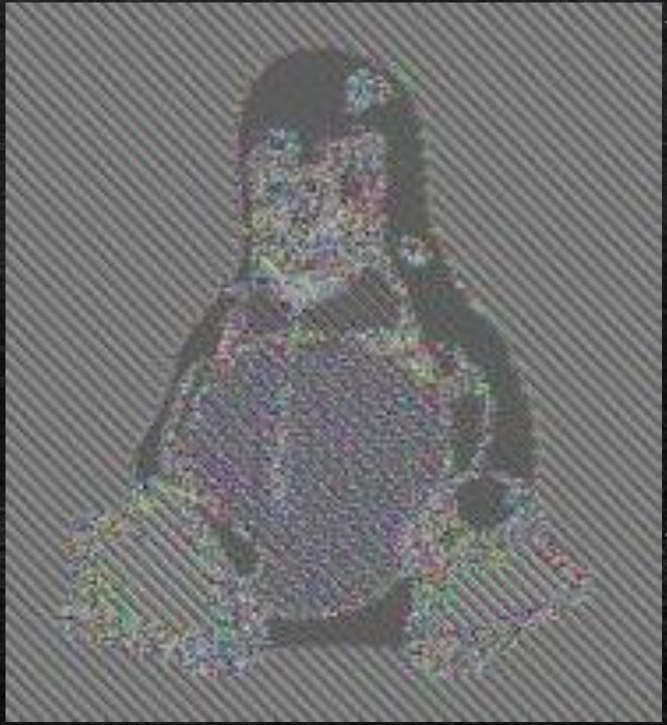


Encrypts each successive block separately.

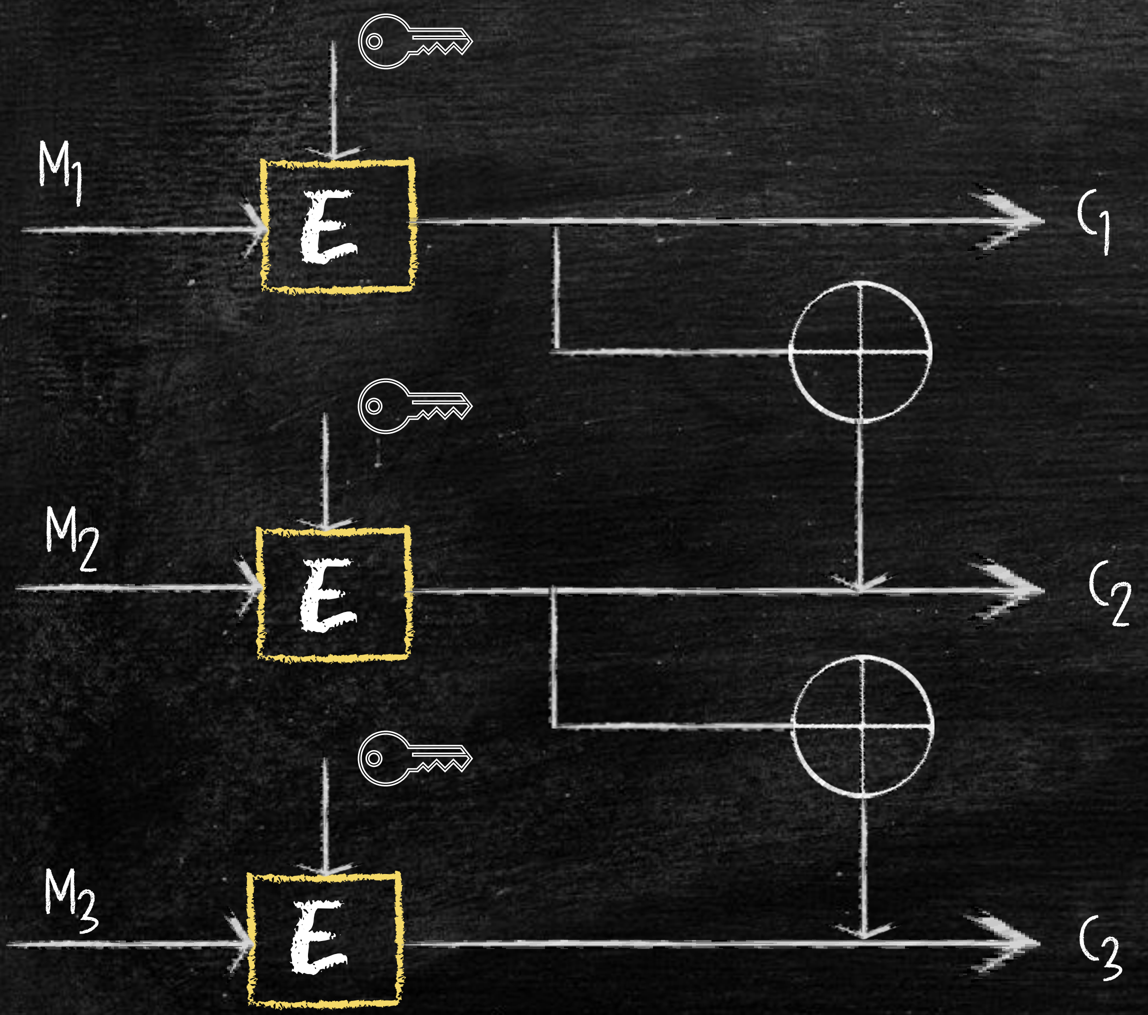
What happens if some blocks in the plain text are identical.

$C_i=E_K(M_i)$ && $C_i=E_K(M_i)$. Then, $C_i=C_j$

ECB mode

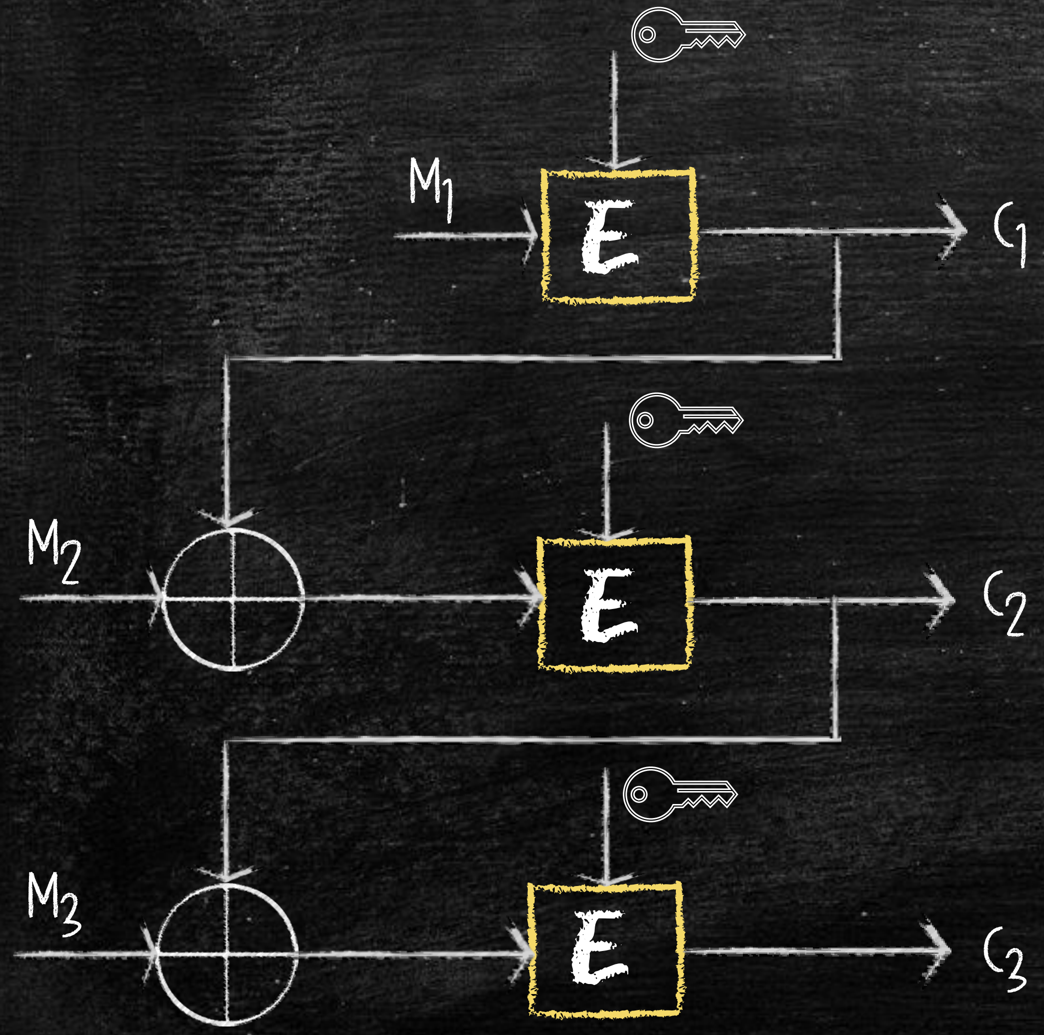


IMPROVING ECB MODE (VI)



- We can provide feedback among different block, to avoid repeating patterns
- Does this avoid repeating patterns?
- We can undo the XOR if we get all the cipher texts

IMPROVING ECB MODE (V2)



Does this solve the issue of encrypting equal block?

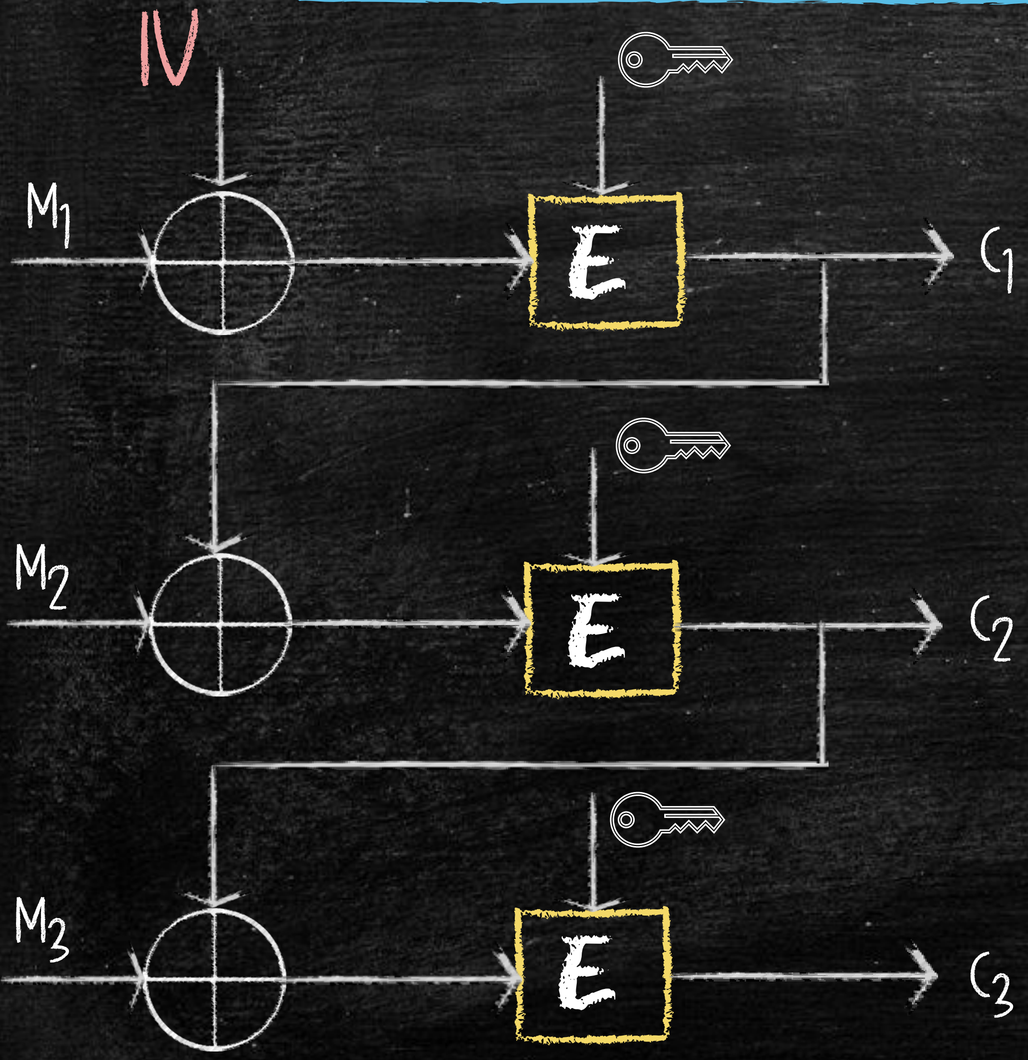
Yes!!, However ...

What would happen if we encrypt the message twice with the same key?

$C_1=E(M)$, $C_2=E(M)$ implies $C_1=C_2$

We could change the key ... but there's a better way

CIPHER BLOCK CHAINING (CBC) MODE



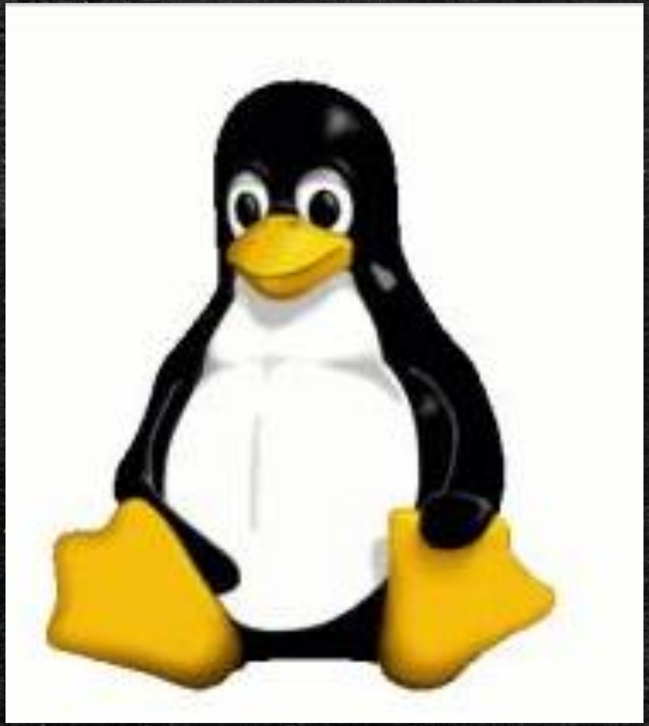
Does this solve the issue of encrypting equal block?

Yes, This is called CBC mode

Can we share IV in the clear?

Yes!!! (IV = Initialization Vector)

CBC MODE



KEY EXCHANGE

❓ How do Alice and Bob share a key?

- Maybe they meet in person
  - In General, this is very hard
- Or we invent a new technology

That's all for today, Folks!

RECAP

What is Confidentiality, Integrity, and Authenticity?

What makes a strong cryptosystem?

One-Time-Pads give perfect Secrecy but are hard to use

Stream Cipher vs. Block Cipher

Modes of Operation.

Key Exchange

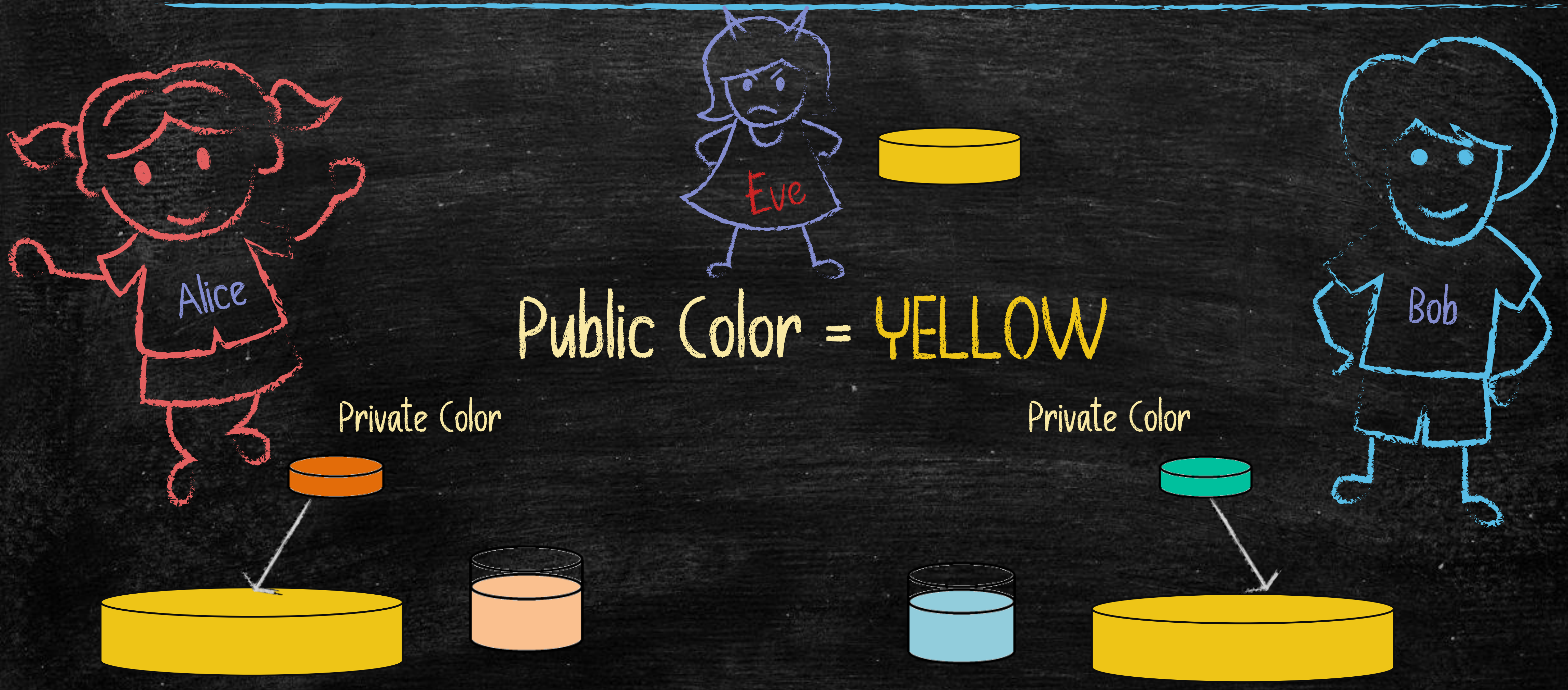
How do Alice and Bob Securely share a key?

Public Key Cryptography

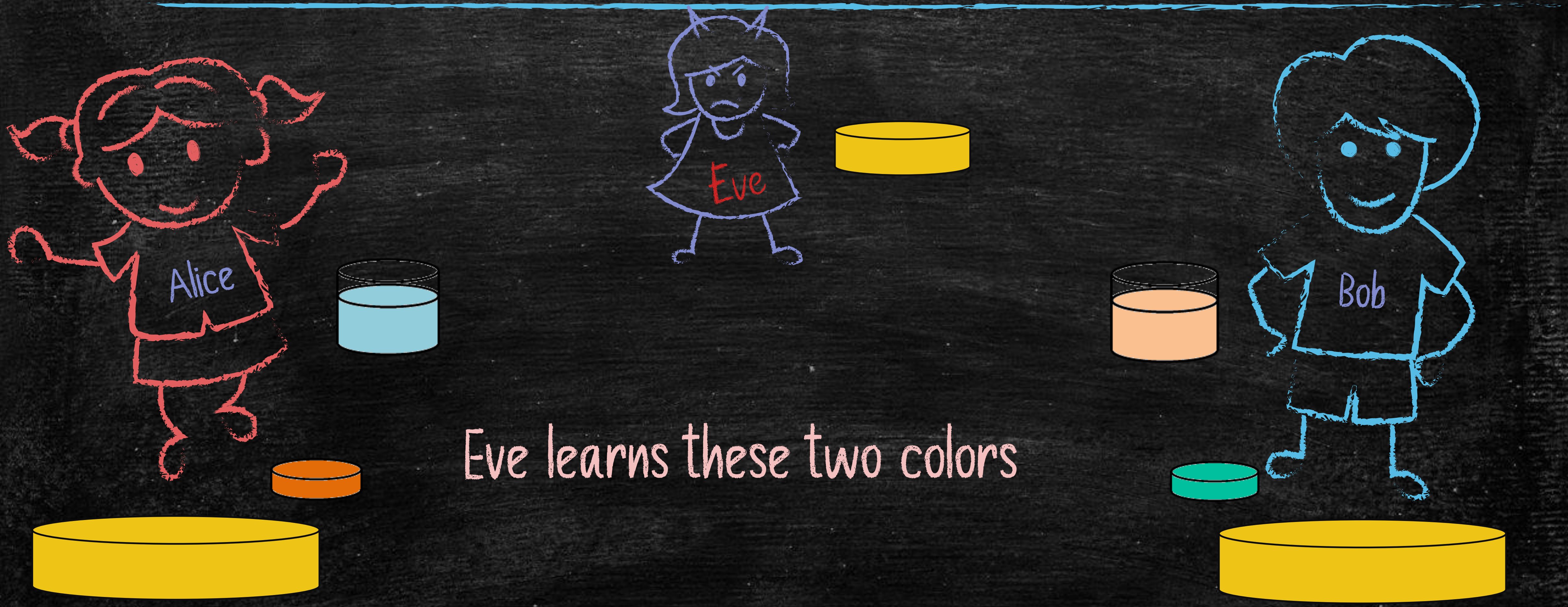
KEY-EXCHANGE

Public Color = YELLOW

KEY-EXCHANGE



Internet Key-Exchange



Eve learns these two colors

KEY-EXCHANGE

KEY-EXCHANGE

How is it that Alice & Bob's final mixtures are identical?

Alice mixed [(Yellow + Teal) from Bob] + Orange

Bob mixed [(Yellow + Orange) from Alice] + Teal

KEY-EXCHANGE

- Why doesn't Eve get know the colors?

- Unmixing a color into its component colors is a hard problem

DISCRETE LOG PROBLEM

$s = g^n \bmod p$: where p is a large prime number

Easy: given g, n, & p, solve for s

Hard: given s, g, & p, solve for n

Property: $g^{a \cdot b} \bmod p = g^{b \cdot a} \bmod p$

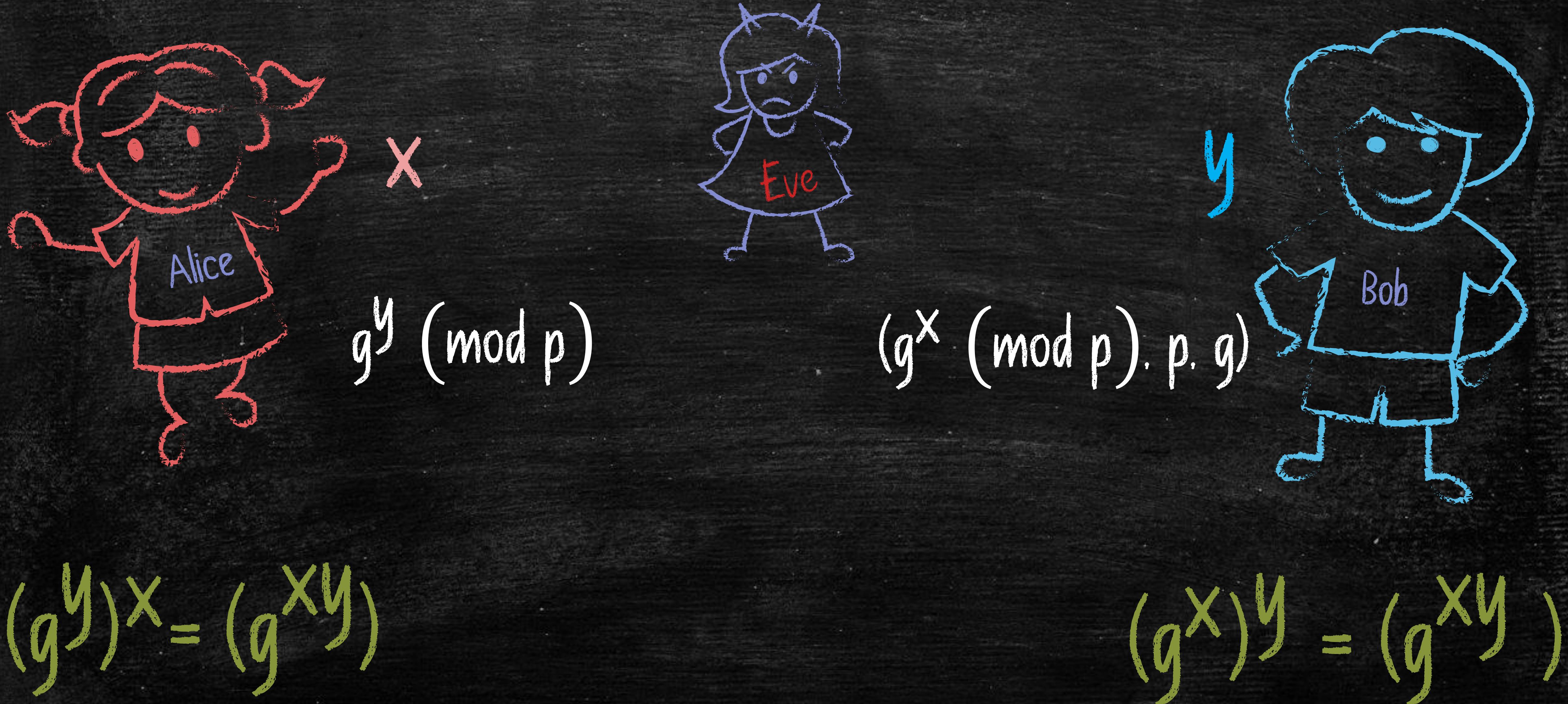
These are ONE-WAY functions

x

y

$(g^x \pmod p), p, g)$

$g^y \pmod p$


Alice
x
Eve
y
Bob
g^y (mod p)
(g^x (mod p), p, g)
(g^y)^x = (g^xy)
(g^x)^y = (g^xy)

What can go wrong?

Eve pretend to be Alice with Bob
And pretend to be Bob with Alice.

ey

ex

x

(g^{x})

$g^{ex} \pmod p$

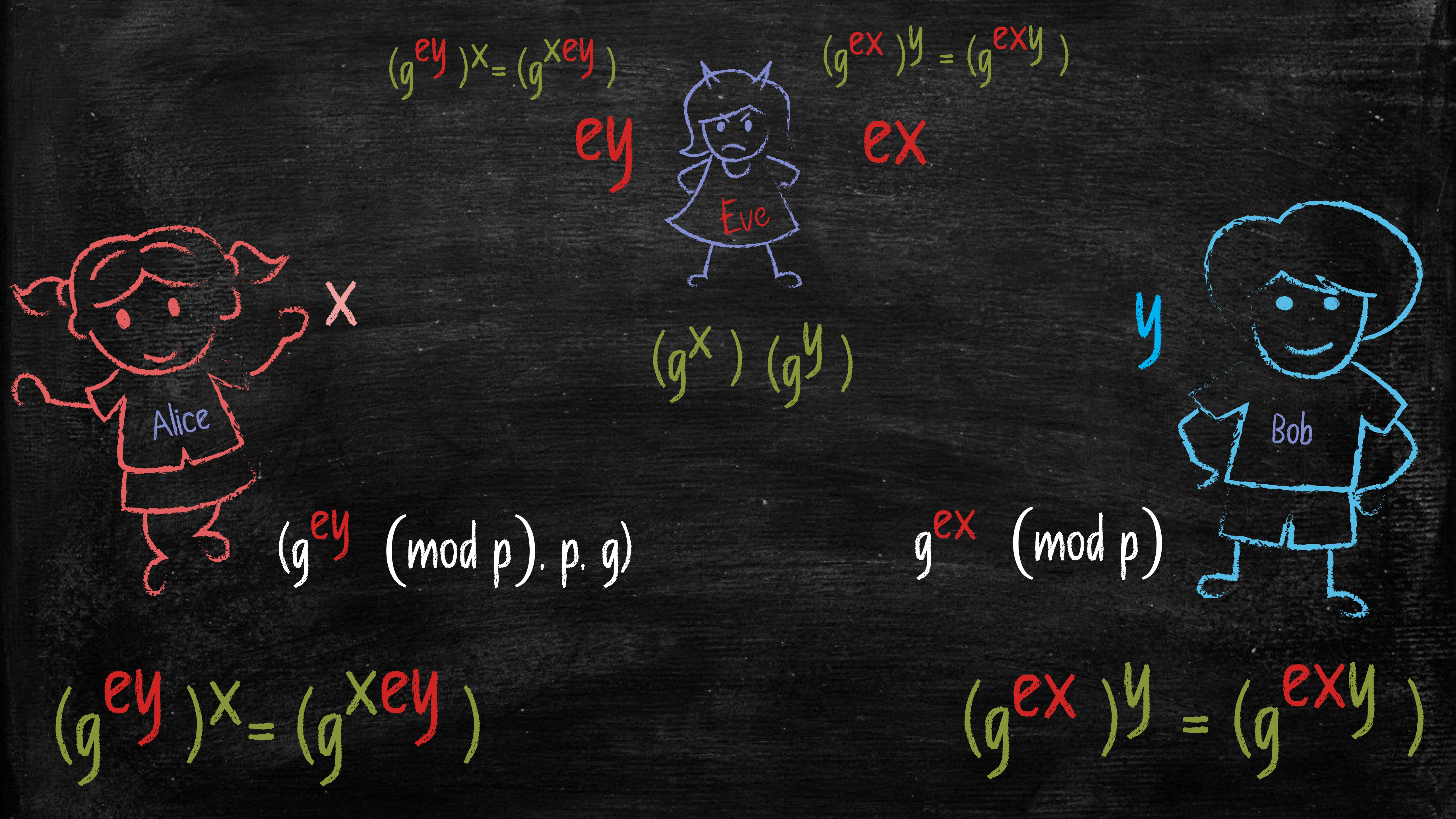
$(g^{ey} \pmod p), p, g)$

y

(g^{y})

$(g^{ey})^{x} = (g^{xey})$

$(g^{ex})^{y} = (g^{exy})$



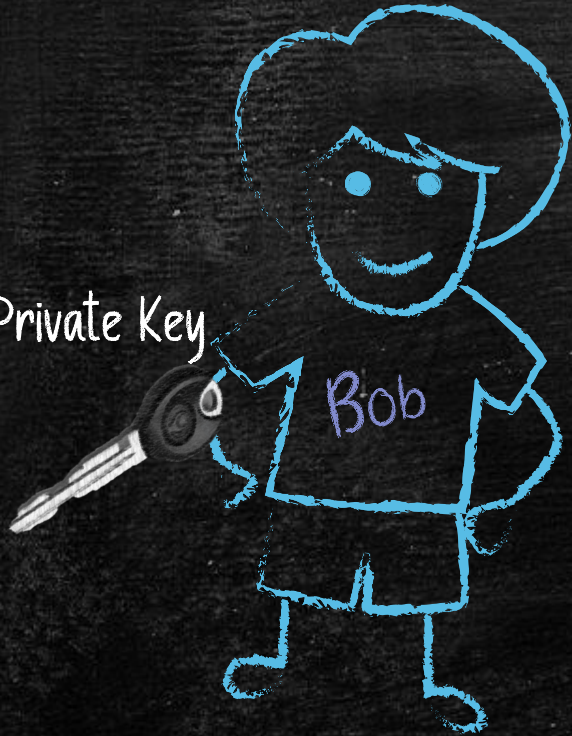
$(g^{ey})^x = (g^{xey})$
$(g^{ex})^y = (g^{exy})$
ey
ex
Eve
Alice
x
(g^x) (g^y)
y
Bob
$(g^{ey} \pmod p), p, g)$
$g^{ex} \pmod p$
$(g^{ey})^x = (g^{xey})$
$(g^{ex})^y = (g^{exy})$

What just happened here?

- Eve negotiated a key with Alice pretending to Bob
- Eve negotiated a key with Bob pretending to Alice

Let us park this problem for a while and will get back to it later

PUBLIC KEY CRYPTOGRAPHY



Private Key

Broadcasts a public key

Anybody can encrypt a message using the public key

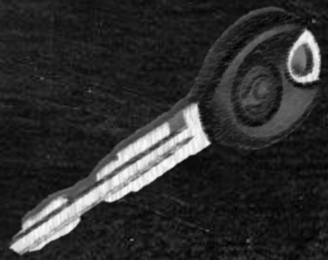
Only Bob can decrypt them using his private key

PUBLIC KEY CRYPTOGRAPHY

Also known as asymmetric cryptography

Allows Alice to send a secret message to Bob without prearranged shared secret

Encryption Key



Decryption Key

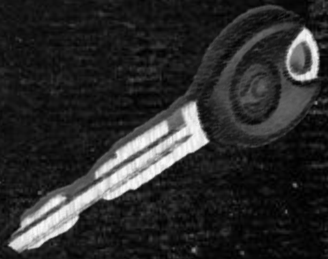
PUBLIC KEY CRYPTOGRAPHY

Invented (in public) 1970s

Also called asymmetric cryptography

Allows Alice to send a secret message to Bob without any prearranged shared secret

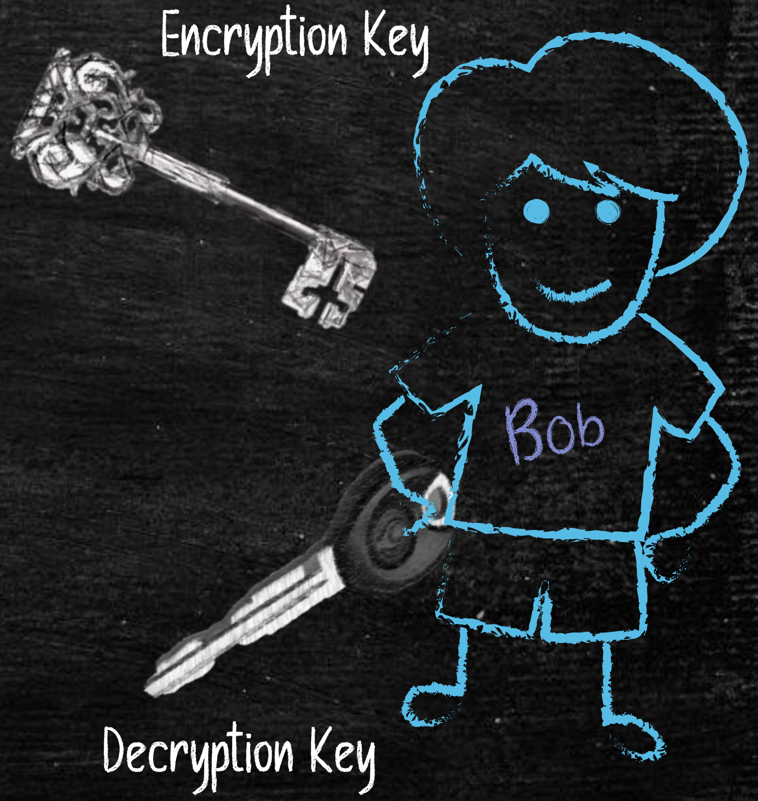
Encryption Key



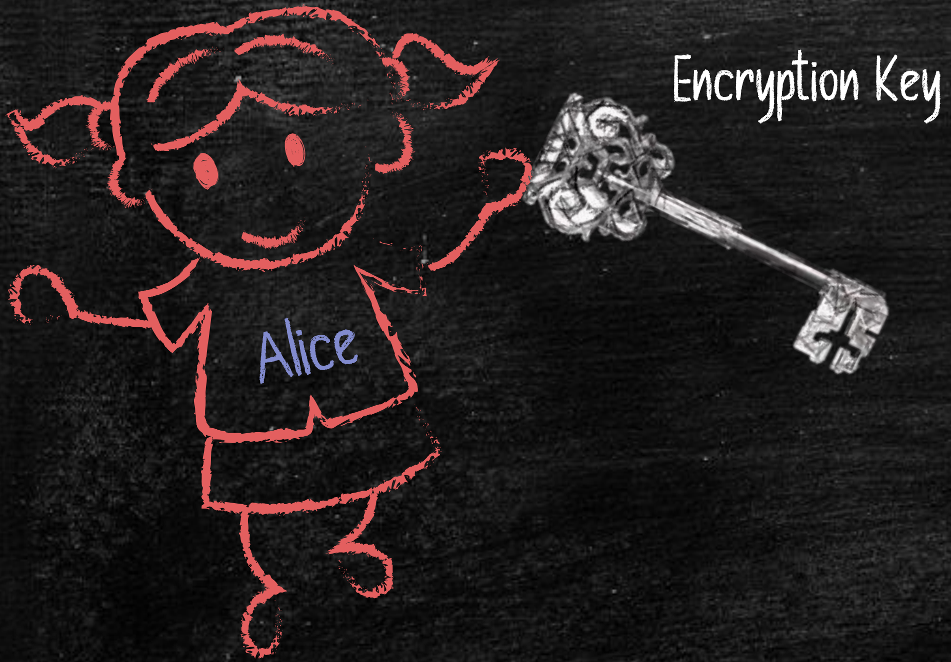
Decryption Key

Examples: RSA ElGamal ECC NTRU McEliece

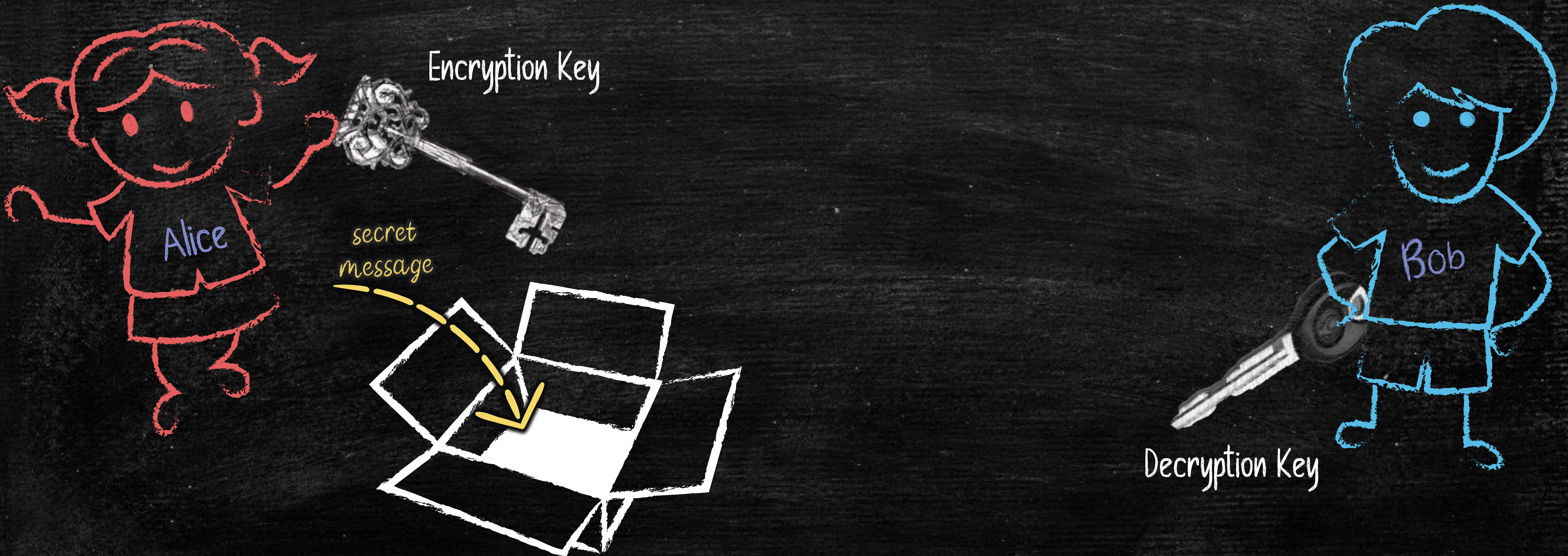
PUBLIC KEY CRYPTOGRAPHY



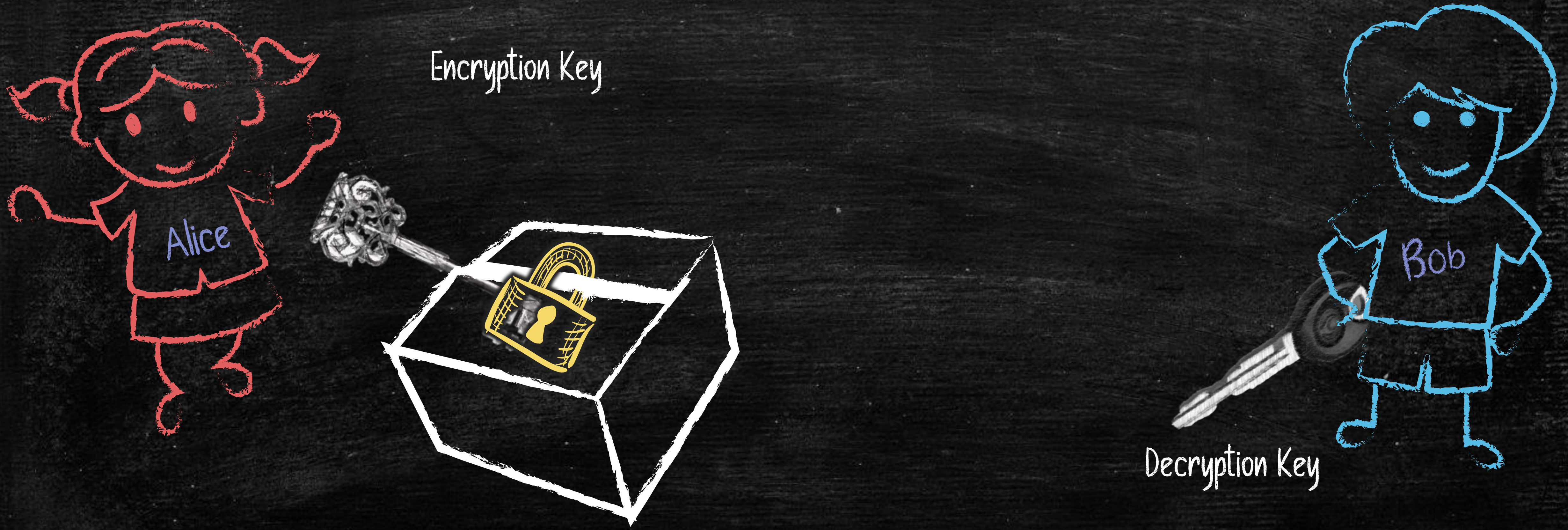
PUBLIC KEY CRYPTOGRAPHY



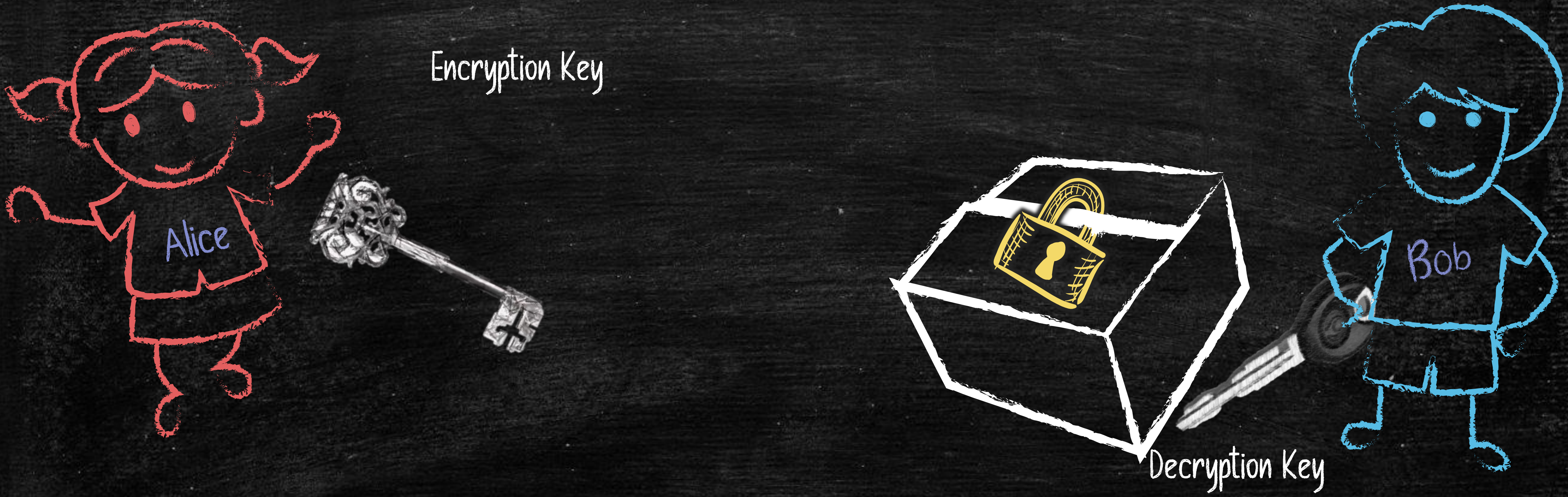
PUBLIC KEY CRYPTOGRAPHY



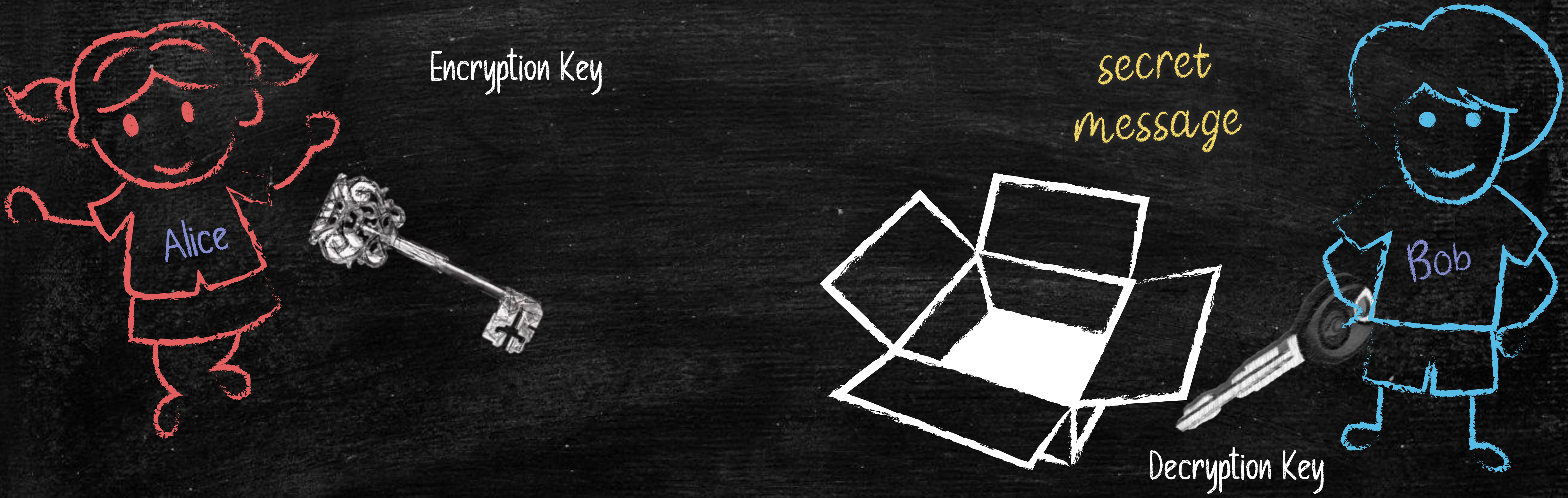
PUBLIC KEY CRYPTOGRAPHY



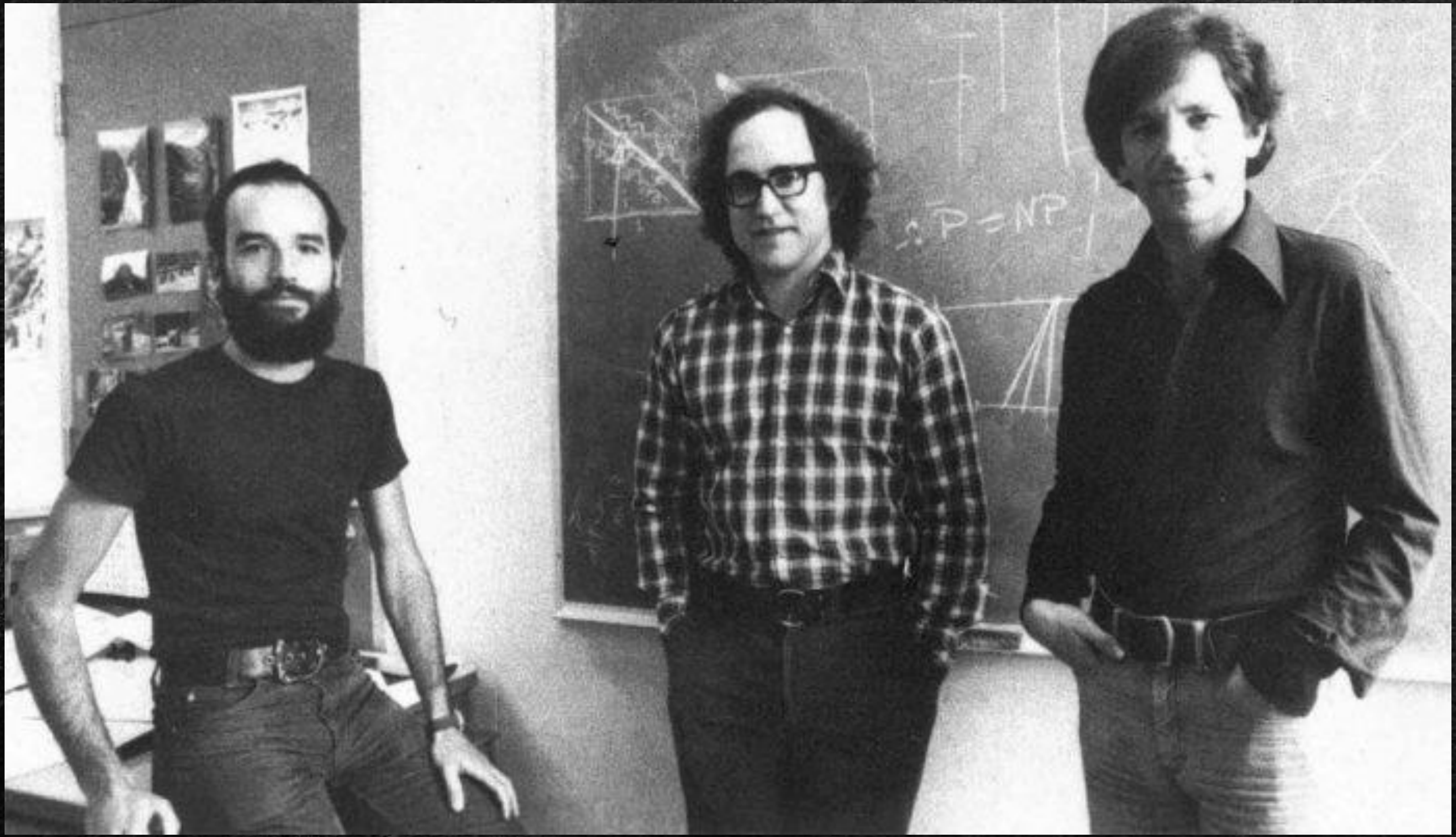
PUBLIC KEY CRYPTOGRAPHY



PUBLIC KEY CRYPTOGRAPHY



RSA CRYPTOGRAPHY



Adi Shamir, Ron Rivest, Leonard Adelman

TEXTBOOK RSA

Choose two large primes p and q.

Compute $n = p \cdot q$

Choose a number e, and find d such that $(m^e)^d \equiv m \pmod{n}$

Public Key: (e, n) Private Key: d

Encryption: $c \equiv m^e \pmod{n}$

Decryption: $m \equiv (c)^d \pmod{n}$

CHOSEN CIPHER TEXT ATTACK

We are Eve. Alice is using RSA and her public key is (e, n)

Bob sends a super secret message m which is encrypted as c=E(m). We intercept c

Choose two large primes p and q.

Compute $n = p \cdot q$

Choose a number e, and find d such that $(m^e)^d \equiv m \pmod{n}$

Public Key: (e, n)   Private Key: d

Encryption: $c \equiv m^e \pmod{n}$

Decryption: $m \equiv (c)^d \pmod{n}$

? Can we ask Alice to decrypt something else (other than c) that helps us generate m?

CHOSEN CIPHER TEXT ATTACK

? Can we ask Alice to decrypt something else (other than c) that helps us generate m?

Choose two large primes p and q.

Compute $n = p \cdot q$

Choose a number e, and find d such that $(m^e)^d \equiv m \pmod{n}$

Public Key: (e, n) Private Key: d

Encryption: $c \equiv m^e \pmod{n}$

Decryption: $m \equiv (c)^d \pmod{n}$

Bob sends $c_1 = E_e(m)$. We intercept c_1.

We ask Alice to decrypt $c_2 \equiv 2^e \cdot c_1$

The decrypt. yields: $(2^e \cdot c_1)^d = 2m$

We divide the result by 2, and we get m.

We fix this by something called padding techniques

PUBLIC KEY SIZES

AES	RSA	ECC
80	1024	160
116	2048	232
128	2600	256
160	4500	320
256	14000	512

HYBRID CRYPTOGRAPHY

What is the advantage/disadvantage of Secret-Key Cryptography?

Shorter Keys, Faster, Same Key to Encrypt-Decrypt

What is the advantage/disadvantage of Public Cryptography?

Longer Keys, Slower, Different Key to Encrypt-Decrypt

HYBRID CRYPTOGRAPHY

Pick a random 128-bit key K for a secret-key cryptosystem
Encrypt the large message with the key K (e.g., using AES)
Encrypt the key K using a public-key cryptosystem
Send the encrypted message and the encrypted key to Bob

The hybrid approach is used for almost every cryptographic application on the internet today.

QUIZ TIME

eA and dA are the public parameters

K is the secret key parameter

eB and dB are the public parameters

How does Alice send a LARGE message to Bob

Alice uses Public Key encryption to send K and encrypts M with K

QUIZ TIME

? So, we can secret messages.
But, what else can Eve do?

Eve can modify our encrypted messages in transit.

? How do we make sure that Bob gets the same message Alice sent?

Integrity

How do we tell if a message changed in transit?

Simplest answer use a checksum!

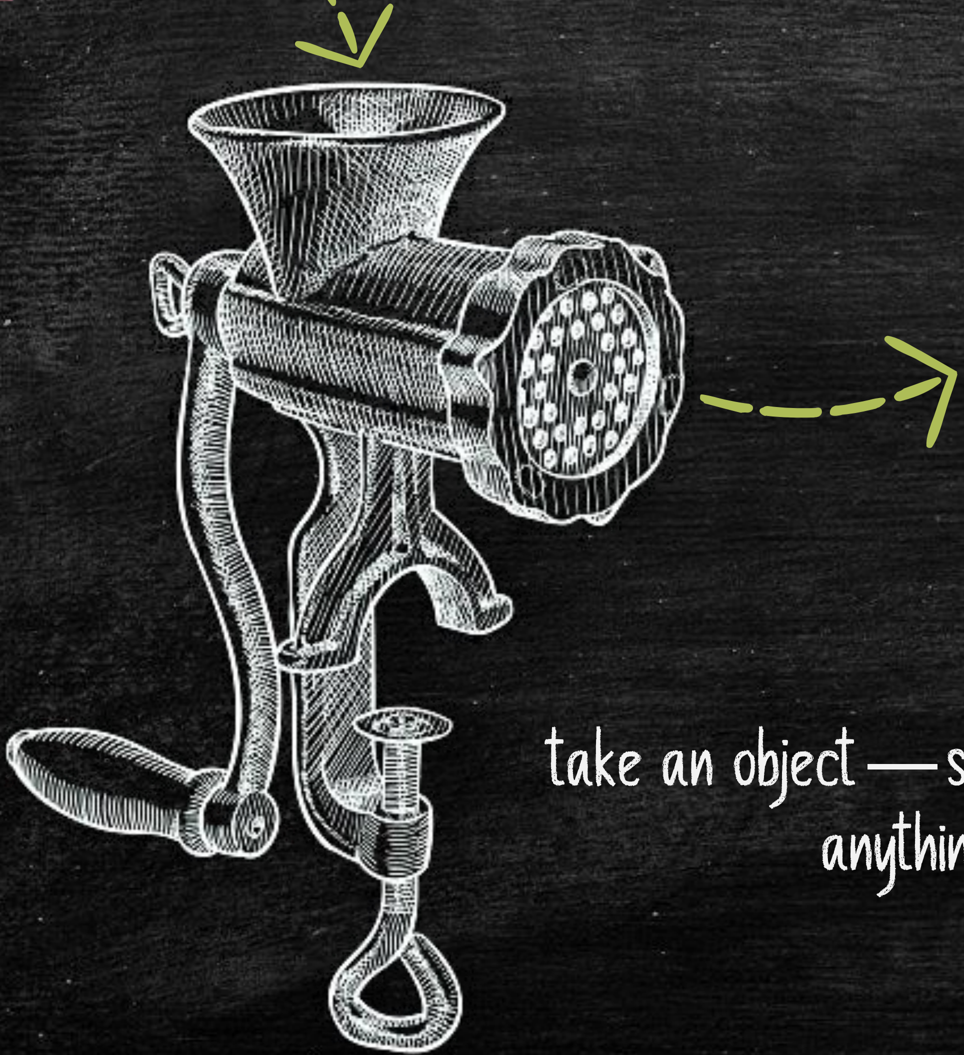
For example, add up all the bytes of a message.

INTEGRITY COMPONENTS

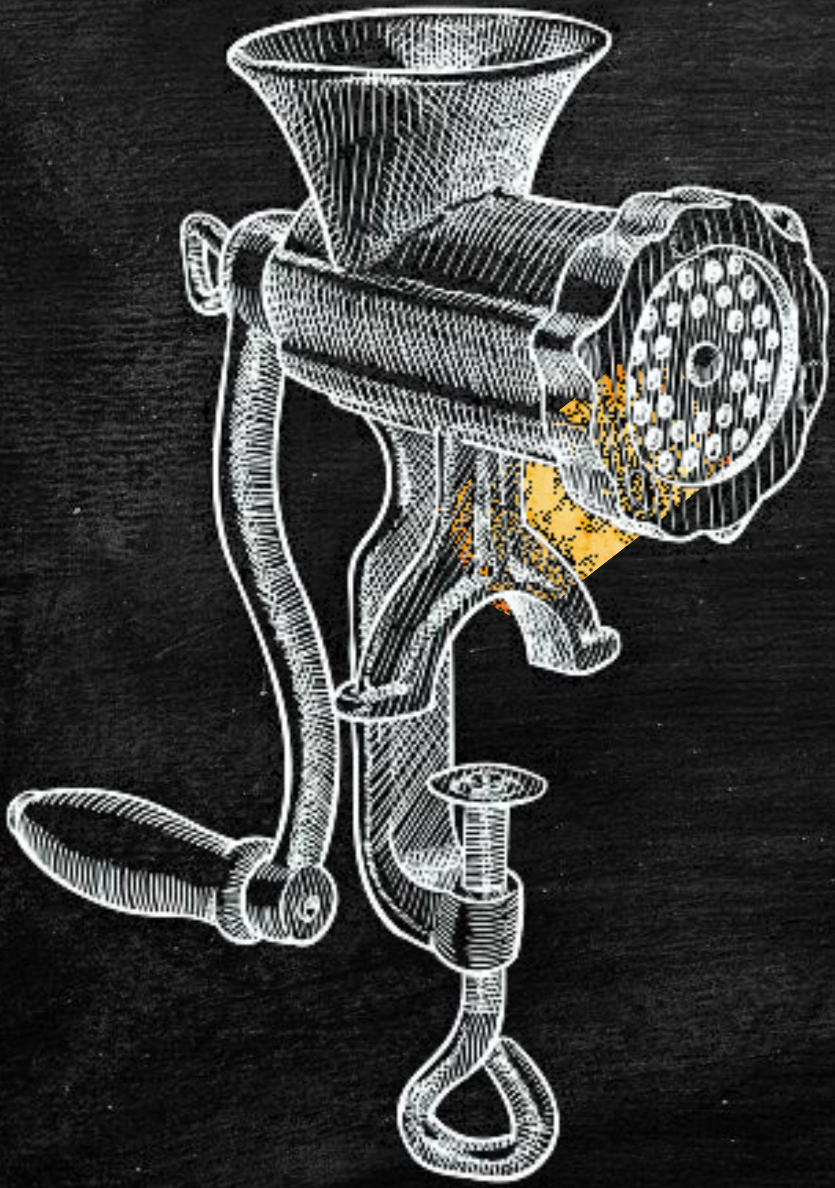
- Does a checksum work?
- What can Eve do?
- Eve can easily change the message in such a way that checksum stays the same
- We need a cryptographic checksum

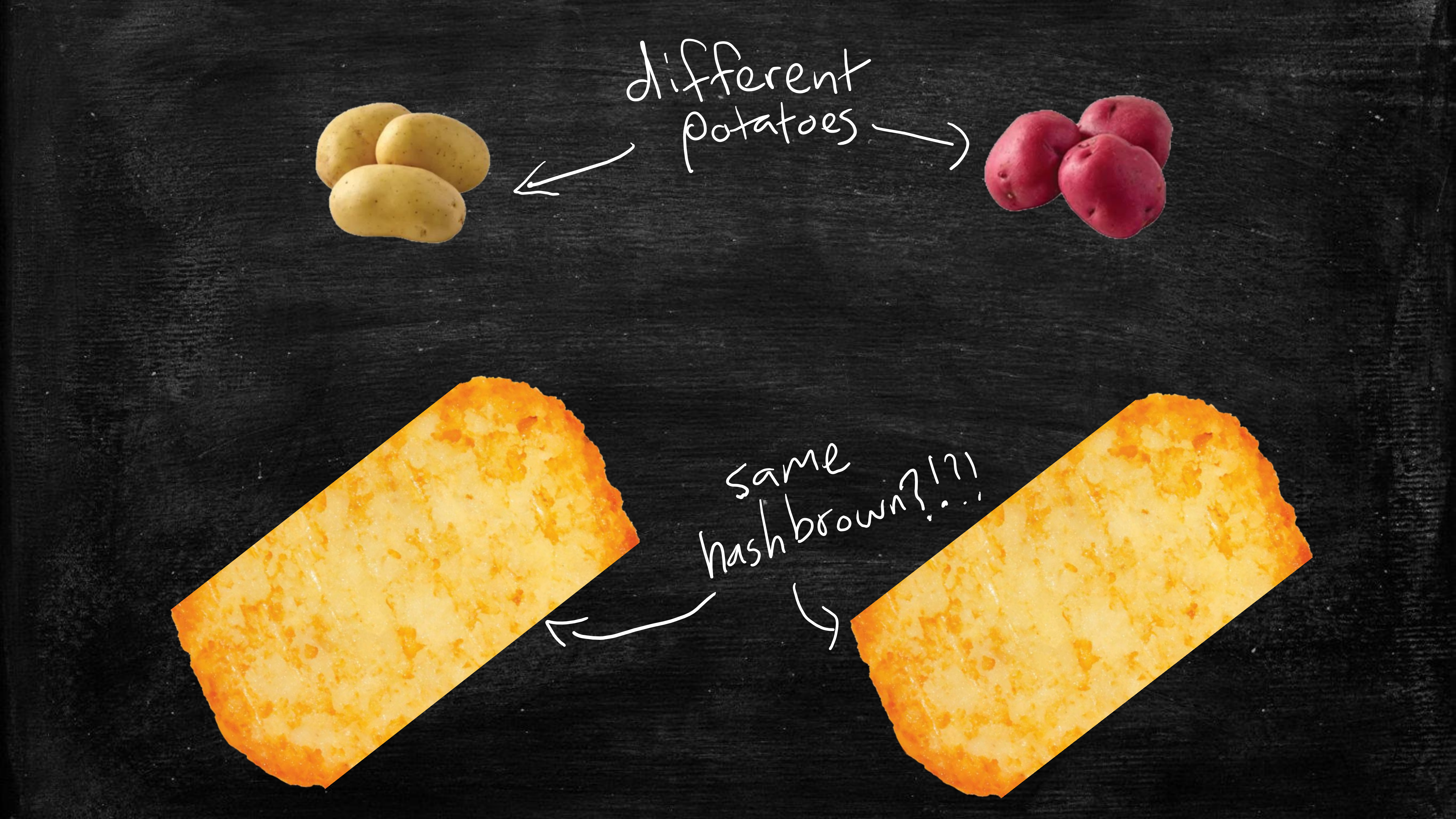
“cryptographic checksum”

- It should be hard for Eve to find a second message with same checksum as any given one



take an object — say, a potato — and then "hash" it up until it looks just like anything else and lacks any of its original structure




different potatoes
same hashbrown?!?!

CRYPTOGRAPHIC HASH FUNCTION

- A function that turns any message into a "short", "unique", and "irreversible" string of bits

collision resistant

preimage resistant

say, 256 bits

- Output of a hash function is called a "hash", a "digest" or a "fingerprint" of the input

CRYPTOGRAPHIC HASH FUNCTION

- A function that turns any message into a "short", "unique", and "irreversible" string of bits

collision resistant

preimage resistant

say, 256 bits

CRYPTOGRAPHIC HASH FUNCTION

“ function ”

predictable mapping of inputs ↦ outputs

- Mapping is deterministic: H(x)=H(x), always

CRYPTOGRAPHIC HASH FUNCTION

- A function that turns any message into a "short", "unique", and "irreversible" string of bits

collision resistant

preimage resistant

say, 256 bits

CRYPTOGRAPHIC HASH FUNCTION

"any message"

input can be *any* bit string of any length

- whether 1 byte or 100 petabyte or more

(Formally, the domain of H is $\{0, 1\}^*$, the set of all finite bit string)

CRYPTOGRAPHIC HASH FUNCTION

- A function that turns any message into a "short", "unique", and "irreversible" string of bits

collision resistant

preimage resistant

say, 256 bits

CRYPTOGRAPHIC HASH FUNCTION

“ short ”

output is a string of some fixed length

- most commonly that's 256 bits (32 bytes), though 128, 192, 512... aren't unheard of

(Formally, the range of H is $\{0, 1\}^\lambda$, the set of all λ-bit strings)

CRYPTOGRAPHIC HASH FUNCTION

A function that turns any message into a "short", "unique", and "irreversible" string of bits

collision resistant

preimage resistant

say, 256 bits

CRYPTOGRAPHIC HASH FUNCTION

"unique"

two inputs "almost always" map to two outputs

- there are 2^{256} possible outputs

(2^{256} is a really, really, *REALLY* big number...)

estimated 2^{260} atoms in the *observable universe*

Note: This this is clearly impossible!?

CRYPTOGRAPHIC HASH FUNCTION

- A function that turns any message into a “short”, “unique”, and “irreversible” string of bits

collision resistant

preimage resistant

say, 256 bits

CRYPTOGRAPHIC HASH FUNCTION

> “irreversible”

↳ no good way to recover the inputs from outputs

- best available method is to guess and check

IS THAT ALL?

? Suppose we don't care about confidentiality!
What can Eve do to change the message?

IS THAT ALL?

Now, What can Eve do to change the message?

CRYPTOGRAPHIC HASH FUNCTIONS

- Hash Functions provide integrity only when there is a secure way of sending the message digest

Authentication

MESSAGE AUTHENTICATION CODES (MAC)

We can use key has functions, that are usually called Message Authentication Code

Only those who know the secret key can generate, on even check, the computed hash value (sometimes called a tag).

MESSAGE AUTHENTICATION CODES (MAC)

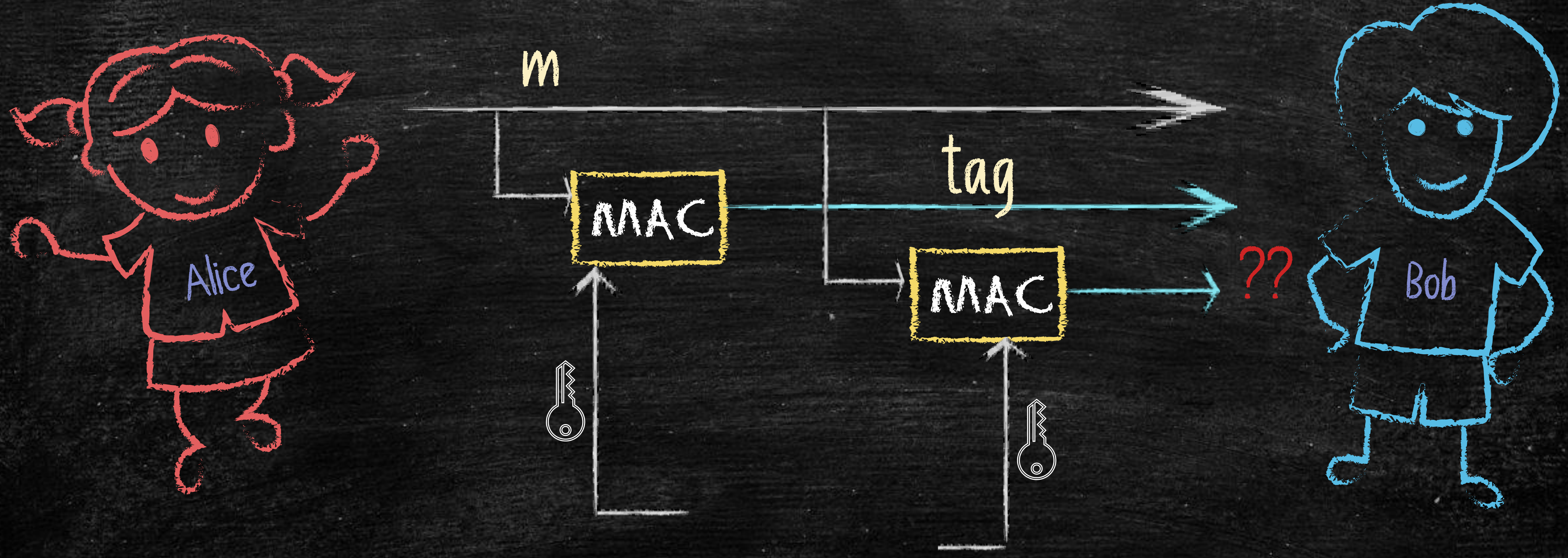
M	H(M)

❓ What is wrong with this?

❓ How can we fix it?

M	H(M| )

MESSAGE AUTHENTICATION CODES (MAC)



COMBINING CIPHERS AND MACS

- What's the issue with this?
- In practice we often need both confidentiality and message integrity!
- What are our options?
- MAC-and-then-Encrypt,
  Encrypt-and-MAC,
  Encrypt-then-MAC

COMBINING CIPHERS AND MACS

MAC-then-Encrypt	$E(m \parallel MAC(m))$
MAC-and-Encrypt	$E(m) \parallel MAC(m)$
Encrypt-then-MAC	$E(m) \parallel MAC(E(m))$

What is recommended strategy?

Encrypt-then-MAC see this blog

REPUDIATION

Bob can be assured that Alice is the one who sent m and that the message has not been modified since she sent it!

We have confidentiality, integrity, and authentication

This is like a "signature" on the message… but not quite the same!

Bob can't prove to Eve that Alice sent m, though.

REPUDIATION

Bob can't prove to Eve that Alice sent m, though.

WHY?

Either Alice or Bob could create any of the message and MAC combinations.
Also, Eve doesn't know the secret keys.

REPUDIATION

Alice can just claim that Bob made up the message m, and calculated the MAC himself

This is called repudiation!!!

REPUDIATION

- Some interactions should be repudiable
- Private conversations

REPUDIATION

- Some interactions should be non-repudiable
- Electronic Commerce

Digital Certificates

ALICE HAS $1 AND BOB NEEDS $1

Alice has $1 and Bob needs $1

So Alice graciously loans the $1 to Bob

Alice has $1 and Bob needs $1

So Alice graciously loans the $1 to Bob

In exchange, Bob writes an IOU for Alice

...and signs it

Signatures are the digital analog of the preceding scenario

A digital signature scheme has three algorithms:

1. Gen creates a pair of keys;
2. Sign produces a signature under a given key;
3. Ver checks a signature using associated the key.

An digital signature scheme has three algorithms:

1. Gen creates a pair of keys:
   - sk ("signing key") creates signatures over messages
   - vk ("verification key") checks if signatures are valid

2. Sign produces a signature under a given key:
3. Ver checks a signature using associated the key.

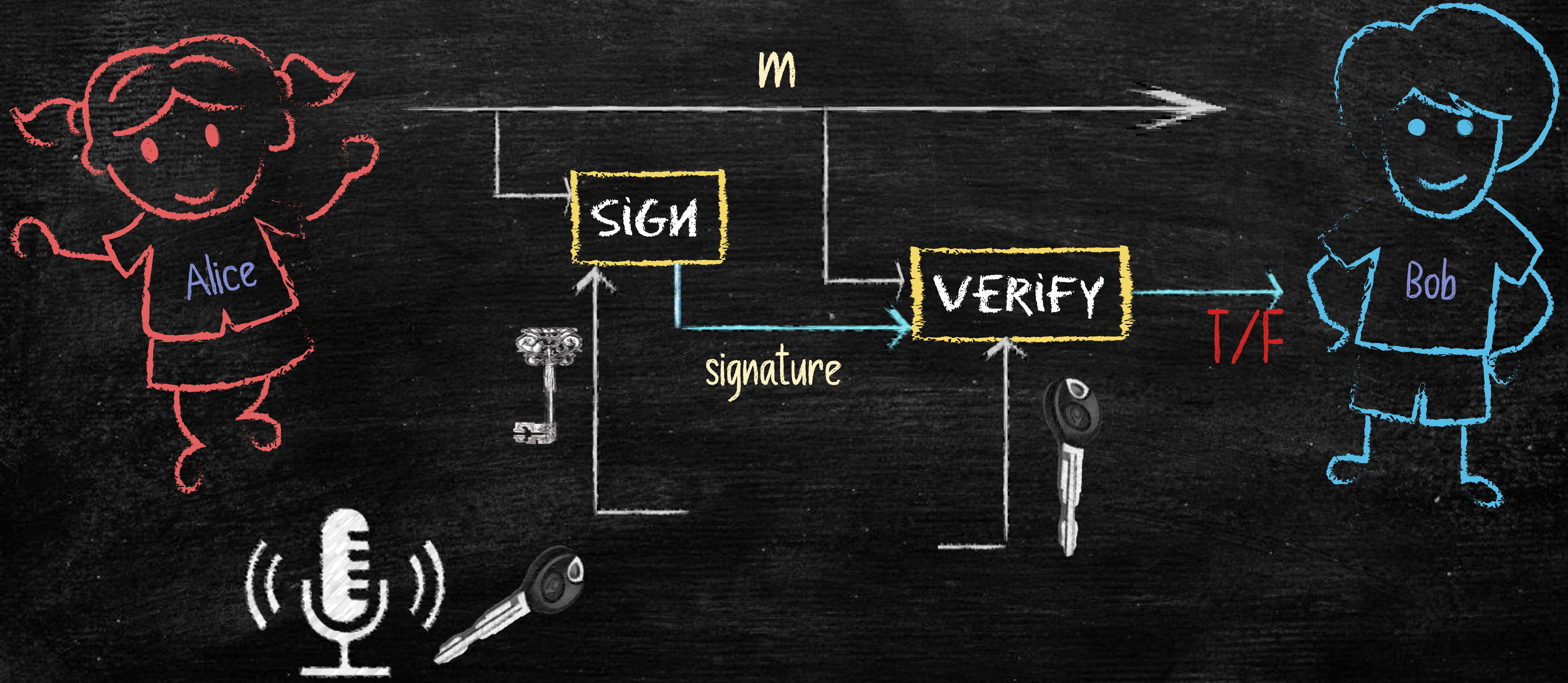
Kind of like the inverse of public-key encryption

Encryption: Anybody can close padlock; only keyholder can open it

vs.

Signatures: Anybody can open padlock; only keyholder can close it

DIGITAL SIGNATURES



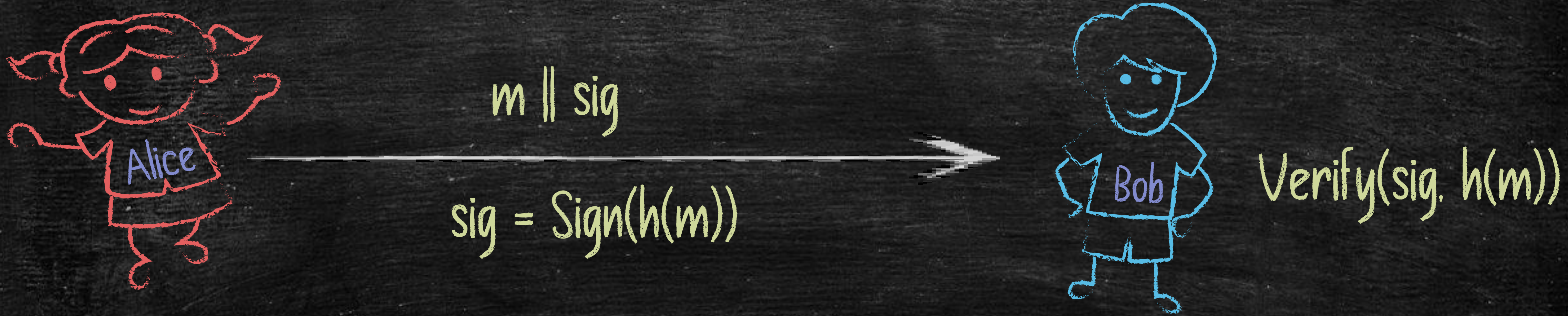
FASTER SIGNATURES

Just like encryption in public-key crypto signing large messages is slow

We can also "hybridize" signatures to make them faster:

Alice sends the (unsigned) message, and also a signature on a hash of the message
The hash is much smaller than the message, and so it is faster to sign and verify

FASTER SIGNATURES



Remember that authenticity and confidentiality are separate: if you want both, you need to do both

Combining Public-Key Encryption and Digital Signatures

Alice has two different key pairs
Alice uses to encryption key to encrypt the message
Alice uses the signing key to sign the cipher text.

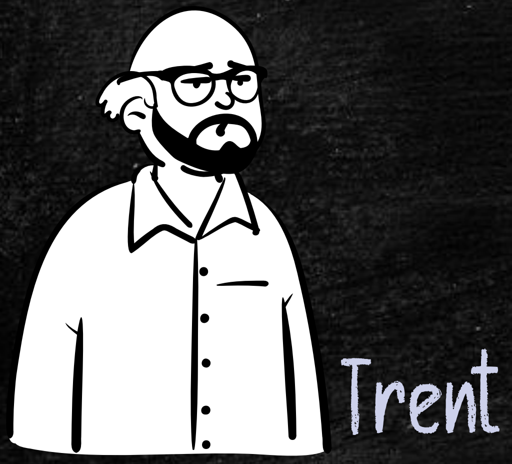
Bob also has two different key pairs
Alice uses to verification key to verify the cipher text
Alice uses the decryption key to decrypt the message.

THE KEY MANAGEMENT PROBLEM

- How can Alice and Bob be sure that are talking to each other?
- By having each other's verification key
- But, How do we verify each others' verification key?

CERTIFICATION AUTHORITY (CA)

A CA is a trusted third party who keeps a directory of people's (and organizations') verification keys

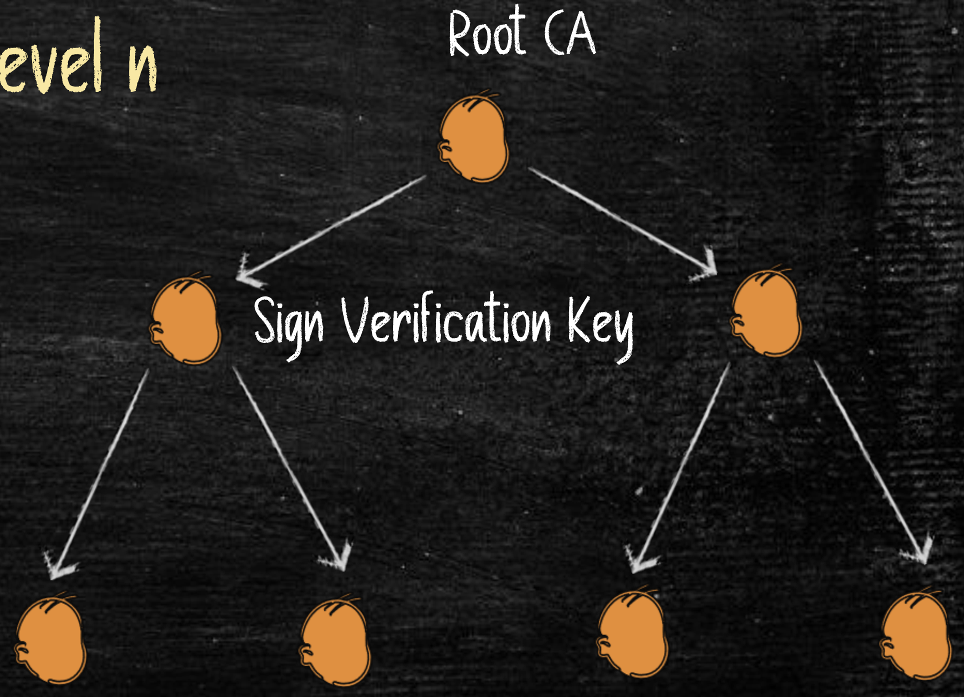


CERTIFICATION AUTHORITY (CA)

Everyone is assumed to have a copy of the CA's verification key (vCAk), so they can verify the signature on the certificate

There can be multiple levels of certificate authorities: level n CA issues certificates for level n+1 CAs – Public-key infrastructure (PKI)

Need to have only verification key of root CA to verify the certificate chain



CERTIFICATION AUTHORITY (CA)

All Root Certificate Authorities are Equal.

If I've a root certificate authority I've never used before, I won't treat it any differently from the same certificate I've always used.

See this amazing talk by Joel Reardon

actors in the French economy, lending

recent stock market history. *PAGE 16*

Iranian activists feel the chill as hacker taps into e-mails

BY SOMINI SENGUPTA

He claims to be 21 years old, a student of software engineering in Tehran who reveres Ayatollah Ali Khamenei and despises dissidents in his country.

He sneaked into the computer systems of a security firm on the outskirts of Amsterdam. He created fake credentials that could allow someone to spy on Internet connections that appeared to be secure. He then shared that bounty with people he declines to identify.

The fruits of his labor are believed to have been used to tap into the online communications of as many as 300,000 unsuspecting Iranians this summer. What is more, he punched a hole in an online security mechanism that is trusted by Internet users all over the world.

Comodohacker, as he calls himself, insists that he acted on his own and is unperturbed by the notion that his work might have been used to spy on anti-government compatriots.

"I'm totally independent," he said in an e-mail exchange with The New York Times. "I just share my findings with some people in Iran. They are free to do anything they want with my findings and things I share with them, but I'm not responsible."

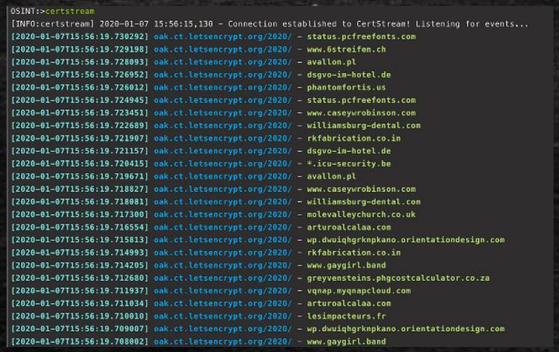
In the annals of Internet attacks, this is most likely to go down as a moment of reckoning. For activists, it shows the

HACKER, PAGE 17

In principle there is nothing wrong if an obscure Dutch CA starts singing many Iranian Websites

CERTIFICATION TRANSPARENCY

Every New Certificate Gets added to a list



```
OSINT:>certstream
[INFO:certstream] 2020-01-07 15:56:15,130 - Connection established to CertStream! Listening for events...
[2020-01-07T15:56:19.730292] oak.ct.letsencrypt.org/2020/ - status.pcfreefonts.com
[2020-01-07T15:56:19.729198] oak.ct.letsencrypt.org/2020/ - www.6streifen.ch
[2020-01-07T15:56:19.728093] oak.ct.letsencrypt.org/2020/ - avallon.pl
[2020-01-07T15:56:19.726952] oak.ct.letsencrypt.org/2020/ - dsgvo-im-hotel.de
[2020-01-07T15:56:19.726012] oak.ct.letsencrypt.org/2020/ - phantomfortis.us
[2020-01-07T15:56:19.724945] oak.ct.letsencrypt.org/2020/ - status.pcfreefonts.com
[2020-01-07T15:56:19.723451] oak.ct.letsencrypt.org/2020/ - www.caseywrobinson.com
[2020-01-07T15:56:19.722689] oak.ct.letsencrypt.org/2020/ - williamsburg-dental.com
[2020-01-07T15:56:19.721907] oak.ct.letsencrypt.org/2020/ - rkfabrication.co.in
[2020-01-07T15:56:19.721157] oak.ct.letsencrypt.org/2020/ - dsgvo-im-hotel.de
[2020-01-07T15:56:19.720415] oak.ct.letsencrypt.org/2020/ - *.icu-security.be
[2020-01-07T15:56:19.719671] oak.ct.letsencrypt.org/2020/ - avallon.pl
[2020-01-07T15:56:19.718827] oak.ct.letsencrypt.org/2020/ - www.caseywrobinson.com
[2020-01-07T15:56:19.718081] oak.ct.letsencrypt.org/2020/ - williamsburg-dental.com
[2020-01-07T15:56:19.717300] oak.ct.letsencrypt.org/2020/ - molevalleychurch.co.uk
[2020-01-07T15:56:19.716554] oak.ct.letsencrypt.org/2020/ - arturoalcalaa.com
[2020-01-07T15:56:19.715813] oak.ct.letsencrypt.org/2020/ - wp.dwuiqhgrknpkano.orientationdesign.com
[2020-01-07T15:56:19.714993] oak.ct.letsencrypt.org/2020/ - rkfabrication.co.in
[2020-01-07T15:56:19.714205] oak.ct.letsencrypt.org/2020/ - www.gaygirl.band
[2020-01-07T15:56:19.712680] oak.ct.letsencrypt.org/2020/ - greyvensteins.phgcostcalculator.co.za
[2020-01-07T15:56:19.711937] oak.ct.letsencrypt.org/2020/ - vqnap.myqnapcloud.com
[2020-01-07T15:56:19.711034] oak.ct.letsencrypt.org/2020/ - arturoalcalaa.com
[2020-01-07T15:56:19.710010] oak.ct.letsencrypt.org/2020/ - lesimpacteurs.fr
[2020-01-07T15:56:19.709007] oak.ct.letsencrypt.org/2020/ - wp.dwuiqhgrknpkano.orientationdesign.com
[2020-01-07T15:56:19.708002] oak.ct.letsencrypt.org/2020/ - www.gaygirl.band
```

Does it prevent Bad Certificates?

Doesn't prevent generation of bad certs.

Provides Accountability

RECAP OF CRYPTO TOOLS

Secret Key Cryptography

- One-Time Pads
- Stream Ciphers
- Block Ciphers

Public Key Cryptography

- Textbook RSA
- Secret vs. Public Key Cryptography
- Hybrid Cryptography

Integrity

- Checksums
- Hash Functions

Authentication

- MACs
- Digital signatures
- Key Management

RECAP QUIZ

- Does it prevent Bad Certificates?
- Doesn't prevent generation of bad certs.

Overview of Security Controls

NETWORKING 101

Alice is sitting in her office at U Waterloo.

She connects her phone to the WiFi.

Goes to Amazon.com

Buys a new laptop.

? How do packets travel in the network?

NETWORKING 101

Link Layer: At the link layer, Alice's mobile device establishes a wireless connection with the Wi-Fi access point. The link layer protocols, such as Wi-Fi (e.g., 802.11), handle the transmission of data between her device and the access point.

NETWORKING 101

Network Layer: Once the Wi-Fi connection is established, Alice's mobile device obtains an IP address through DHCP (Dynamic Host Configuration Protocol). The network layer protocols, such as IP (Internet Protocol), come into play. Alice's mobile device sends an IP packet containing the source and destination IP addresses.

NETWORKING 101

Transport Layer: Alice's mobile device chooses a transport layer protocol, typically TCP (Transmission Control Protocol) for web browsing. A TCP connection is established between her device and Amazon's server. The transport layer segments the data into TCP segments and adds the source and destination port numbers.

NETWORKING 101

Application Layer: At the application layer, Alice's mobile device initiates a request to Amazon's server using an HTTP (Hypertext Transfer Protocol) request. The request includes the specific Amazon URL, such as "https://www.amazon.com," and any additional parameters or data required for the purchase.

JOURNEY OF A PACKET

Application Layer

Transport Layer

Network Layer

Link Layer

Where do we need to apply cryptography?

All the Layers

WEP

“Link Layer”

→ WEP was used for Wireless Networks

WEP

WEP was intended to enforce three security goals:

Data Confidentiality

- Prevent an adversary from learning the contents of the wireless traffic

Data Integrity

- Prevent an adversary from modifying the wireless traffic or fabricating traffic

Access Control

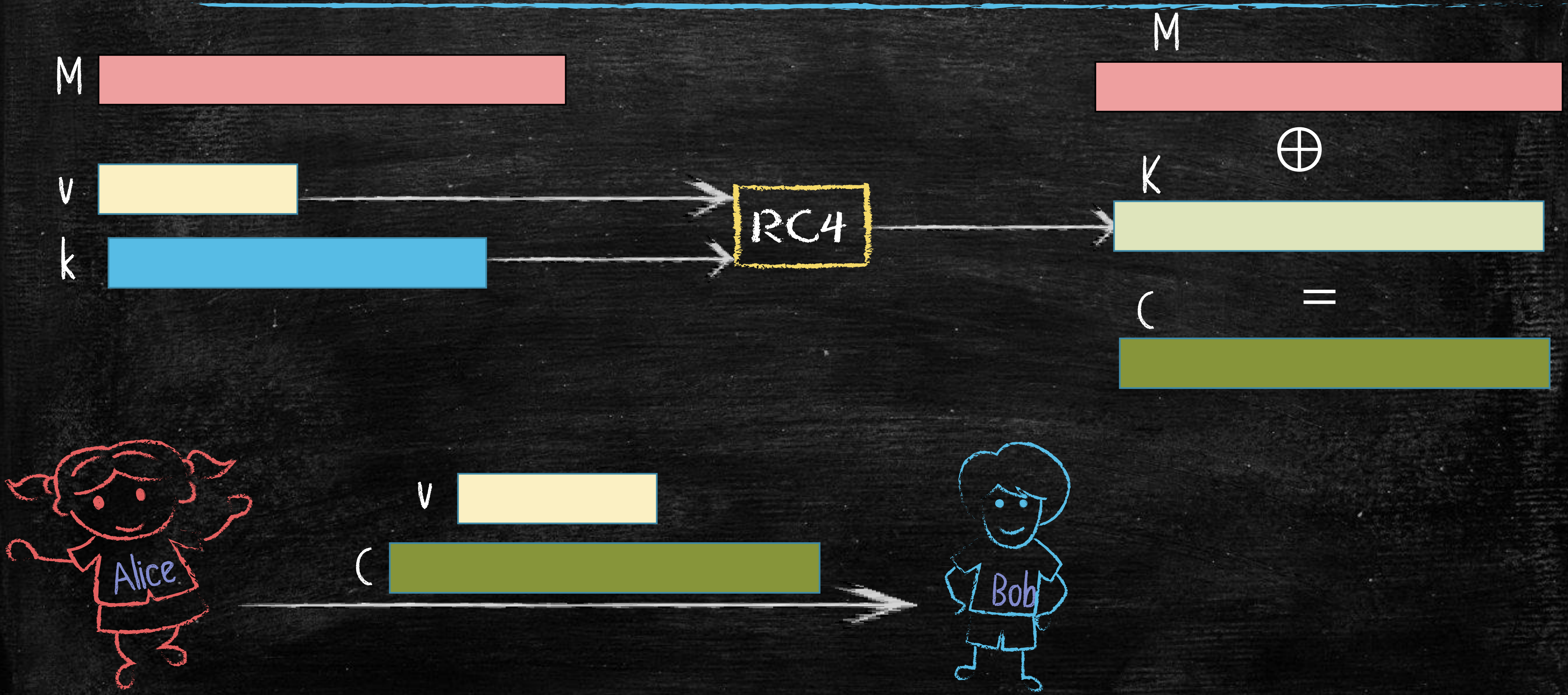
- Prevent an adversary from using your wireless infrastructure

WEP

WEP was intended to enforce three security goals:

Unfortunately, none of these is *actually* enforced!!!

WEP



WEP

In order to transmit a message M:

- Compute a checksum c(M) (which does nor depend on k)
- Pick an IV v and generate a keystream K=RC4(v,k)
- Ciphertext $C = K \oplus \langle M \parallel c(M) \rangle$
- Transmit v and C over the wireless link

WEP DESCRIPTION

In order to transmit a message M:

- Compute a checksum c(M) (which does nor depend on k)
- Pick an IV v and generate a keystream K=RC4(v,k)
- Ciphertext $C = K \oplus \langle M \| c(M) \rangle$
- Transmit v and C over the wireless link

What kind of cipher is this?

It is a stream cipher (symmetric)

What does the receiver do with v and C?

- Use the received v and the shared k for K = RC4(v, k)
- Decrypt as $K \oplus C = K \oplus K \oplus \langle M' \| c' \rangle = M' \| c'$
- Check to see if $c' = c(M')$
- If it is, accept M′ as the message transmitted

PROBLEM 1 (KEY REUSE)

IV(v) is too short: only three bytes = 24 bits

Secret (k) is rarely changed.

What is the problem with this?

Key-stream gets re-used after 2^{24} iterations

PROBLEM 2 (INTEGRITY BREACH)

The checksum algorithm in WEP is CRC32, which has two important (and undesirable) properties:

- It is independent of k and v
- It is linear: $c(M \oplus \delta) = c(M) \oplus c(\delta)$

? Why is linearity a pessimal property for your integrity mechanism to have when used in conjunction with a stream cipher?

PROBLEM 2 (INTEGRITY BREACH)

- The sender transmits C and v. If Eve wants to modify the plain text M into $M' = M \oplus \delta$

- Calculate $C' = C \oplus \langle \delta || c(\delta) \rangle$
  Send (C', v) instead of (C, v)
  This passes the integrity check of the recipient

WEP AUTHENTICATION (DISASTER)

WEP's authentication protocol to prove that a client knows k:

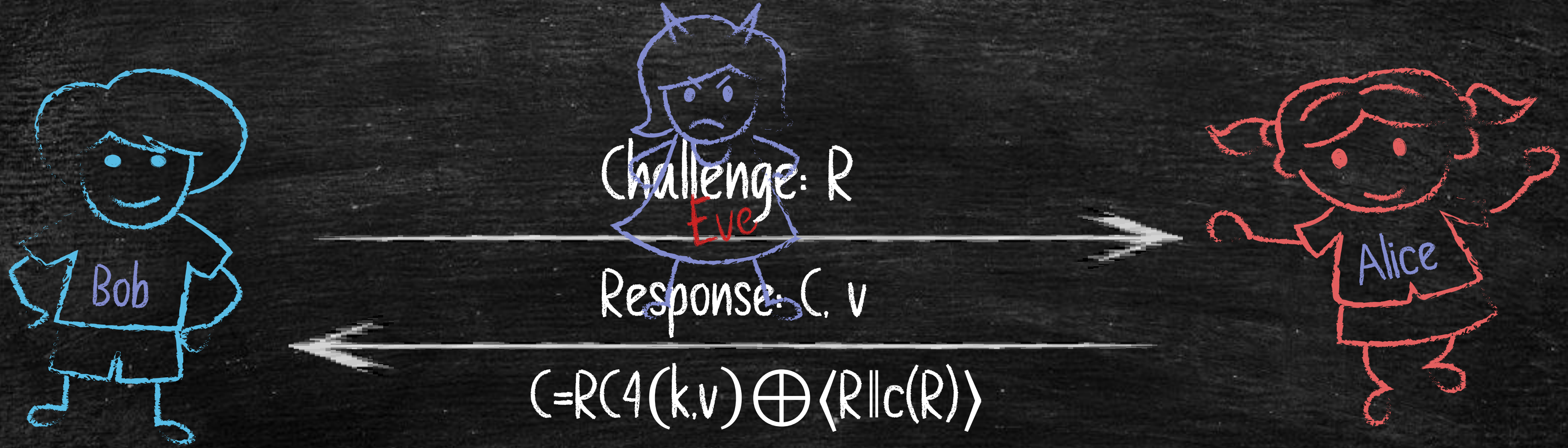
The access point sends a challenge string R to the client

The client sends back the challenge,

WEP-encrypted with the shared secret k

The wireless access point checks if the challenge is correctly encrypted, and if so, accepts the client

The adversary has seen both R and (C, v)


Eve
Challenge: R
Bob
Alice
Response: C, v
C=RC4(k,v) ⊕ ⟨R||c(R)⟩

NETWORK LAYER SECURITY

Suppose every link in our network had strong link-layer security. Why would this not enough?

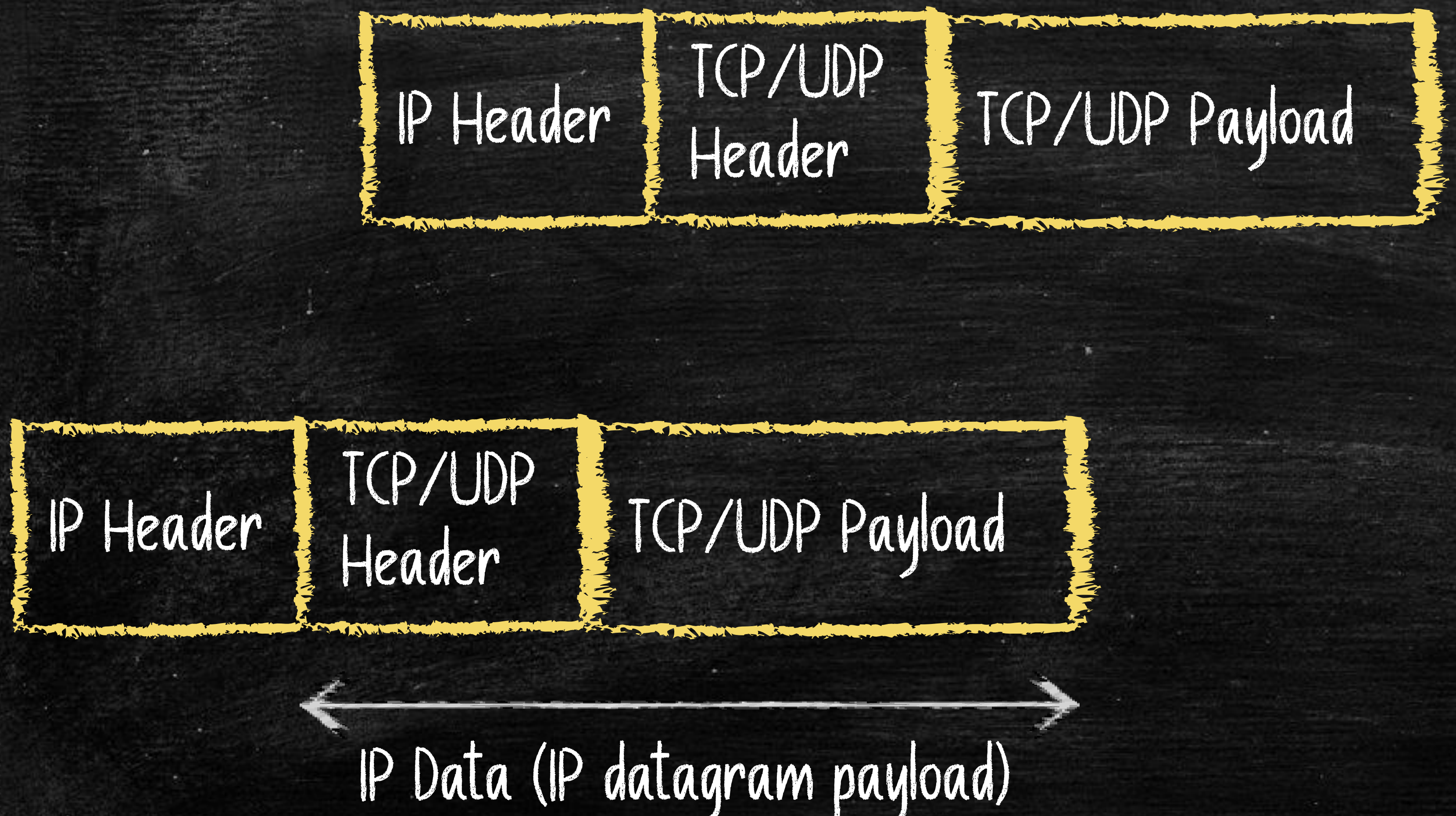
Source, destinations IPs may not share the same link. Network layer threats such as IP spoofing still exist.

We need end-to-end security across networks, i.e., security network layer packets from one host to another so that routers or other hosts in the middle cannot modify or read the packet payload

NETWORK LAYER SECURITY

The IP Security suite (IPSec) extends the Internet Protocol (IP) to provide confidentiality and integrity of packets transmitted across the network. IPSec enables various architectures of Virtual Private Networks (VPNs) which is the foundation in network-layer security

RECALL THE IP-DIAGRAM



IPSEC OVERVIEW

Internet Key Exchange (IKE) to agree on a shared symmetric key.

We use this key to encrypt and compute MACs over IP packets or parts of it.

Modes of operation

Transport mode

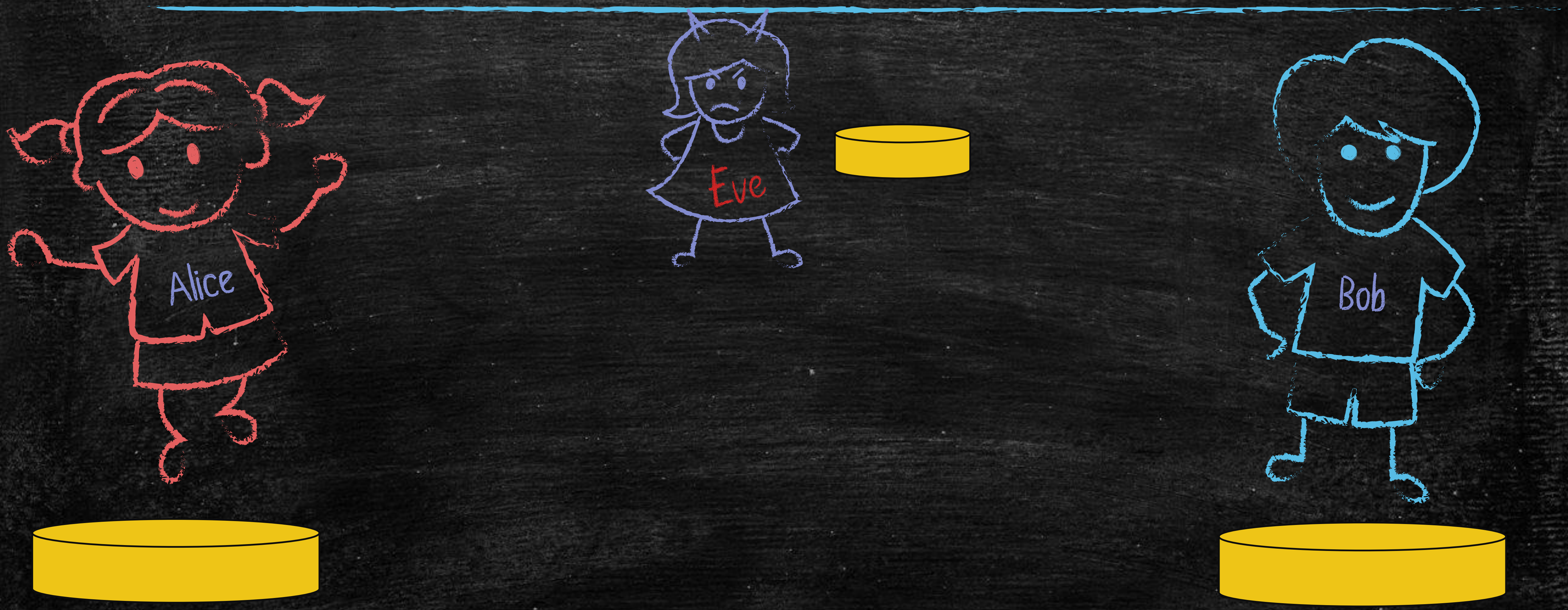
Tunnel mode

Header Types

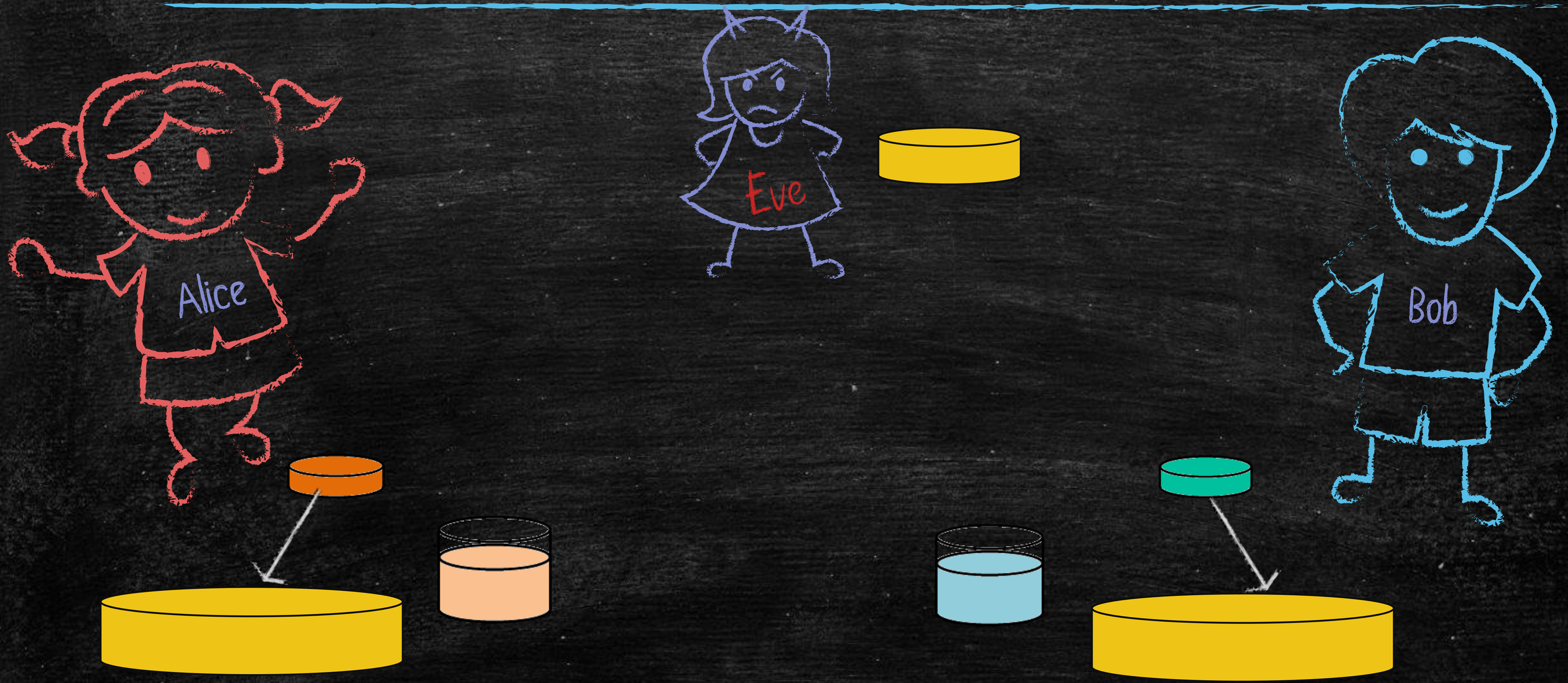
Authentication Header

Encapsulated Security Payload

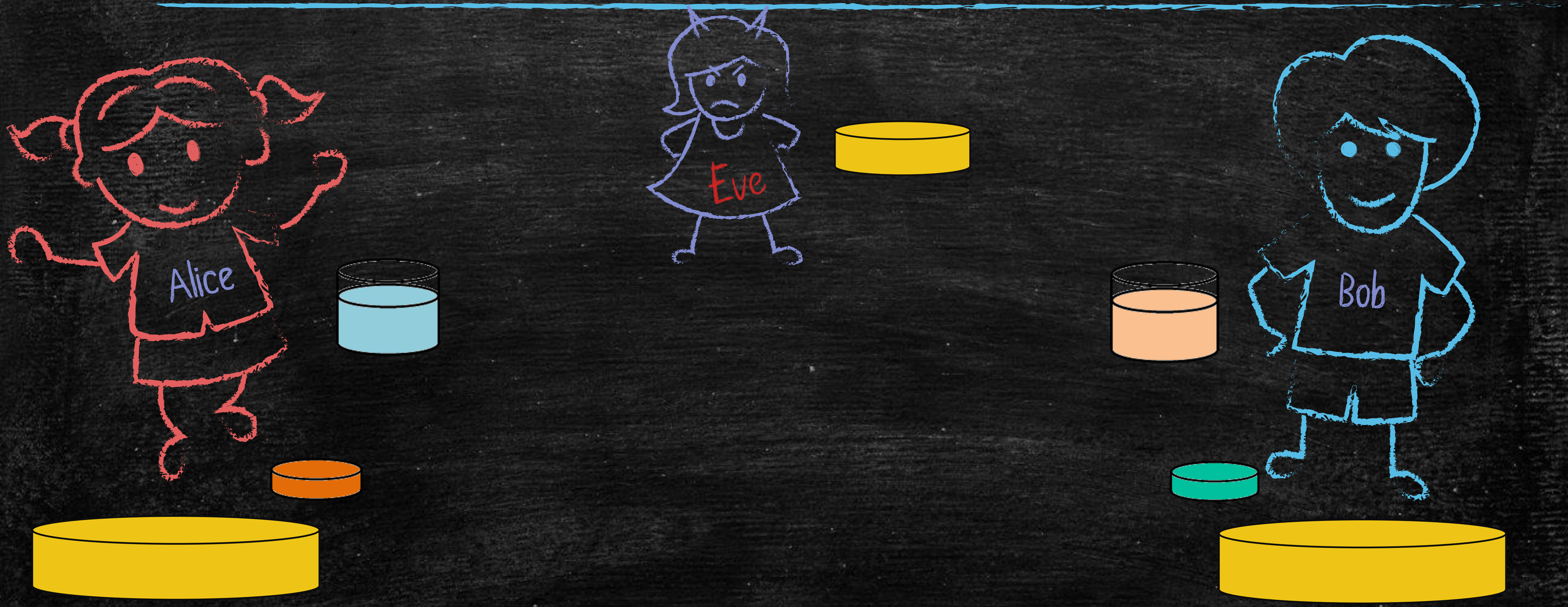
INTERNET KEY-EXCHANGE



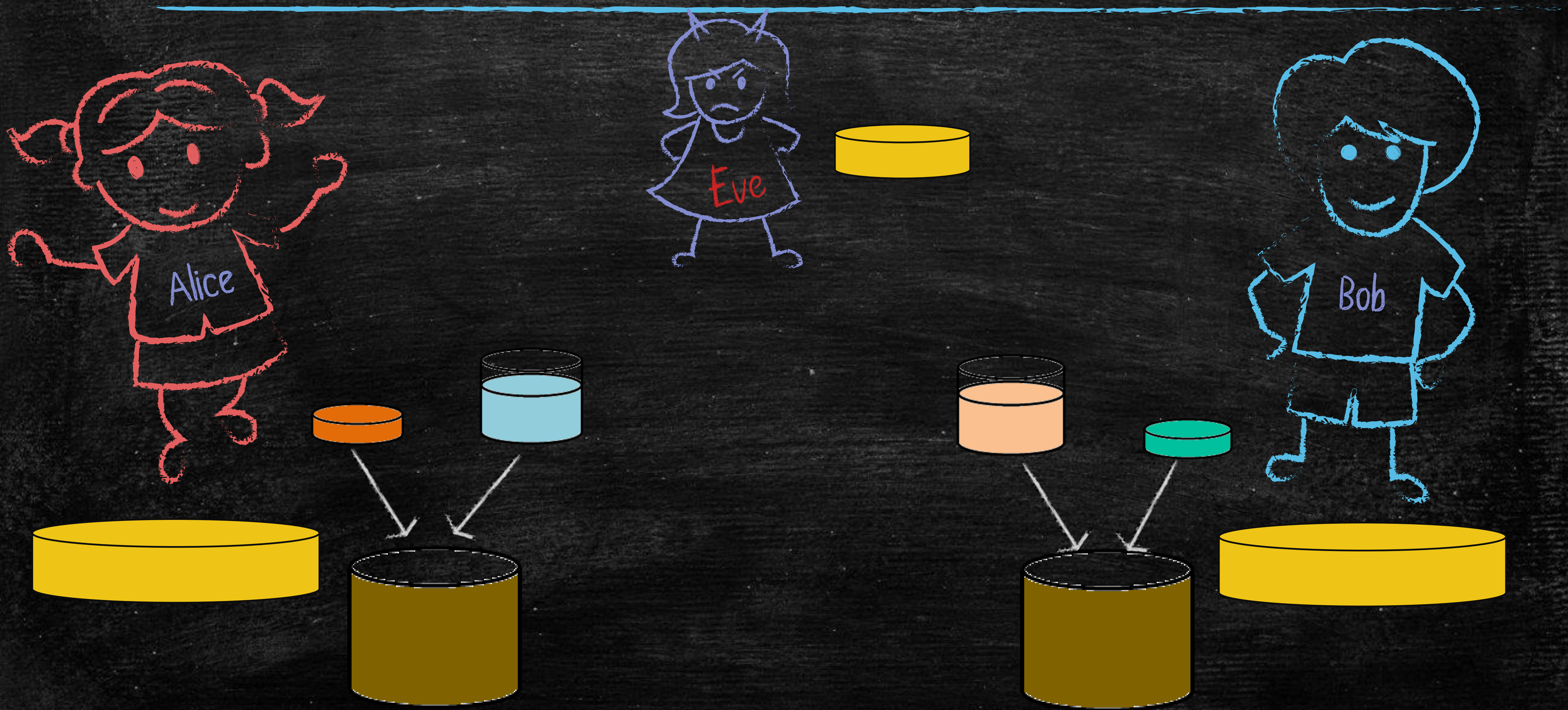
Internet Key-Exchange



INTERNET KEY-EXCHANGE



INTERNET KEY-EXCHANGE




Alice
x
Eve
y
Bob
$(g^x \pmod p), p, g)$
$g^y \pmod p$

MODES OF OPERATION

IPSec has two main modes of operation:

- Transport Mode: uses the original IP header
- Tunnel Mode: encapsulates the original header

TRANSPORT MODE

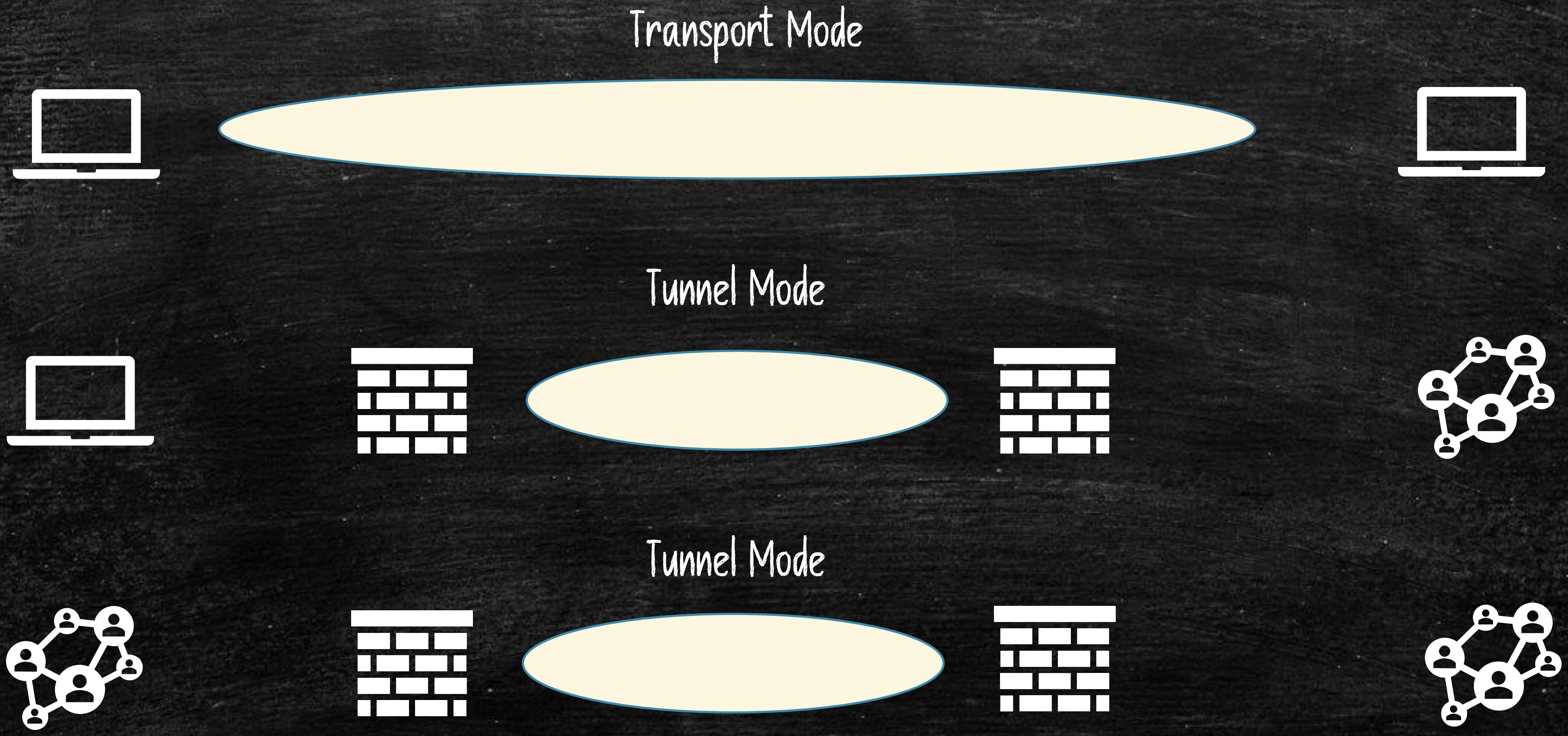
uses the original IP header

Tunnel Mode: encapsulates the original header

TUNNEL MODE

Tunnel Mode: encapsulates the original header

TRANSPORT VS. TUNNEL MODE



Authentication Header

Provides source authentication and data integrity via hash-based MAC

Protects against Replay Attacks by using monotonically increasing sequence numbers.

Does not provide confidentiality

AUTHENTICATION HEADER

IP Header
Sequence Number
Authentication Data
Data

ENCAPSULATING SECURITY PAYLOAD

Provides confidentiality (via Symmetric Key Cryptography)

If you want confidentiality you have to use ESP

If you want integrity only, you could you ESP or AH

If you want to both integrity and confidentiality, use both ESP and AH or only ESP

AH VS ESP

IPSEC HEADERS

Authentication Header (AH)

Offers integrity and data source authentication

Authenticates payload and parts of IP header that do not get modified during transfer, e.g., source IP address

Offers protection against replay attacks

Via extended sequence numbers

Encapsulated Security Payload (ESP)

Offers confidentiality

IP data is encrypted during transmission

Offers authentication functionality similar to AH

But authenticity checks only focus on the IP payload

Applies padding and generates dummy traffic

Makes traffic analysis harder

IPSEC DEPLOYMENT CHALLENGES

Needs to be included in the kernel's network stack.

There may be legitimate reasons to modify some IP header fields: IPSec breaks networking functionalities that require such changes.

- with AH, you cannot replace a private address for a public one at a NAT box.
- with ESP, it depends:
  - In transport usually does not work due to TCP and UDP checksums
  - In tunnel mode it is fine

IPSec is complex, hard to audit, and prone to misconfigurations

TRANSPORT-LAYER SECURITY

Network-layer security mechanisms arrange to send individual IP packets securely from one network to another

Transport-layer security mechanisms transform arbitrary TCP connections to add security and privacy

The main transport-layer security mechanism is TLS (formerly known as SSL)

The main transport-layer privacy mechanism Tor

TLS/SSL

In the mid-1990s, Netscape invented a protocol called Secure Sockets Layer (SSL) meant for protecting HTTP (web) connections

The protocol, however, was general, and could be used to protect any TCP-based Connection HTTP + SSL = HTTPS

Historical note: there was a competing protocol called S-HTTP. But Netscape and Microsoft both chose HTTPS, so that's the protocol everyone else followed

SSL went through a few revisions, and was eventually standardized into the protocol known as TLS (Transport Layer Security, imaginatively enough)

TLS AT A HIGH-LEVEL

Client connects to server, indicates it wants to speak TLS, with:

- Client key-share under ECDHE
- The list of ciphersuites it knows

Server sends its certificate to client, which contains:

- Server key-share under ECDHE
- Its host name,
- Its verification key,
- Some other administrative information,
- A signature from a Certificate Authority (CA)

Both client and server derives the same session key K (which is hard for Eve to derive) based on the two key shares

Server also chooses which ciphersuite to use

All remaining traffic will be encrypted and authenticated under K

TLS Connection Establishment

SECURITY PROPERTIES TLS

Server Authentication

Message Integrity

Message Confidentiality

Client Authentication

CA in TLS

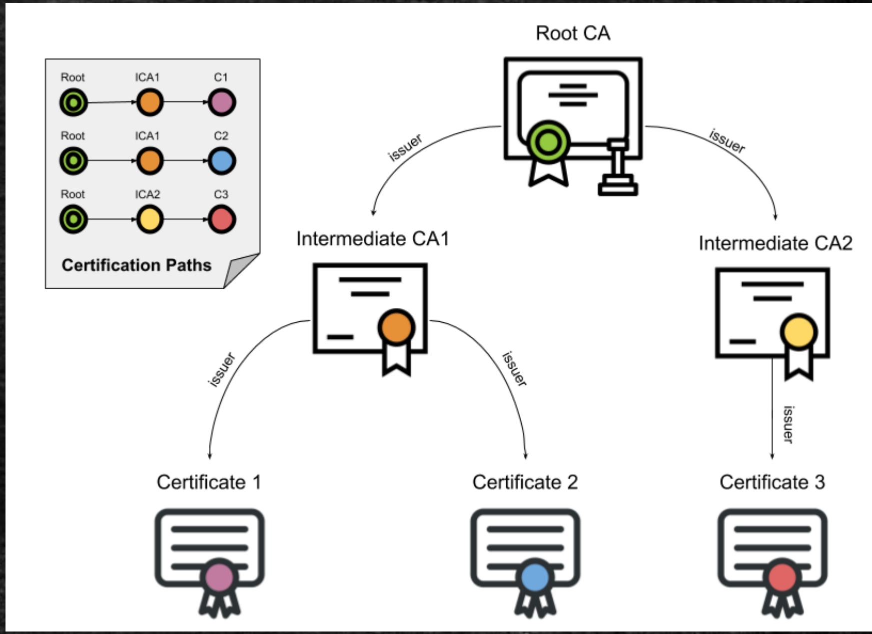
A certification authority acts a trusted third-party that:

- Issues digital certificates
- Certificates the ownership of a public key by the named subject of the certificate
- Manages certificate revocation lists (CRLs)

CA in TLS

A certification authority acts a trusted third-party that:

- Issues digital certificates
- Certificates the ownership of a public key by the named subject of the certificate
- Manages certificate revocation lists (CRLs)



WHAT CAN GO WRONG

Basic Idea: Alice accepts the connection if she receives a certificate

The certificate is signed by a CA she trusts vkCA

The certificate is for the domain she's requesting

When talking to the web server, Alice can verify the signatures with vkWS

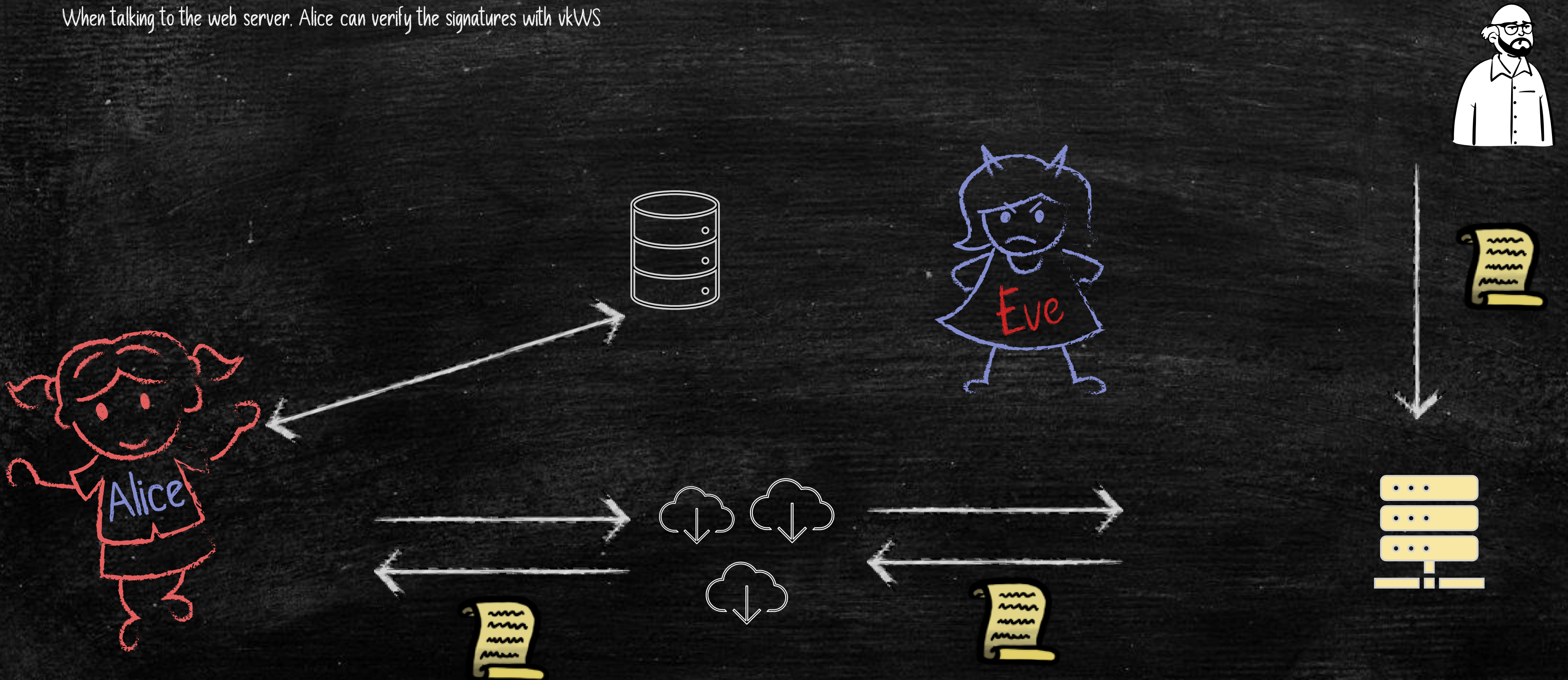
WHAT CAN GO WRONG

Basic Idea: Alice accepts the connection if she receives a certificate

The certificate is signed by a CA she trusts vkCA

The certificate is for the domain she's requesting

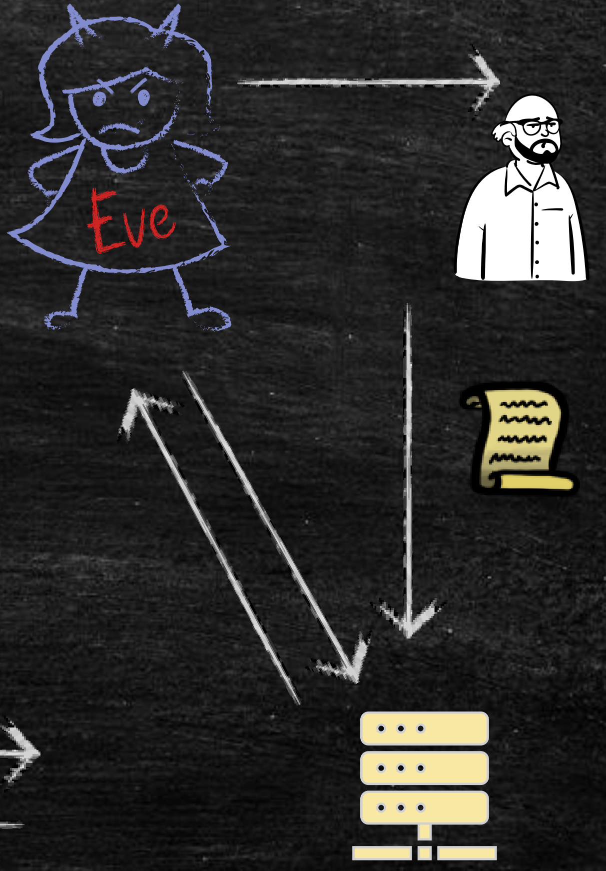
When talking to the web server, Alice can verify the signatures with vkWS

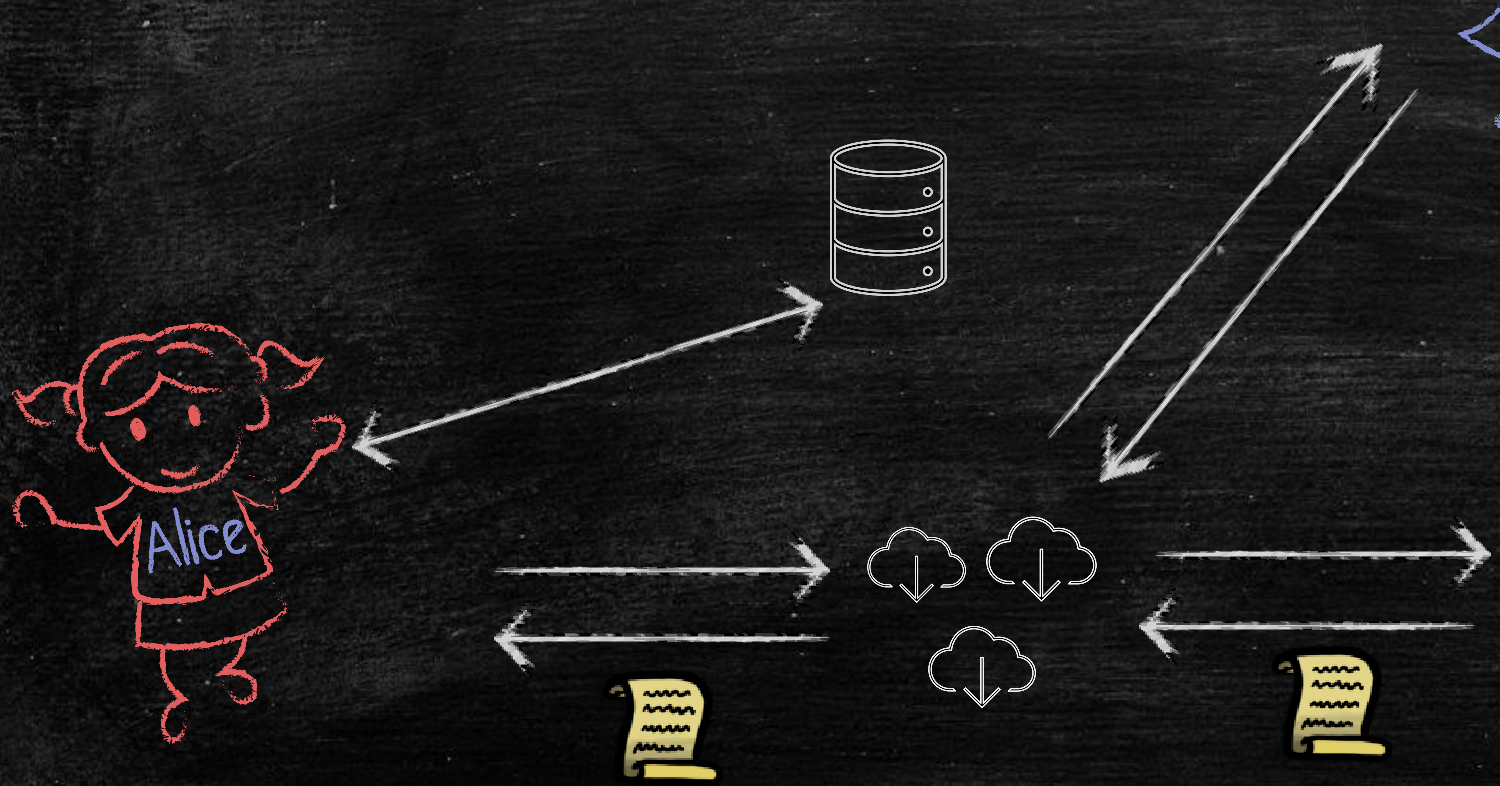


CA in TLS

❓ What can go wrong with TLS?

An adversary can compromise a CA to plant fake certificates
(e.g., DigiNotar's fake *.google.com certificates used by an ISP in Iran)





CA in TLS

What can go wrong with TLS?

An adversary can install a custom CA on users' devices, allowing them to sign certificates that clients will accept for any site (e.g. in 2019, Kazakhstan's ISPs mandated the installation of a root certificate mandated by the government.

WIREGUARD

IPSec

Is complex, hard to audit, and prone to misconfigurations
Big book of IPSec RFCs: Internet security architecture (Loshin, '99)
Does not prevent you from making bad choices
Supports all ciphers, including obsolete ones and NULL

SSL VPNs

Also, on the complex side
Tends to be slow
Also does not prevent you from making bad choices

RECAP OF CRYPTOGRAPHY USE CASES

Link Layer: WEP Problems

Short IV → two-time pad → make it bigger!
Checksum → integrity breach → use MACs
Protocol disaster → packet injection

Network layer: IPSec

IKE: Diffie-Hellman
Modes: Transport, Tunnel
Headers: AH, ESP

Transport TLS

Protocol summary (ECDHE, etc.)
Key management: CAs
Issues with TLS: MITM

Wireguard

Better VPN

RECAP OF CRYPTO TOOLS

Secret Key Cryptography

- One-Time Pads
- Stream Ciphers
- Block Ciphers

Public Key Cryptography

- Textbook RSA
- Secret vs. Public Key Cryptography
- Hybrid Cryptography

Integrity

- Checksums
- Hash Functions

Authentication

- MACs
- Digital signatures
- Key Management

RECAP QUIZ

- What is Hybrid Cryptography
- Under what conditions does Hashing provide integrity?
- What is the point of MACs?
- What is the point of Digital Signatures?

RECAP QUIZ

- What is one thing which Digital Signatures Provide, that MAC do not?
- What is Repudiation? When do we need it?
- What is the point of CAs?
- What are the problems with Root CA?

Overview of Security Controls

SECURITY CONTROL USING CRYPTOGRAPHY

We use cryptography as security control in situations where trust cannot be assumed

We will focus on network security (link layer, network layer, transport layer, and application layer).

But first, we will see other use cases.

SECURITY CONTROL USING CRYPTOGRAPHY

- Apps can be installed only if digitally signed by the vendor (BlackBerry) or upgraded only if signed by the original developer (Android)
- OS allows execution of programs only if signed (iOS)
- OS allows loading of certified device drivers only (Windows)
- Secure boot: OS components booted only if correctly signed

NETWORK SECURITY AND PRIVACY

Entities you can only communicate with over a network are inherently less trustworthy (e.g., they may not be who they claim to be).
This makes networking a primary scenario for cryptography.

This is a separation of concern, and in particular, "separating the security of the medium from the security of the message"

NETWORKING 101

Alice is sitting in her office at U Waterloo.

She connects her phone to the WiFi.

Goes to Amazon.com

Buys a new laptop.

? How do packets travel in the network?

NETWORKING 101

Link Layer: At the link layer, Alice's mobile device establishes a wireless connection with the Wi-Fi access point. The link layer protocols, such as Wi-Fi (e.g., 802.11), handle the transmission of data between her device and the access point.

NETWORKING 101

Network Layer: Once the Wi-Fi connection is established, Alice's mobile device obtains an IP address through DHCP (Dynamic Host Configuration Protocol). The network layer protocols, such as IP (Internet Protocol), come into play. Alice's mobile device sends an IP packet containing the source and destination IP addresses.

NETWORKING 101

Transport Layer: Alice's mobile device chooses a transport layer protocol, typically TCP (Transmission Control Protocol) for web browsing. A TCP connection is established between her device and Amazon's server. The transport layer segments the data into TCP segments and adds the source and destination port numbers.

NETWORKING 101

Application Layer: At the application layer, Alice's mobile device initiates a request to Amazon's server using an HTTP (Hypertext Transfer Protocol) request. The request includes the specific Amazon URL, such as "https://www.amazon.com," and any additional parameters or data required for the purchase.

JOURNEY OF A PACKET

Application Layer

Transport Layer

Network Layer

Link Layer

Where do we need to apply cryptography?

All the Layers

WEP

> Link Layer

↳ WEP was used for Wireless Networks

WEP

WEP was intended to enforce three security goals:

Data Confidentiality

Prevent an adversary from learning the contents of the wireless traffic

Data Integrity

Prevent an adversary from modifying the wireless traffic or fabricating traffic

Access Control

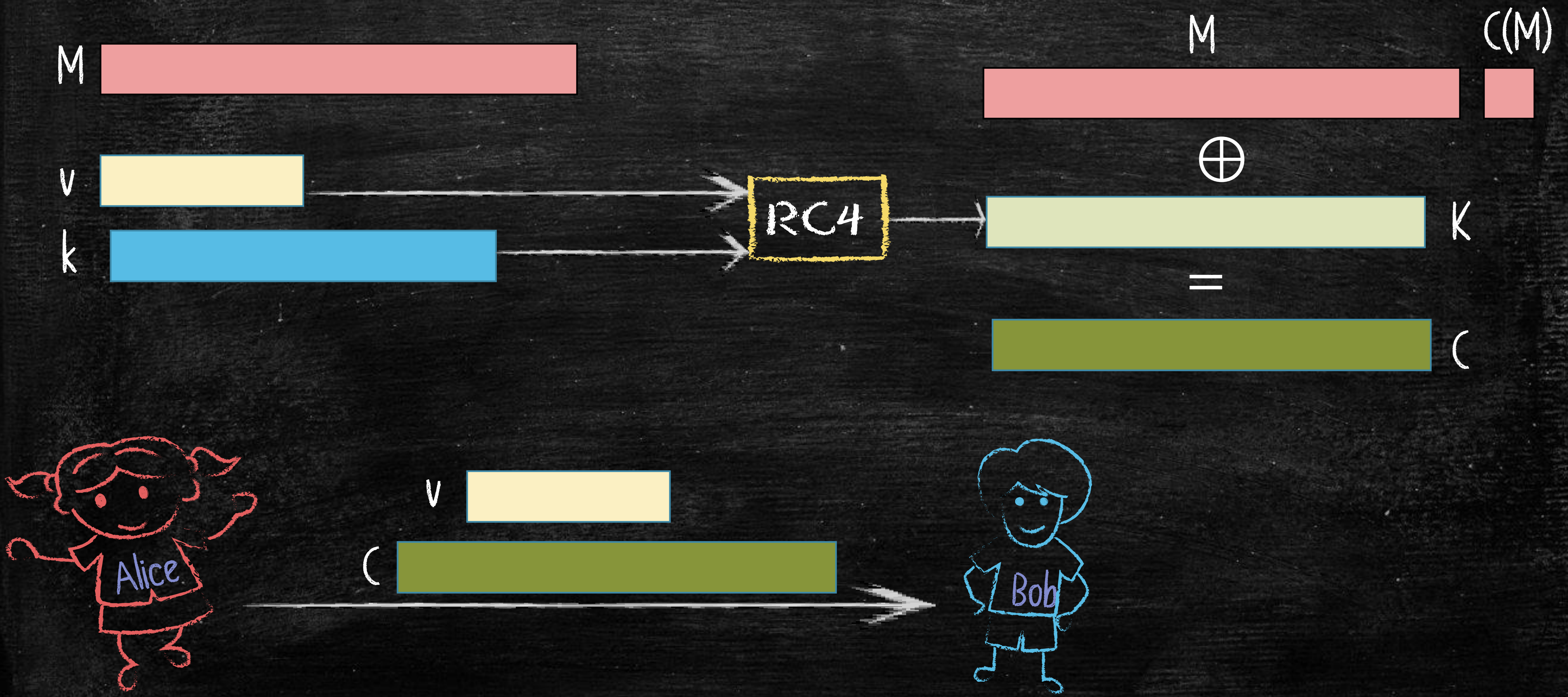
Prevent an adversary from using your wireless infrastructure

WEP

WEP was intended to enforce three security goals:

Unfortunately, none of these is *actually* enforced!!!

WEP



WEP

In order to transmit a message M:

- Compute a checksum c(M) (which does nor depend on k)
- Pick an IV v and generate a keystream K=RC4(v,k)
- Ciphertext $C = K \oplus \langle M \parallel c(M) \rangle$
- Transmit v and C over the wireless link

WEP DESCRIPTION

In order to transmit a message M:

- Compute a checksum c(M) (which does nor depend on k)
- Pick an IV v and generate a keystream K=RC4(v,k)
- Ciphertext $C = K \oplus \langle M \| c(M) \rangle$
- Transmit v and C over the wireless link

What kind of cipher is this?

It is a stream cipher (symmetric)

What does the receiver do with v and C?

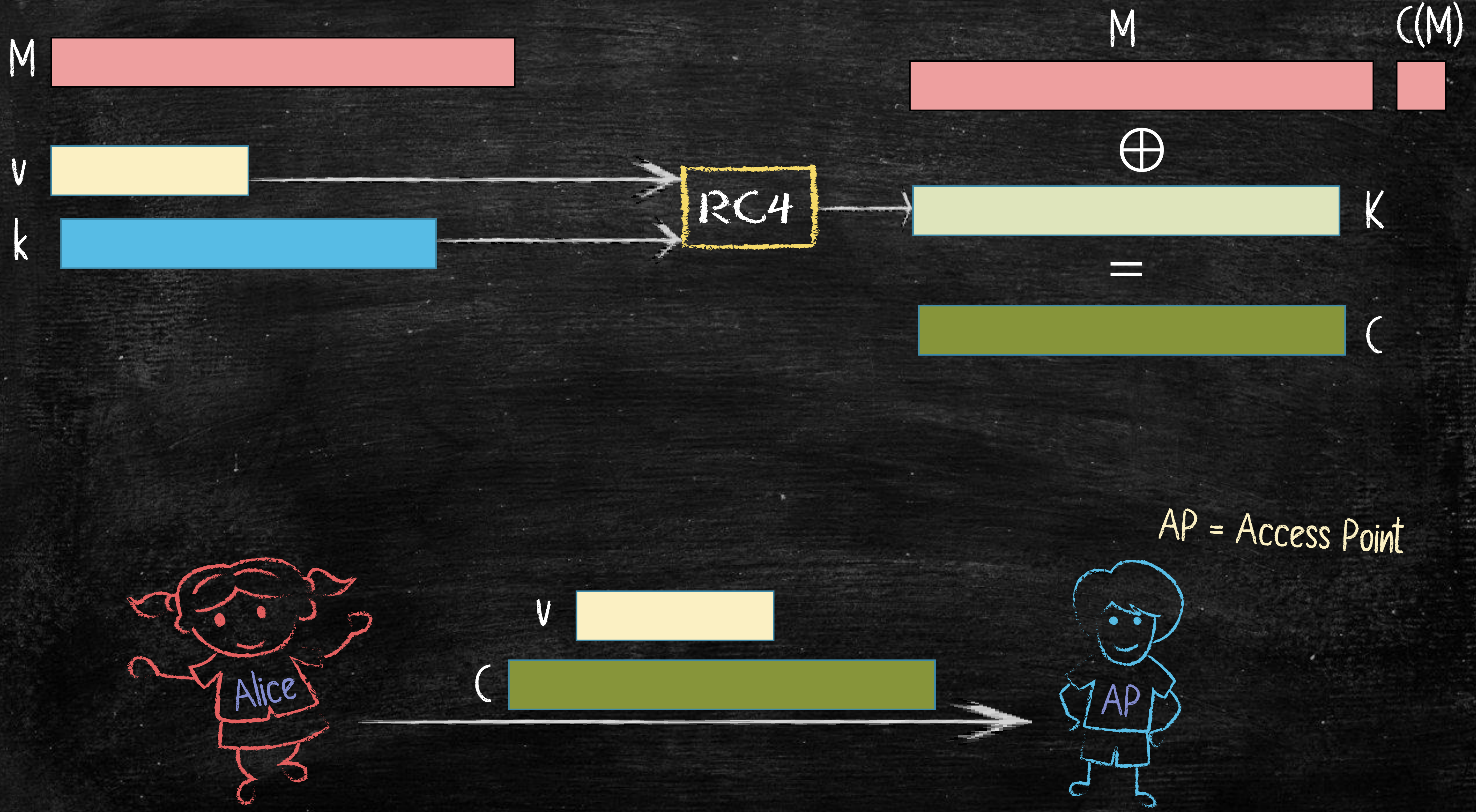
- Use the received v and the shared k for K = RC4(v, k)
- Decrypt as $K \oplus C = K \oplus K \oplus \langle M' \| c' \rangle = M' \| c'$
- Check to see if $c' = c(M')$
- If it is, accept M′ as the message transmitted

PROBLEM 1 (KEY REUSE)

!! IV(v) is too short: only three bytes = 24 bits
Secret (k) is rarely changed.

? What is the problem with this?

Key-stream gets re-used after 2^{24} iterations



M
v
k
RC4
M
C(M)
⊕
K
=
C
AP = Access Point
Alice
v
C
AP

PROBLEM 2 (INTEGRITY BREACH)

The checksum algorithm in WEP is CRC32, which has two important (and undesirable) properties:

- It is independent of k and v
- It is linear: $c(M \oplus \delta) = c(M) \oplus c(\delta)$

? Why is linearity a pessimal property for your integrity mechanism to have when used in conjunction with a stream cipher?

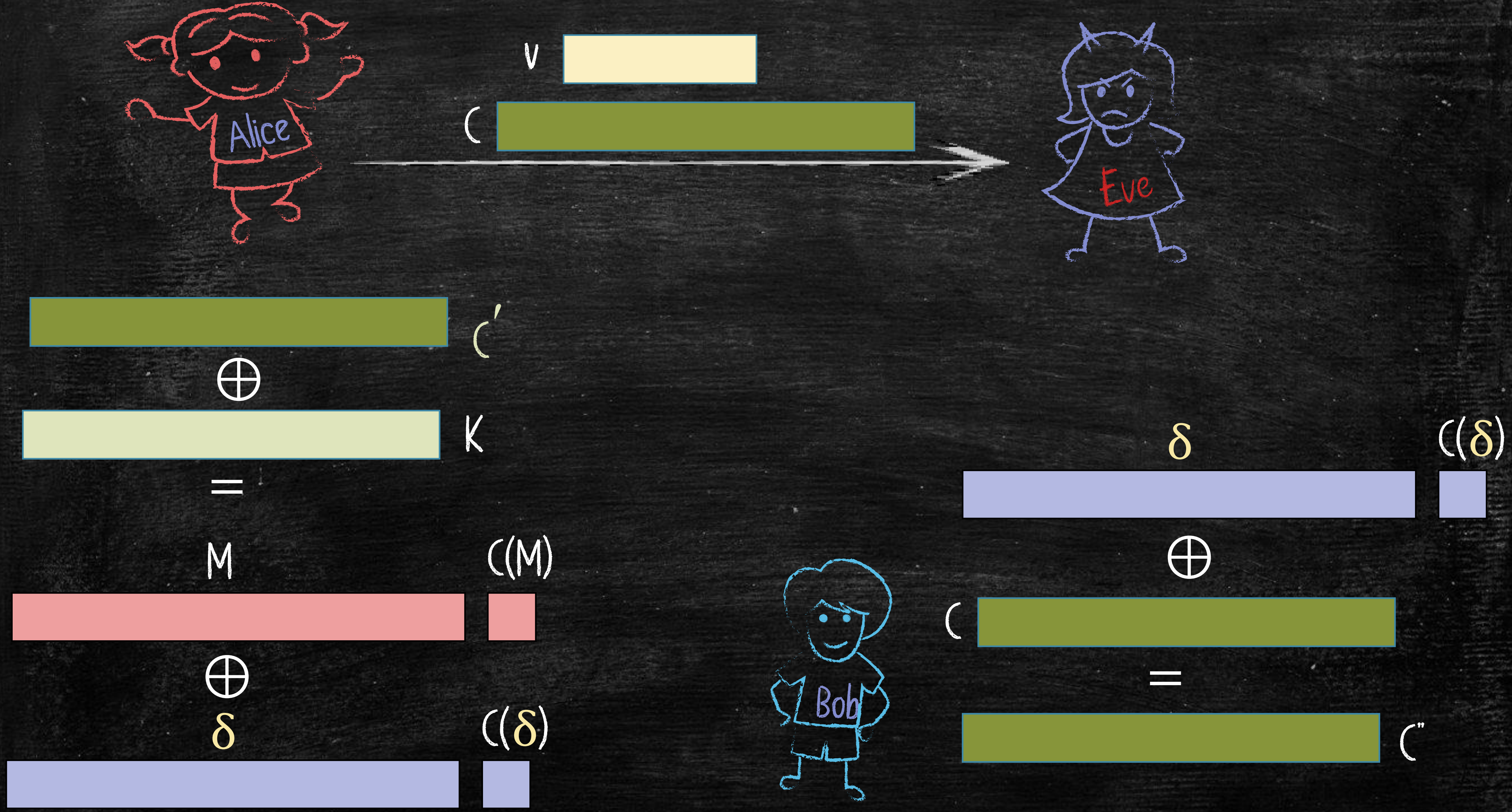
PROBLEM 2 (INTEGRITY BREACH)

- The sender transmits C and v. If Eve wants to modify the plain text M into $M' = M \oplus \delta$

- Calculate $C' = C \oplus \langle \delta || c(\delta) \rangle$

  Send (C', v) instead of (C, v)

  This passes the integrity check of the recipient

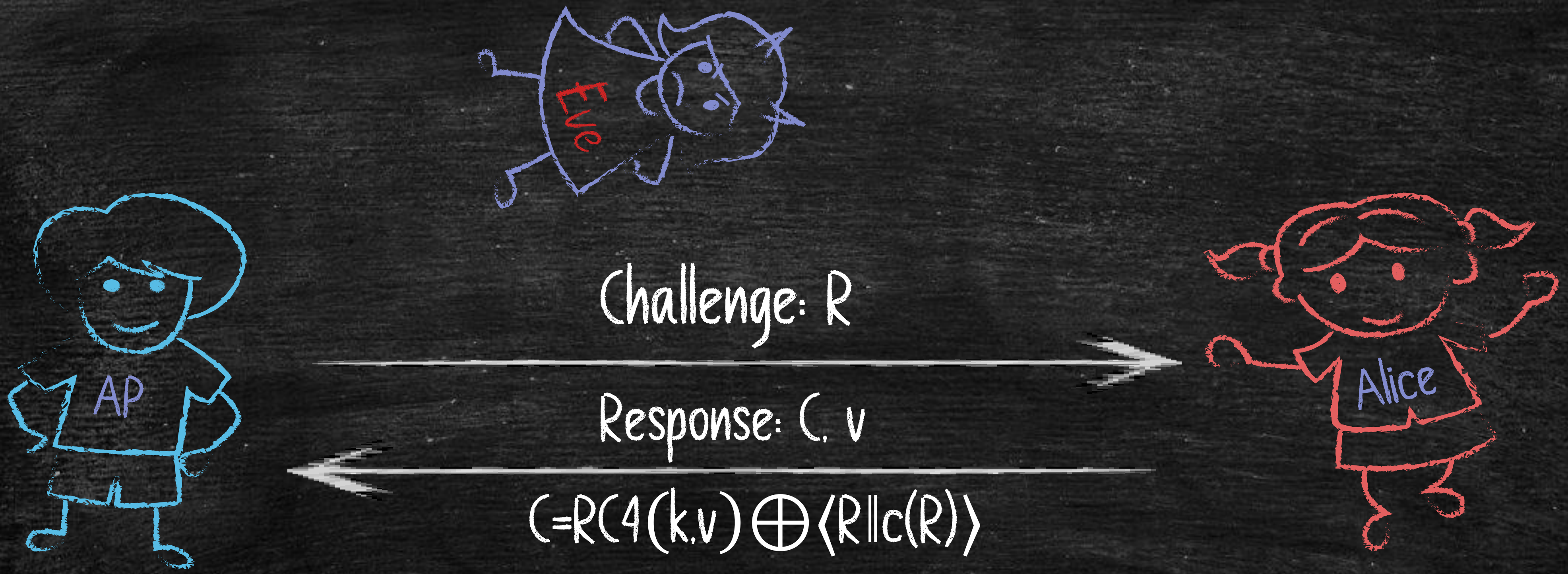

Alice
V
C
Eve
C'
⊕
K
=
M
C(M)
⊕
δ
C(δ)
Bob
δ
C(δ)
⊕
C
=
C''

WEP AUTHENTICATION (DISASTER)

WEP's authentication protocol to prove that a client knows k:

- The access point sends a challenge string R to the client
- The client sends back the challenge, WEP-encrypted with the shared secret k
- The wireless access point checks if the challenge is correctly encrypted, and if so, accepts the client

The adversary has seen both R and (C, v)



The adversary has seen both R and (C, v)

What can Eve do with this information?

Compute a valid v and RC4(k,v)

WEP AUTHENTICATION (DISASTER)

The adversary has seen both R and (C, v)

Eve wants to authenticate herself to the AP. The AP sends Eve a new challenge R′. Can Eve successfully run the authentication protocol?

Yes! Note that Eve knows $RC4(k,v) = C \oplus \langle R \,\|\, c(R) \rangle$. Eve can just compute $C' = RC4(k,v) \oplus \langle R' \,\|\, c(R') \rangle$ and C' and v to the AP

PROBLEM 3 PACKET INJECTION

We saw that seeing R, C, and v gives Eve a value of v and the corresponding keystream RC4(v, k)

The same way Eve encrypted the challenge R' in the previous slide, she can encrypt any other value F: $C' = RC4(k,v) \oplus \langle F \| c(F) \rangle$

Send (C', v) instead of (C, v)

C' is in fact the correct encryption of F, so the message is accepted

MORE PROBLEMS WITH WEP

Somewhat surprisingly, the ability to modify and inject packets leads to ways in which Mallory can trick the AP to decrypt packets!

Note that none of the attacks so far use the fact that the stream cipher was RC4. It turns out that when RC4 is used with similar keys, the output keystream has a subtle weakness, which lead the recovery of either a 104-bit or 40-bit WEP key in under 60 seconds, most of the time.

Check this talk by Ian Goldberg

REPLACING WEP

Wi-fi Protected Access (WPA) was rolled out as a short-term patch to WEP while formal standards for a replacement protocol (IEEE 802.11i, later called WPA2) were being developed

- Replaces CRC-32 with a real MAC
- IV is 48 bits
- Key is changed frequently (TKIP)
- Ability to run on older WEP hardware

WEP RECAP

What have we learned from WEP?

Use sufficiently long IVs, don't share a key with many people, don't reuse short-term secret keys and IVs

Do not use checksums for integrity. Use keyed MACs instead

NETWORK LAYER SECURITY

Suppose every link in our network had strong link-layer security.
Why would this not enough?

Source, destinations IPs may not share the same link. Network layer threats such as IP spoofing still exist.

We need end-to-end security across networks, i.e., security network layer packets from one host to another so that routers or other hosts in the middle cannot modify or read the packet payload

NETWORK LAYER SECURITY

The IP Security suite (IPSec) extends the Internet Protocol (IP) to provide confidentiality and integrity of packets transmitted across the network. IPSec enables various architectures of Virtual Private Networks (VPNs) which is the foundation in network-layer security

IPSEC OVERVIEW

Internet Key Exchange (IKE) to agree on a shared symmetric key.

We use this key to encrypt and compute MACs over IP packets or parts of it.

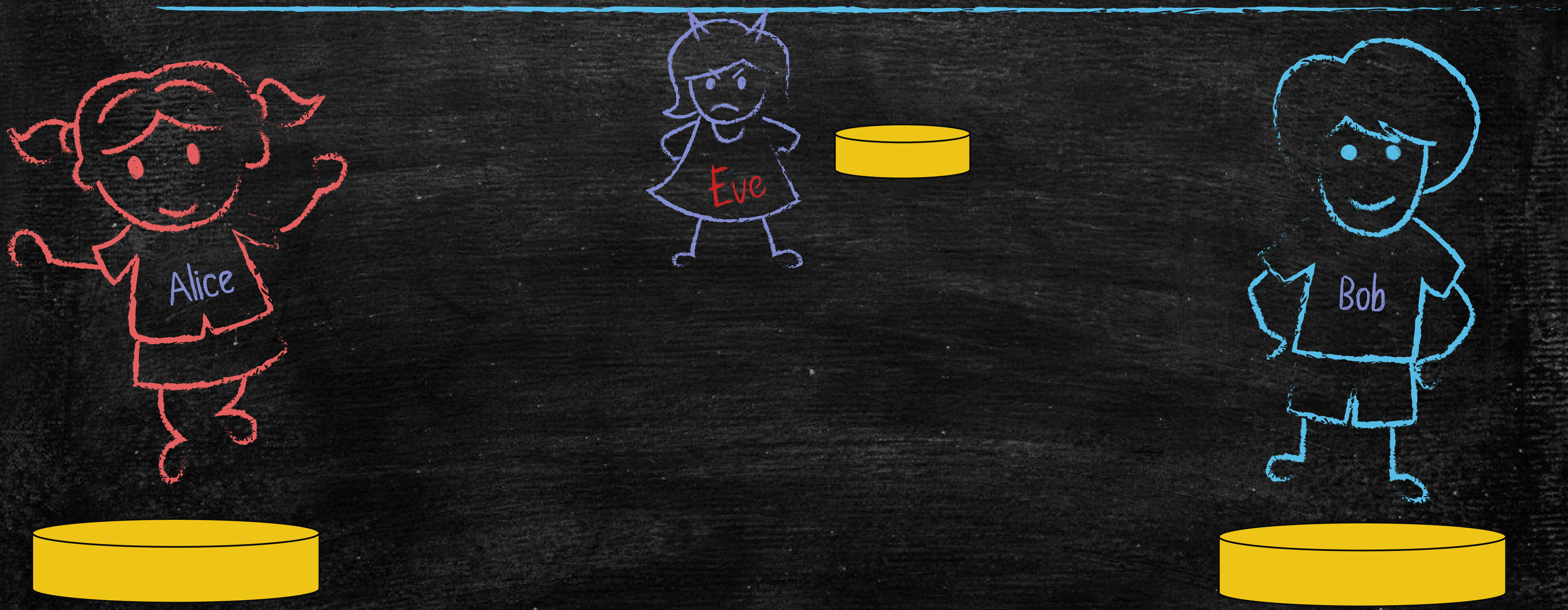
Modes of operation

- Transport mode
- Tunnel mode

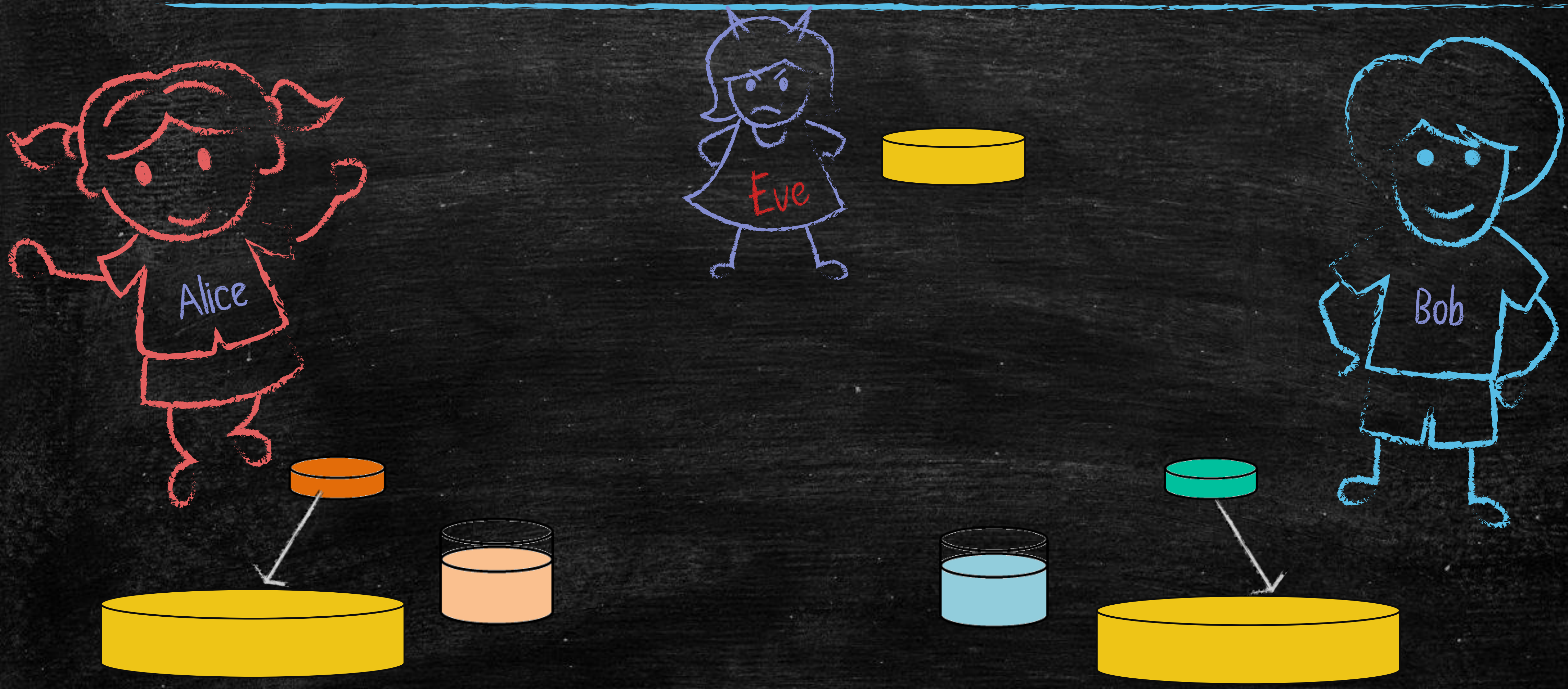
Header Types

- Authentication Header
- Encapsulated Security Payload

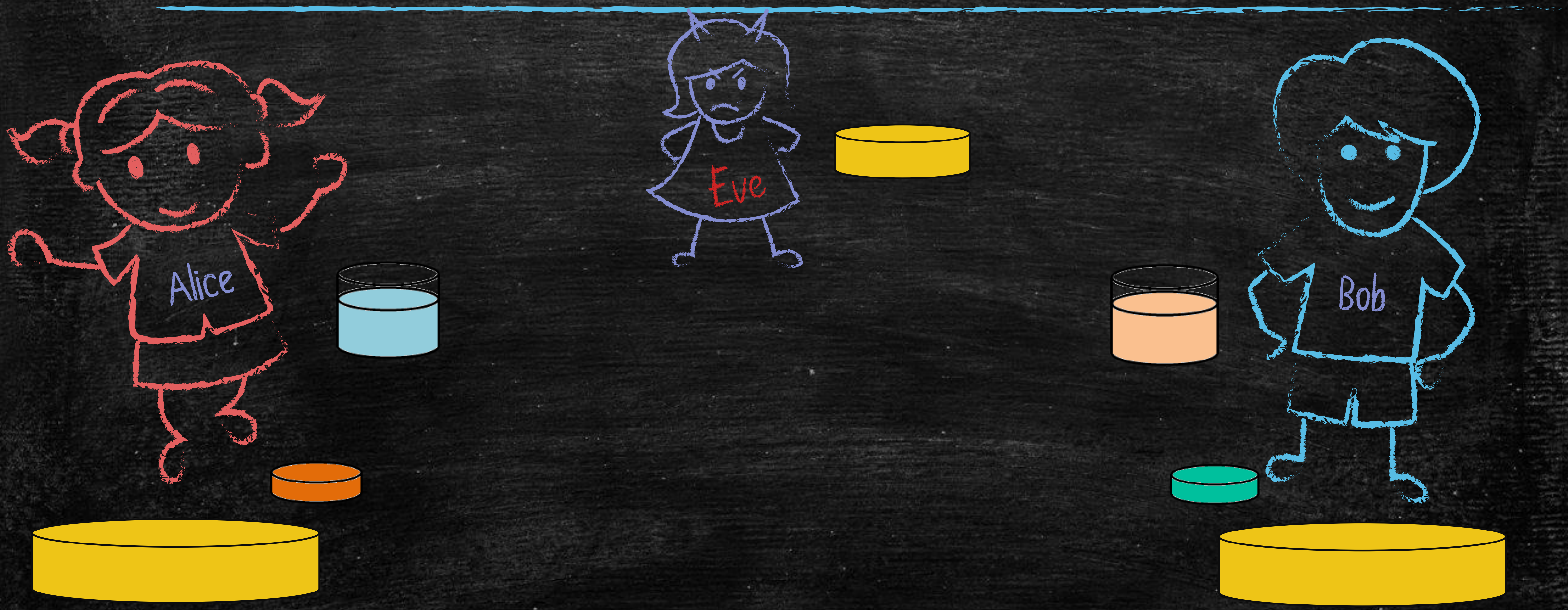
INTERNET KEY-EXCHANGE



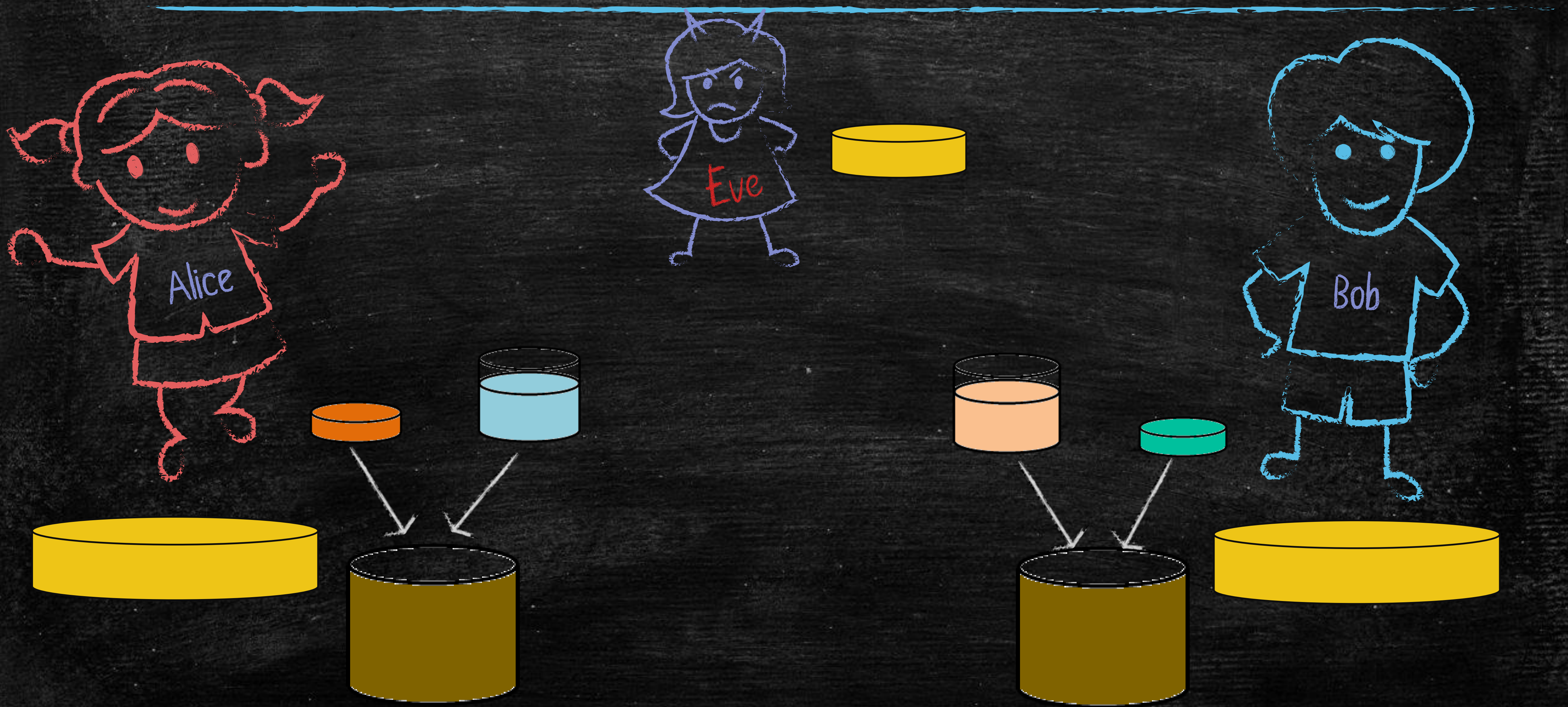
INTERNET KEY-EXCHANGE



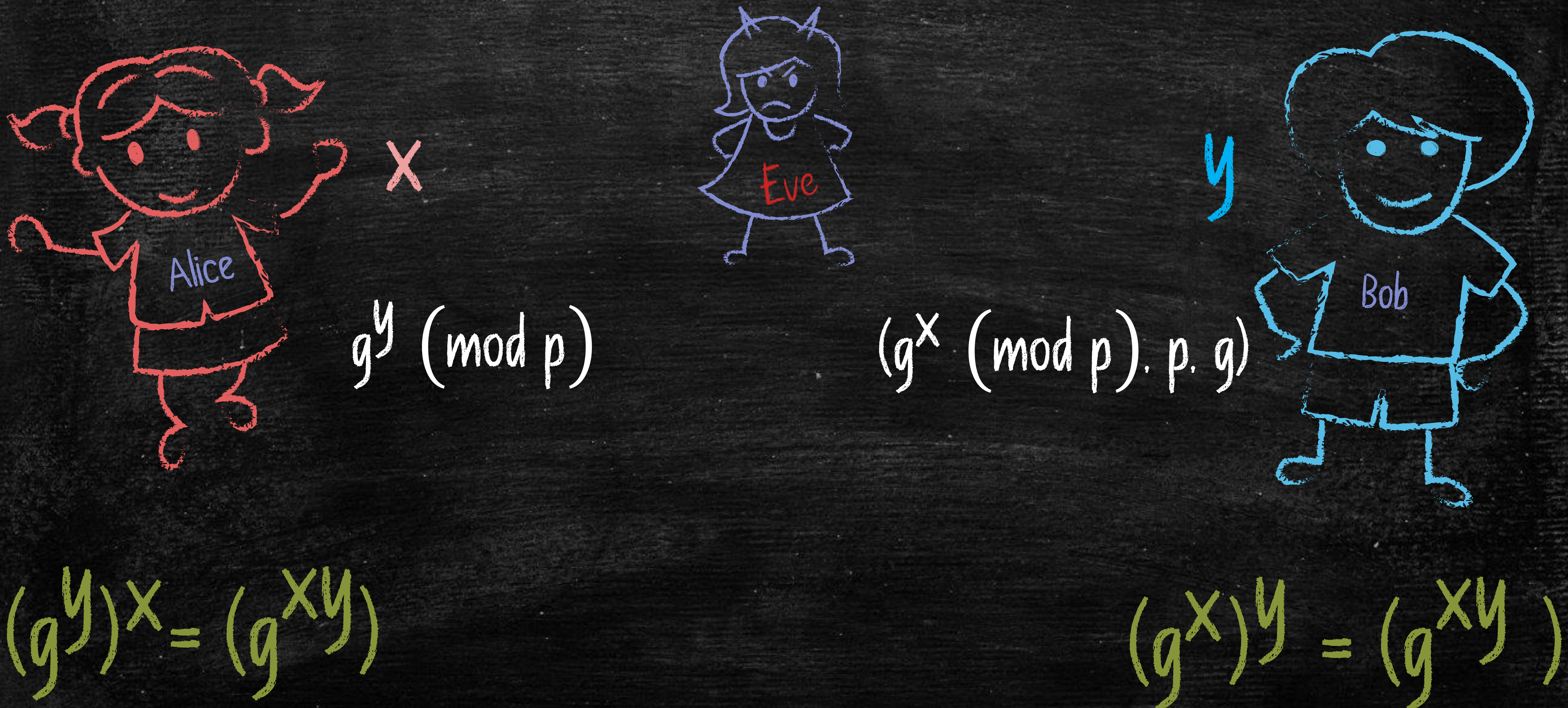
INTERNET KEY-EXCHANGE



INTERNET KEY-EXCHANGE




Alice
x
Eve
y
Bob
(g^x (mod p), p, g)
g^y (mod p)


Alice
x
Eve
y
Bob
$g^y \pmod p$
$(g^x \pmod p), p, g)$
$(g^y)^x = (g^{xy})$
$(g^x)^y = (g^{xy})$

MODES OF OPERATION

IPSec has two main modes of operation:

- Transport Mode
- Tunnel Mode

TRANSPORT MODE

Transport Mode :

The original IP header remains intact.

In transport mode, only the payload (the actual data being transmitted) of the IP packets is encrypted and authenticated.

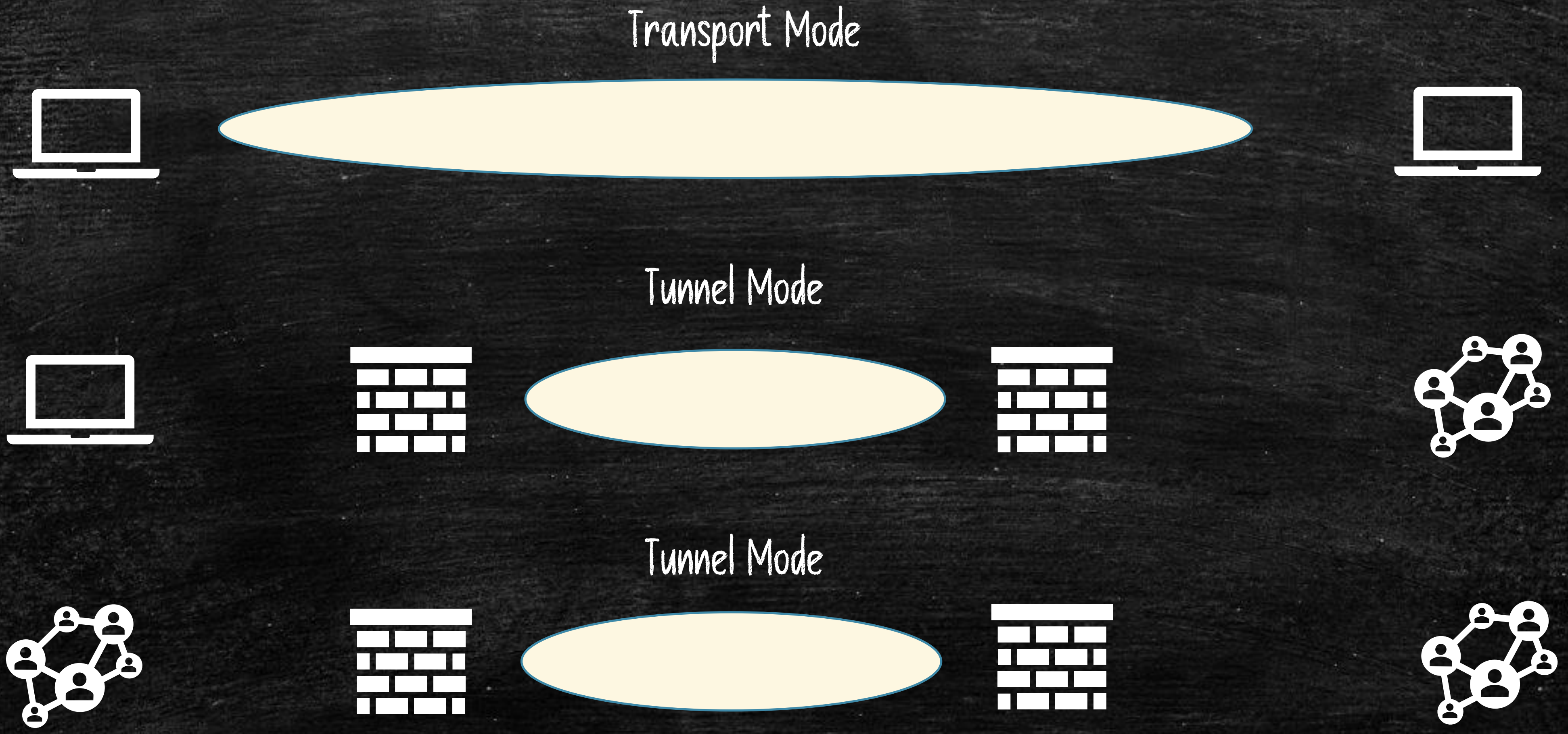
TUNNEL MODE

Tunnel Mode: encapsulates the original header

In tunnel mode, both the original IP header and the payload are encapsulated within a new IP packet. This new packet has a new IP header, which is used to route the traffic between the VPN gateways.

The original IP packet is encrypted and authenticated, providing end-to-end security.

TRANSPORT VS. TUNNEL MODE



TRANSPORT MODE VS. TUNNEL MODE

Transport Mode is typically used for end-to-end communication between two hosts or devices. Transport mode provides protection for the data while it is in transit but does not hide the original IP addresses of the communicating devices.

Tunnel mode is commonly used in site-to-site VPNs, where the entire IP packet is protected and the original source and destination IP addresses are hidden. It allows for secure communication between networks over an untrusted network (such as the Internet).

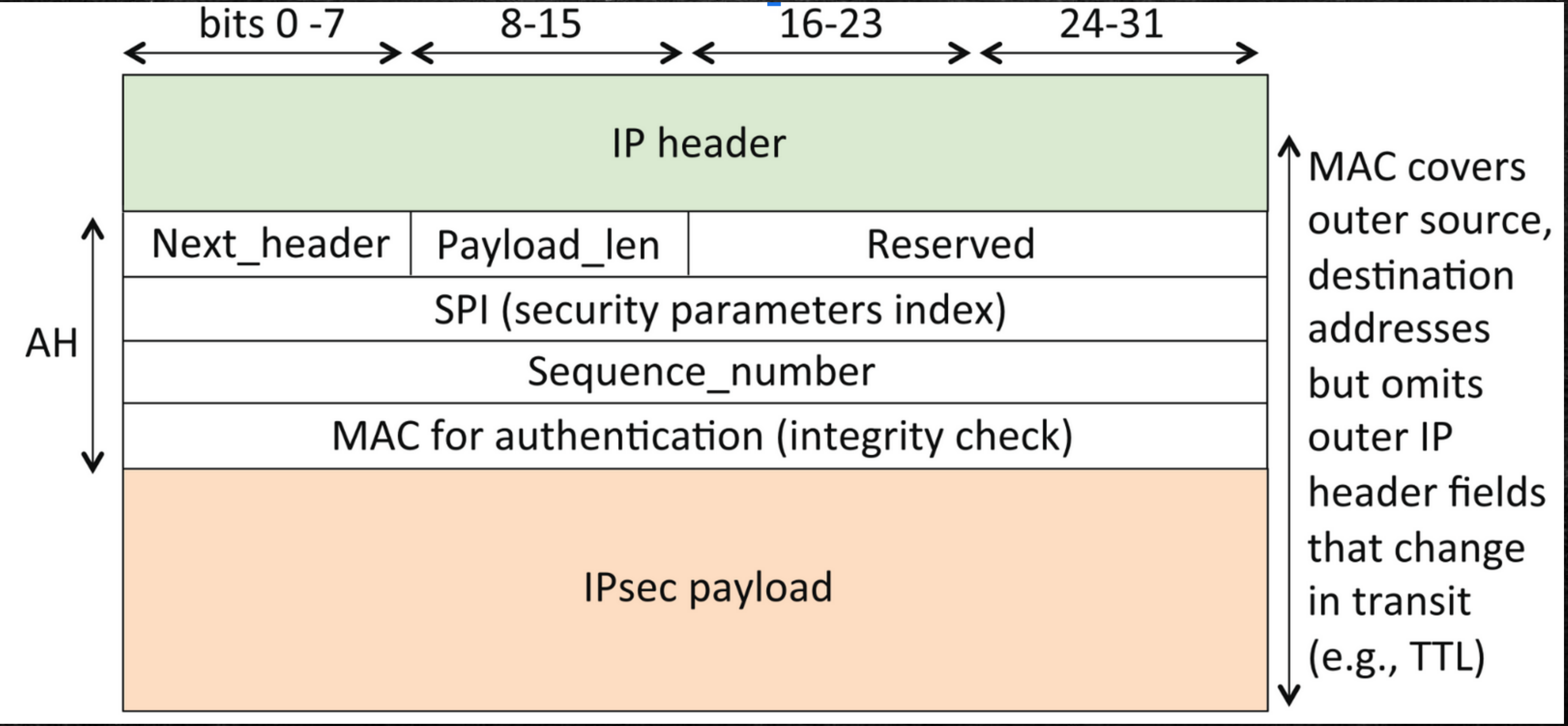
Authentication Header

Provides source authentication and data integrity via hash-based MAC

Protects against Replay Attacks by using monotonically increasing sequence numbers.

Does not provide confidentiality

AUTHENTICATION HEADER



ENCAPSULATING SECURITY PAYLOAD

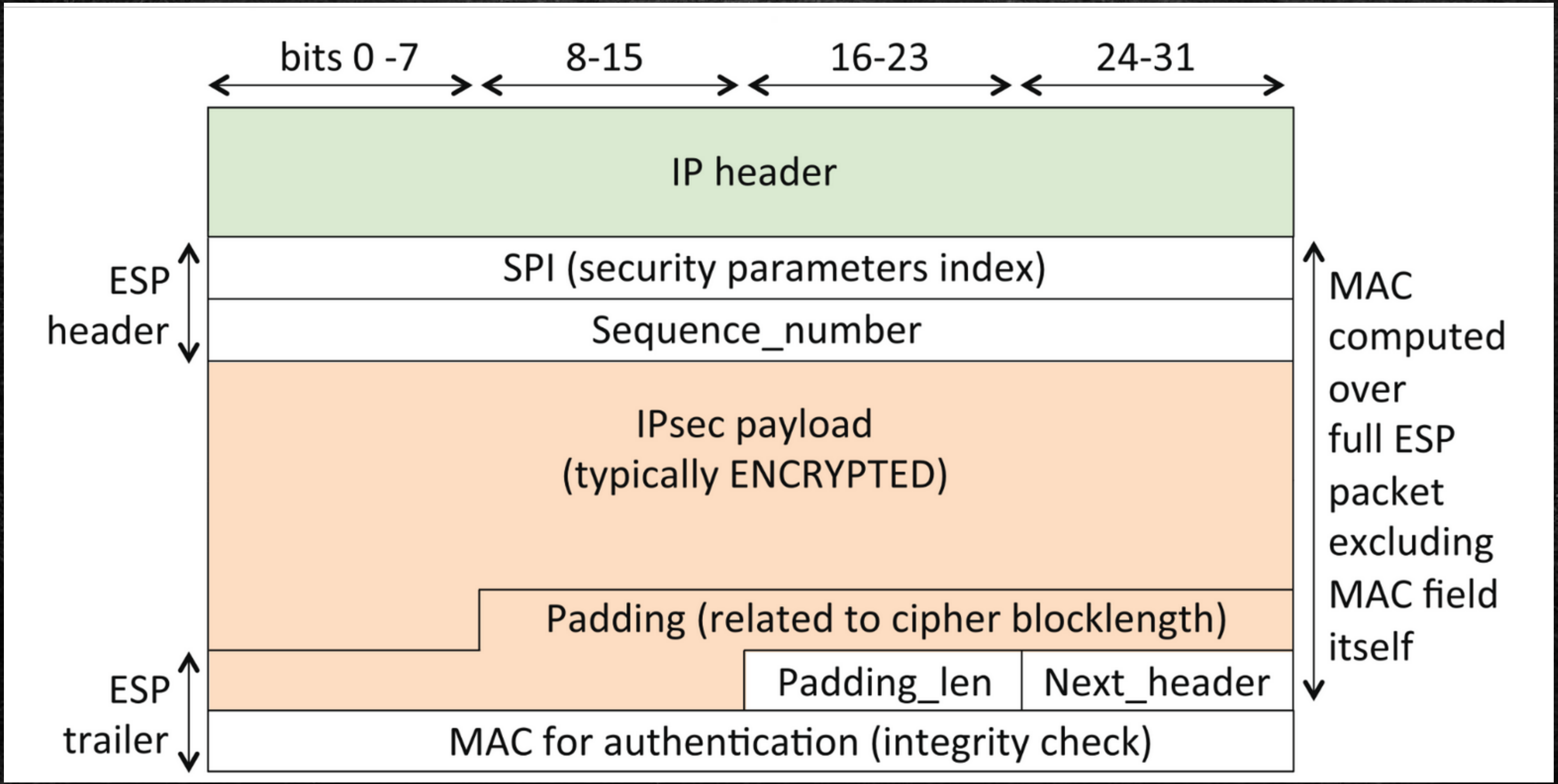
Provides confidentiality (via Symmetric Key Cryptography)

If you want confidentiality you have to use ESP

If you want integrity only, you could you ESP or AH

If you want to both integrity and confidentiality, use both ESP and AH or only ESP

Encapsulating Security Payload



AH VS ESP

Neither of these reasons for the existence of AH is particularly persuasive. The designers of AH/ESP could have made minor modifications to the protocol so that ESP alone could overcome these drawbacks. But there is a more convincing reason given for the existence of AH. At one meeting where the IPSec standard was being developed, "someone from Microsoft gave an impassioned speech about how AH was useless ..." and "... everyone in the room looked around and said, Hmm. He's right, and we hate AH also, but if it annoys Microsoft let's leave it in since we hate Microsoft more than we hate AH" [162]. So now you know the rest of the story.

IPSEC HEADERS

Authentication Header (AH)

Offers integrity and data source authentication

Authenticates payload and parts of IP header that do not get modified during transfer, e.g., source IP address

Offers protection against replay attacks

Via extended sequence numbers

Encapsulated Security Payload (ESP)

Offers confidentiality

IP data is encrypted during transmission

Offers authentication functionality similar to AH

But authenticity checks only focus on the IP payload

Applies padding and generates dummy traffic

Makes traffic analysis harder

IPSEC DEPLOYMENT CHALLENGES

- Needs to be included in the kernel's network stack.
- There may be legitimate reasons to modify some IP header fields: IPSec breaks networking functionalities that require such changes.
  - with AH, you cannot replace a private address for a public one at a NAT box.
  - with ESP, it depends:
    - In transport usually does not work due to TCP and UDP checksums
    - In tunnel mode it is fine
- IPSec is complex, hard to audit, and prone to misconfigurations

IPSEC

What does IPSec protect us against and what does it not protect.

IPSec lets you make a secure tunnel between Alice and a VPN server. This is protecting against eavesdroppers on Alice's network but not against the VPN server itself or eavesdroppers on the VPN server's network or along the way to the actual destination.

TRANSPORT-LAYER SECURITY

Network-layer security mechanisms arrange to send individual IP packets securely from one network to another

Transport-layer security mechanisms transform arbitrary TCP connections to add security and privacy

TRANSPORT-LAYER SECURITY

The main transport-layer security mechanism is TLS (formerly known as SSL)

The main transport-layer privacy mechanism Tor

TLS/SSL

In the mid-1990s, Netscape invented a protocol called Secure Sockets Layer (SSL) meant for protecting HTTP (web) connections

The protocol, however, was general, and could be used to protect any TCP-based Connection HTTP + SSL = HTTPS

TLS/SSL

Historical note: there was a competing protocol called S-HTTP. But Netscape and Microsoft both chose HTTPS, so that's the protocol everyone else followed

SSL went through a few revisions, and was eventually standardized into the protocol known as TLS (Transport Layer Security, imaginatively enough)

TLS AT A HIGH-LEVEL

Client connects to server, indicates it wants to speak TLS, with:

- Client key-share under ECDHE
- The list of ciphersuites it knows

Server sends its certificate to client, which contains:

- Server key-share under ECDHE
- Its host name,
- Its verification key,
- Some other administrative information,
- A signature from a Certificate Authority (CA)

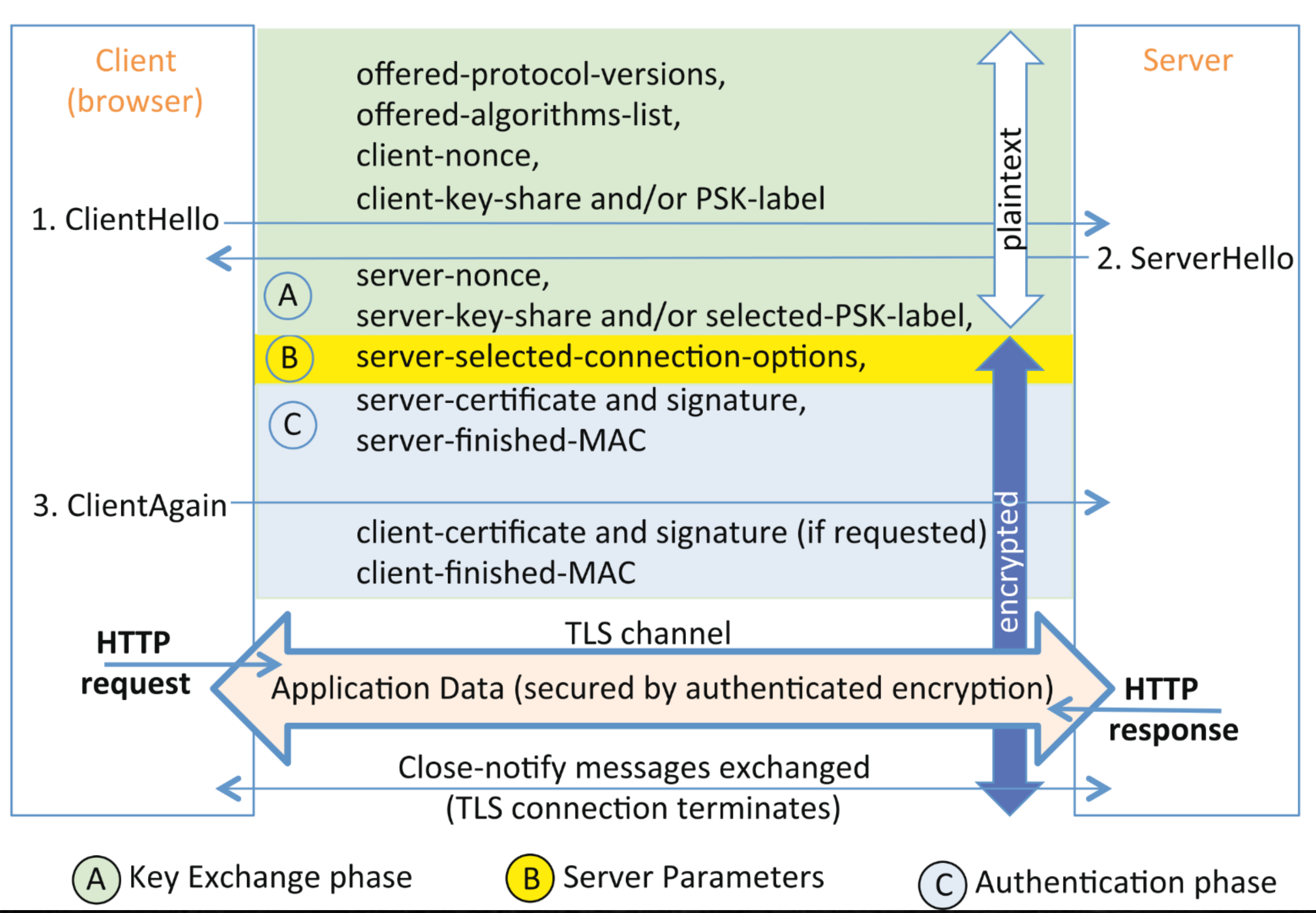
TLS AT A HIGH-LEVEL

Both client and server derives the same session key K (which is hard for Eve to derive) based on the two key shares

Server also chooses which ciphersuite to use

All remaining traffic will be encrypted and authenticated under K

TLS CONNECTION ESTABLISHMENT



SECURITY PROPERTIES TLS

Server Authentication

Message Integrity

Message Confidentiality

Client Authentication

CA in TLS

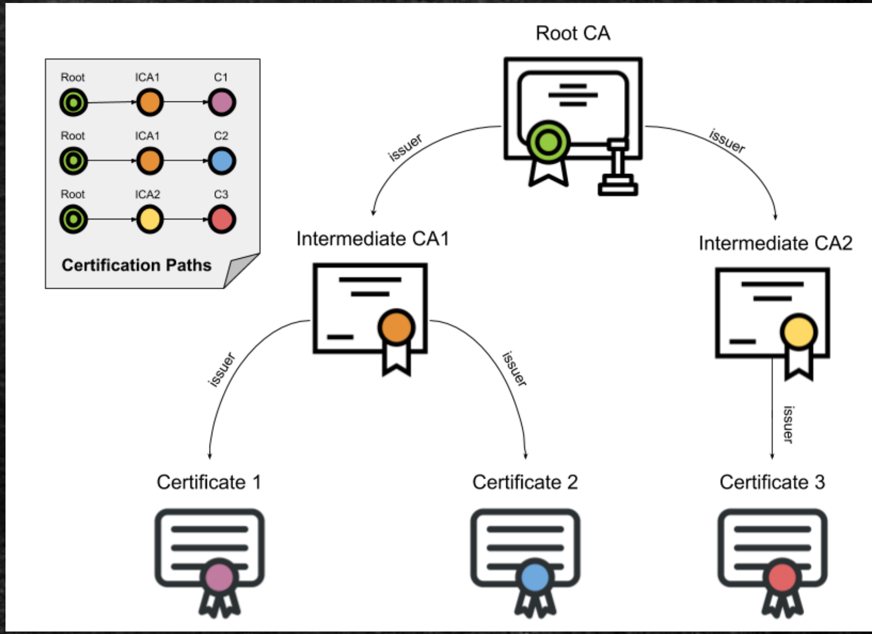
A certification authority acts a trusted third-party that:

- Issues digital certificates
- Certificates the ownership of a public key by the named subject of the certificate
- Manages certificate revocation lists (CRLs)

CA in TLS

A certification authority acts a trusted third-party that:

- Issues digital certificates
- Certificates the ownership of a public key by the named subject of the certificate
- Manages certificate revocation lists (CRLs)



WHAT CAN GO WRONG

Basic Idea: Alice accepts the connection if she receives a certificate

The certificate is signed by a CA she trusts vkCA

The certificate is for the domain she's requesting

When talking to the web server, Alice can verify the signatures with vkWS

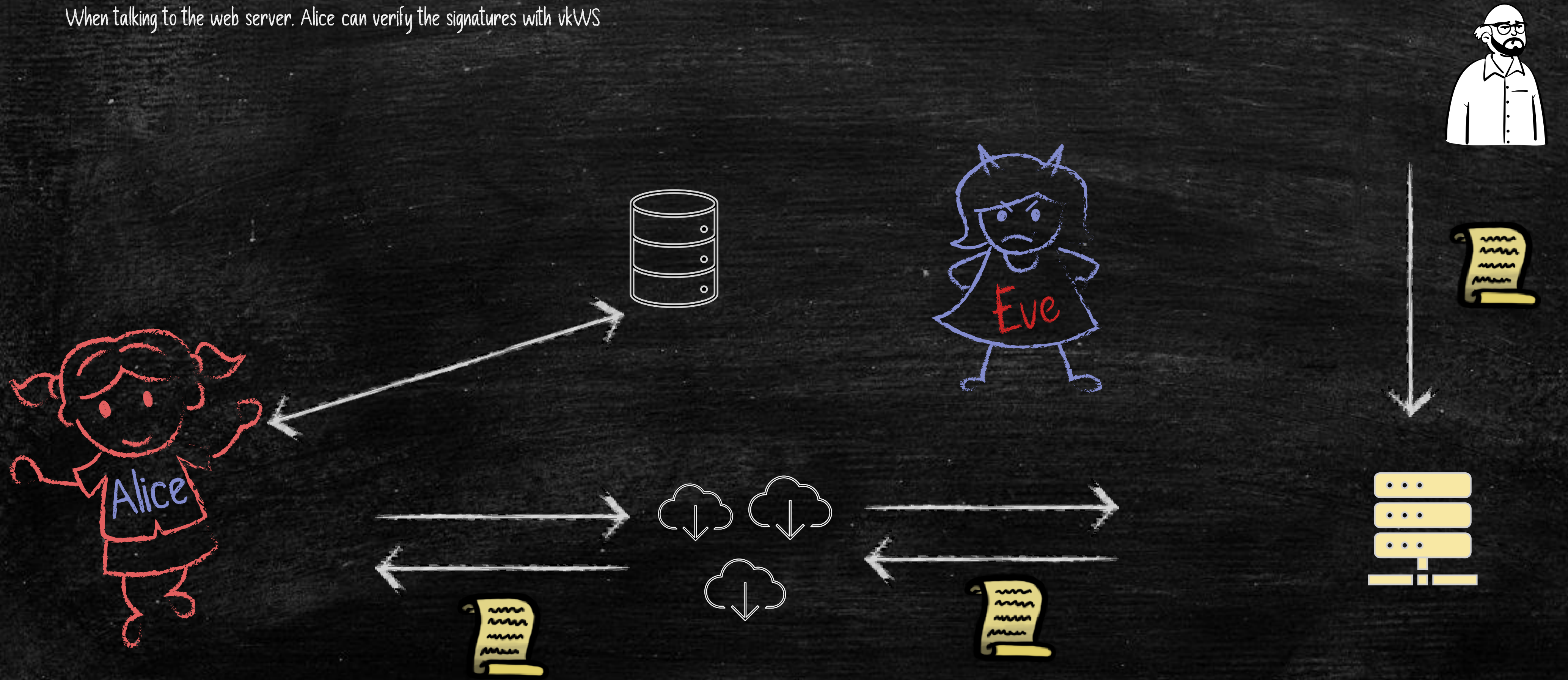
WHAT CAN GO WRONG

Basic Idea: Alice accepts the connection if she receives a certificate

The certificate is signed by a CA she trusts vkCA

The certificate is for the domain she's requesting

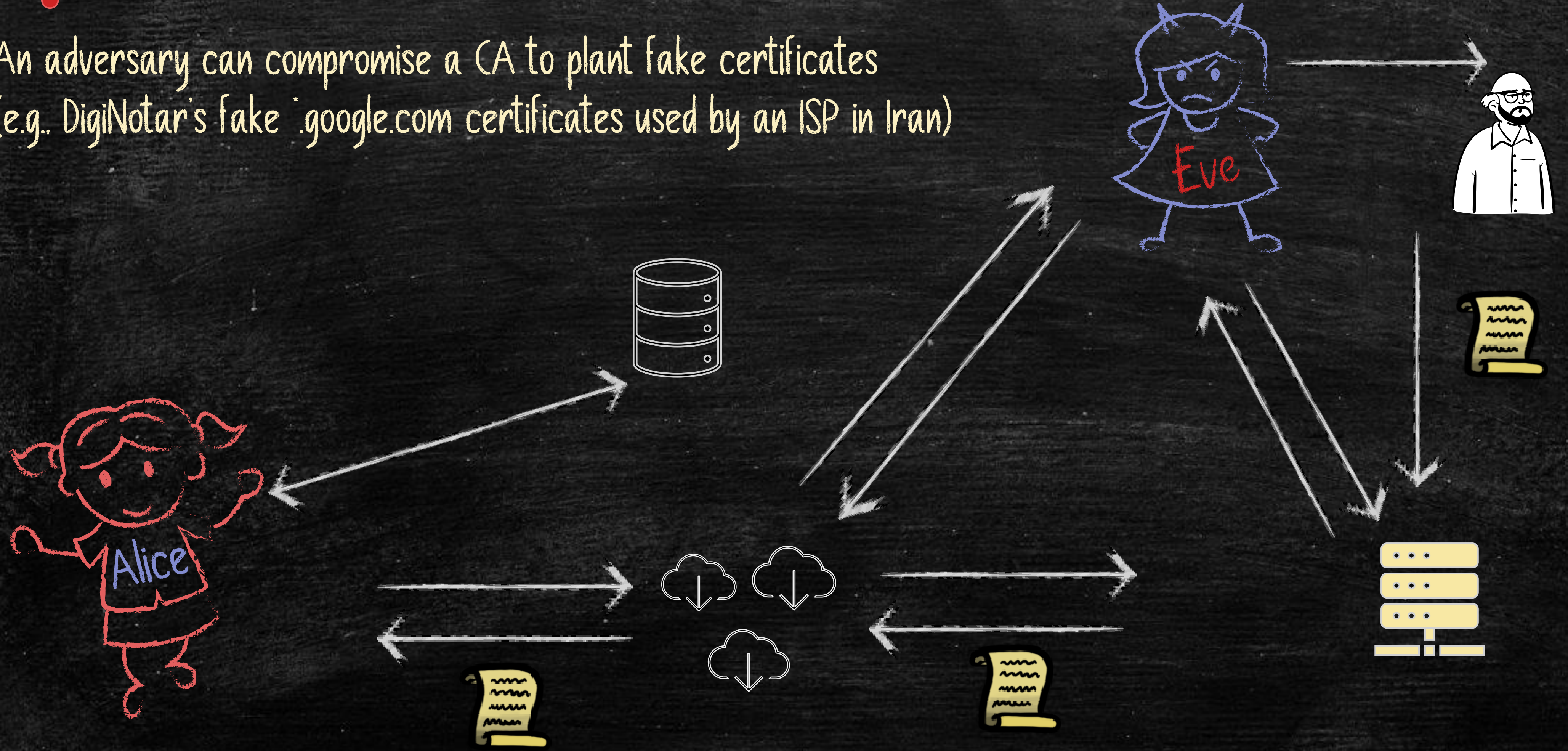
When talking to the web server, Alice can verify the signatures with vkWS



CA in TLS

❓ What can go wrong with TLS?

An adversary can compromise a CA to plant fake certificates
(e.g., DigiNotar's fake *.google.com certificates used by an ISP in Iran)



CA in TLS

❓ What else can go wrong with TLS?

An adversary can install a custom CA on users' devices, allowing them to sign certificates that clients will accept for any site (e.g. in 2019, Kazakhstan's ISPs mandated the installation of a root certificate mandated by the government.

CA in TLS

What else else can go wrong with TLS?

Companies may think it is an excellent idea e.g., Lenovo's Superfish or Sennheiser HeadSetup root certificates (for advertisement and communication purposes, respectively)

There have been many issues with TLS/SSL implementations

SSL-BASES VPNS

- We can use SSL/TLS to create secure site-to-site tunnels
  - Similarly, to IPSec
- A more flexible "user-space VPN"
  - In contrast to IPSec, it does not require kernel-level access
  - Virtual network interfaces are used instead
- Several solutions available:
  - e.g., OpenVPN, Cisco AnyConnect

WIREGUARD

IPSec

Is complex, hard to audit, and prone to misconfigurations
Big book of IPSec RFCs: Internet security architecture (Loshin, '99)
Does not prevent you from making bad choices
Supports all ciphers, including obsolete ones and NULL

SSL VPNs

Also, on the complex side
Tends to be slow
Also does not prevent you from making bad choices

WIREGUARD

New (and simpler) VPN design built from the ground-up

Offers a kernel and a user-space implementation

Faster than IPSec and TLS-based VPN solutions

WIREGUARD

- Easy to configure
  - But no PKI, keys are distributed manually
- Easy to audit
  - 4,000 LoCs vs IPSec's 400,000 LoCs
- Hard to get it wrong
  - Single cipher suite

RECAP OF CRYPTOGRAPHY USE CASES

Link Layer: WEP Problems

Short IV → two-time pad → make it bigger!
Checksum → integrity breach → use MACs
Protocol disaster → packet injection

Network layer: IPSec

IKE: Diffie-Hellman
Modes: Transport, Tunnel
Headers: AH, ESP

Transport TLS

Protocol summary (ECDHE, etc.)
Key management: CAs
Issues with TLS: MITM

Wireguard

Better VPN