Digital Signatures, Key Distribution

22 May 2025
Lecture 8

Topics for Today

- Digital Signatures
- Key Distribution

Physical Signatures

Consider a paper check used to transfer money from one person to another

Signature confirms authenticity

- Only legitimate signer can produce signature

In case of alleged forgery

- 3rd part can verify authenticity

Checks are cancelled so they can't be reused

Checks are not alterable

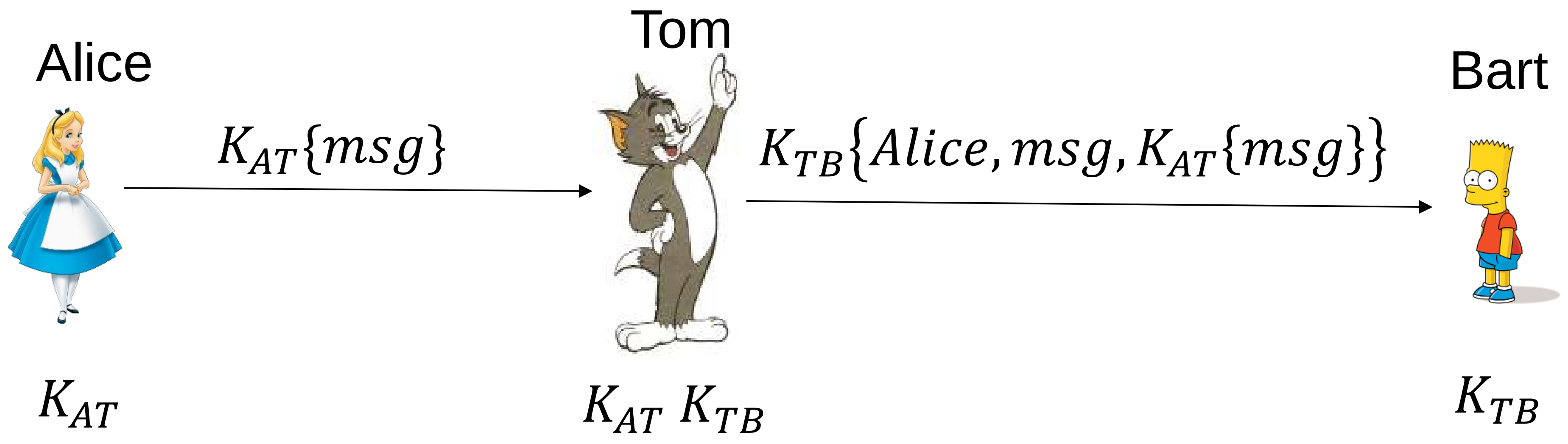
- Or alterations are easily detectable

Digital Signatures: Requirements

- **Digital Signature**: A mark that only one principal can make, but others can easily recognize
- Not forgeable
  - If P signs a message M with signature $S_P\{M\}$, no one else could produce the pair $(M, S_P\{M\})$.
- Authenticity
  - If R receives the pair $(M, S_P\{M\})$ (seemingly) from P, R can validate that the signature is really from P.
- Non-alterable
  - After being transmitted, $(M, S_P\{M\})$ cannot be changed by P, R, or an interceptor
- Not reusable
  - A duplicate message will be detected by the recipient.

- Non-repudiation
  - P deny a real signature
  - Related to unforgeable: If P can show the signature could have been forged, P can deny it (repudiate).

Digital Signatures with Shared Keys



- Tom is a trusted 3rd part (or arbiter)
- **Authenticity:** Tom verifies Alice’s message, Bart trusts Tom
- **No Forgery:** Bart can keep msg, $K_{AT}\{msg\}$ which only Alice (or Tom, but he’s trusted not to) could produce

Preventing Reuse and Alteration

To prevent reuse of the signature

- Incorporate a *timestamp* (or sequence number)

Alteration

- If a block cipher is used, recipient could splice-together new messages from individual blocks

To prevent alteration

- Timestamp must be part of each block
- Or … use *GCM* (ok, but fragile) or *HMAC* (2 steps)

Digital Signatures with Public Keys

Assume the algorithm is commutative

- $D_{Priv_K}(E_{Pub_K}\{M\}) = E_{Pub_k}(D_{Priv_k}\{M\})$

Alice's Keys:

- Pub_{K_A} public key
- $Priv_{k_A}$ private key

To sign msg, Alice sends $D_{Priv_{k_A}}\{msg\}$

Bart can verify the message with Alice's public key

- Works with RSA: $(m^e)^d = m^{ed} = \left(m^d\right)^e$

Digital Signatures with Public Keys

Alice

$$D_{Priv_{k_A}}\{msg\}$$

Bart

$Priv_{k_A}, Pub_{K_A},$
Pub_{K_B}

$Priv_{k_B}, Pub_{K_B},$
Pub_{K_A}

- No trusted 3rd party
- Simpler algorithm
- More expensive
- No confidentiality

Variations on Public Key Signatures

- Timestamps again (to prevent replay)
  - Signed certificates valid for only some time

- Add an extra layer of encryption to guarantee confidentiality
  - Alice sends $E_{Pub_{K_B}}\{D_{Priv_{k_A}}\{msg\}\}$ to Bart

- Combine with hashes
  - Send $(msg, D_{Priv_k}\{SHA256(msg)\})$

    Or

    $(msg, D_{Priv_k}\{SHA512(msg)\})$

This is what we do in practice

Adding Hashes

Alice

Bart

$$msg, D_{Priv_{k_A}}\{digest\}$$

$$\xrightarrow{\hspace{10em}}$$

$Priv_{k_A}, Pub_{K_A},$
Pub_{K_B}

- No message recovery
- Bart must calculate digest from msg received to validate

$Priv_{k_B}, Pub_{K_B},$
Pub_{K_A}

Thinking about Digital Signatures

- We have seen two uses of encryption so far:
  - Secrecy (encrypt/decrypt)
  - Authentication (digital signatures)

- The two have very different requirements
  - Strength of cipher
  - Lifetime
  - Storage
- It's normal to have separate encryption and signing key pairs
  - Why?

Enc Sig

- What risks are associated with digital signatures that are not present in secret communication?
  - The other way around?

So Far

- Digital Signatures
- Key Distribution

Key Establishment

Establish a "session key"

A shared key used for encrypting communications for a short duration – a session

Must authenticate first

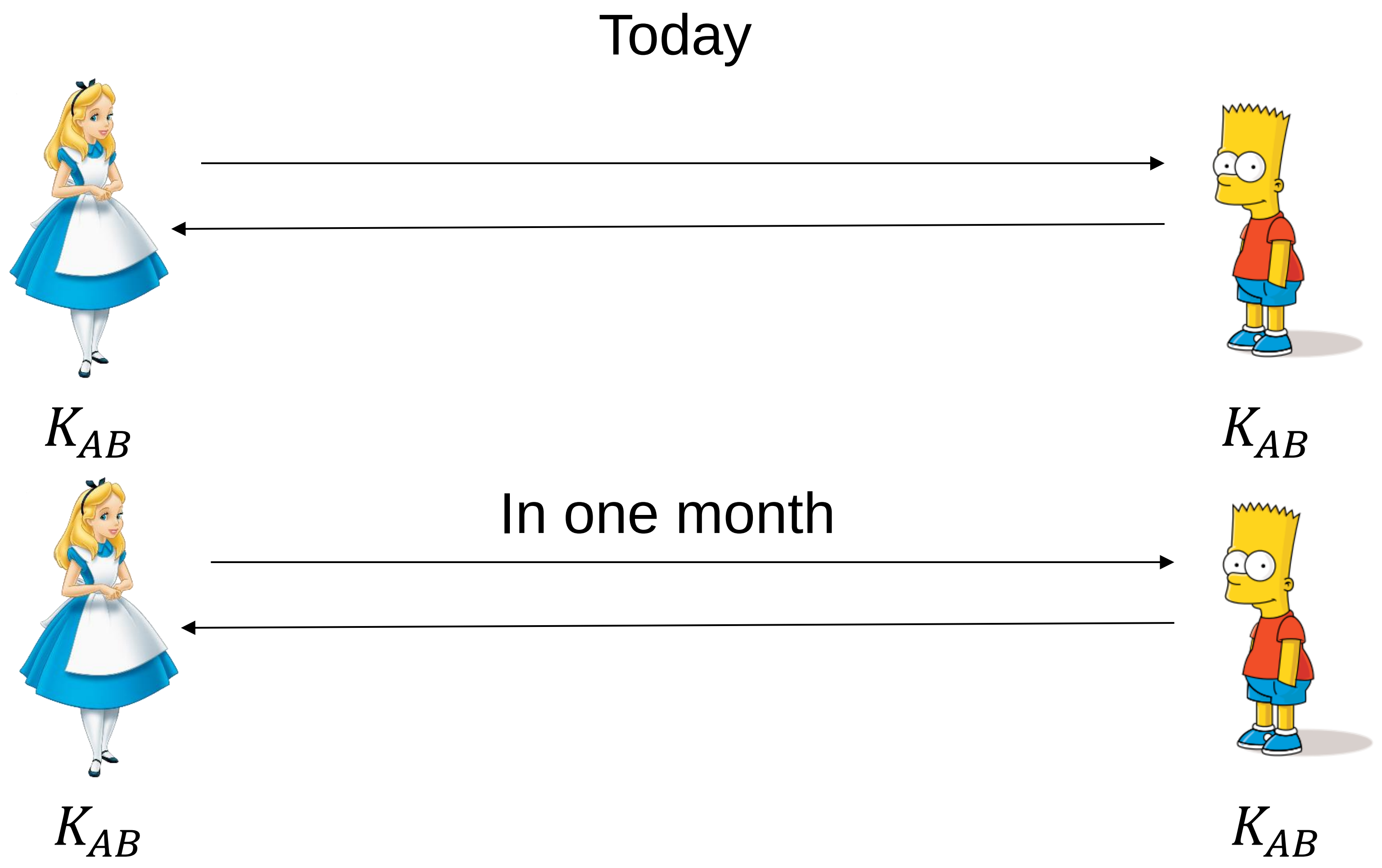
New session, new key

Symmetric Key Mechanisms

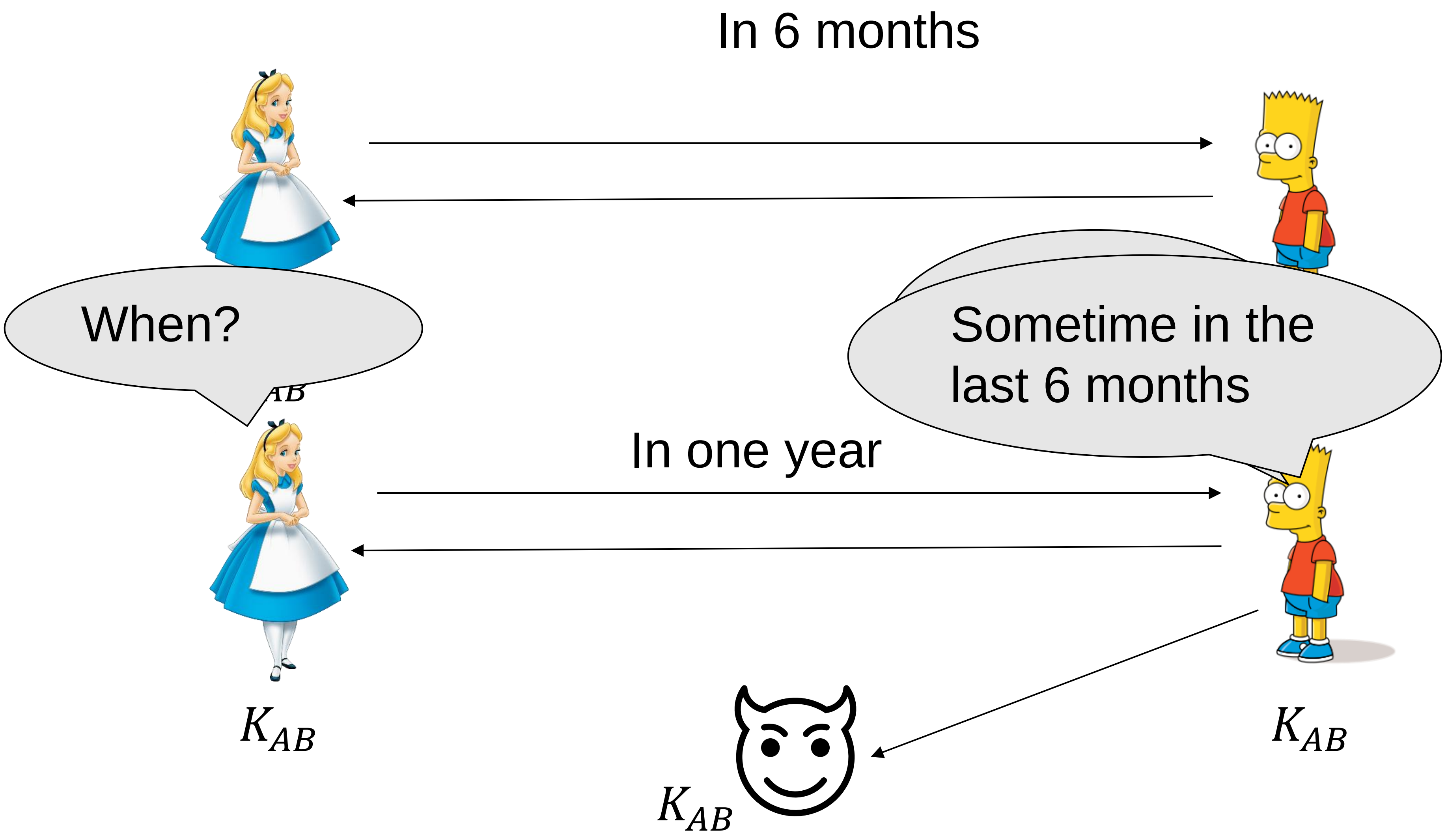
1. Point-to-Point
2. Needham-Schroeder
3. Kerberos

Image source: By Wilhelm Janson (Holland, Amsterdam), Antonio Tempesta (Italy, Florence, 1555-1630) – Image: http://collections.lacma.org/sites/default/files/remote_images/piction/ma-31724610-O3.jpg Gallery: http://collections.lacma.org/node/234467 archive copy, Public Domain, https://commons.wikimedia.org/w/index.php?curid=27299872

Why session keys



Why session keys



Symmetric Keys

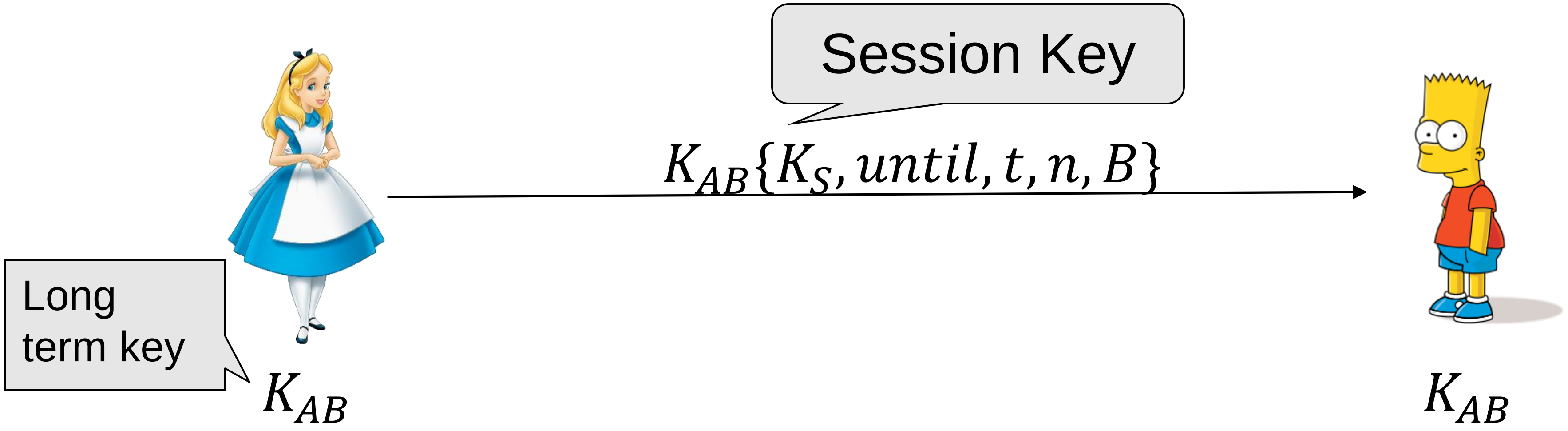
Set Up

- Key establishment using only symmetric keys requires pre-distributed keys to get things going
  - How?
- After that, can bootstrap from them

Protocols

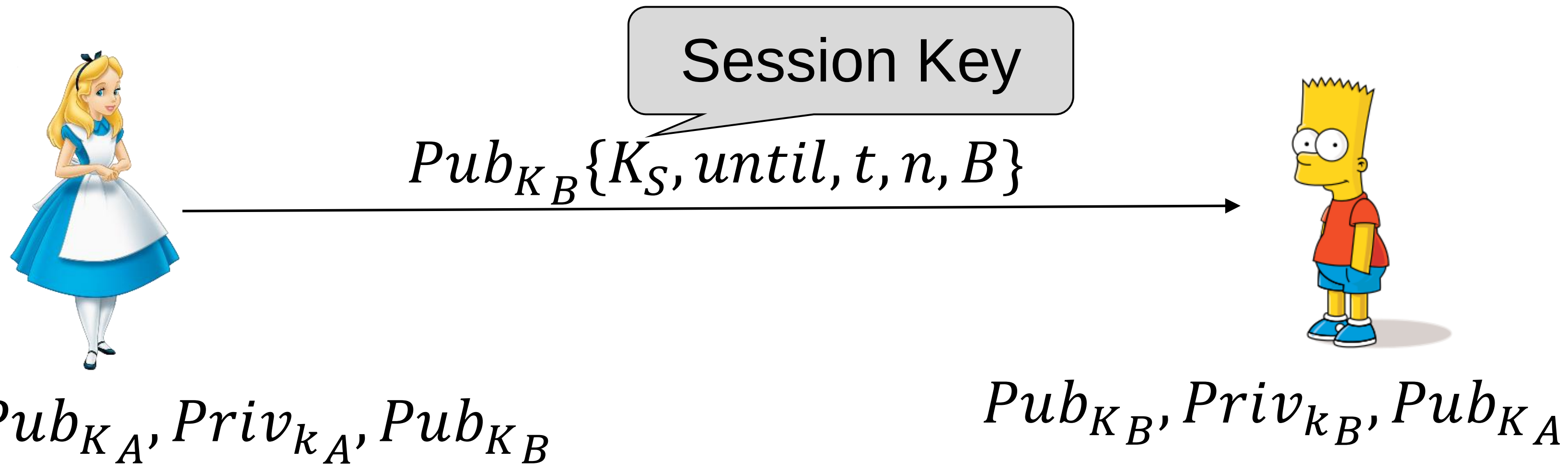
- Protocol can be based on
  1. Point to point distribution
  2. Key Distribution Center (KDC)

Point-to-Point with Symmetric



- When sending over network, use Key Wrap algorithm to encrypt the key (ex. *AES Key Wrap*)
- Should also use timestamps (t) and nonce (n).
- Session key should include a validity period ($until$)
- Should write the target as well (B)

Point-to-Point with PKE


Session Key
$Pub_{K_B}\{K_S, until, t, n, B\}$
$Pub_{K_A}, Priv_{k_A}, Pub_{K_B}$
$Pub_{K_B}, Priv_{k_B}, Pub_{K_A}$

- Can use public key encryption too
  - Use OAE or Cipher Key Encapsulation algorithm
  - Authentication of recipient
- Can add a digital signature on ciphertext to authenticate Alice.

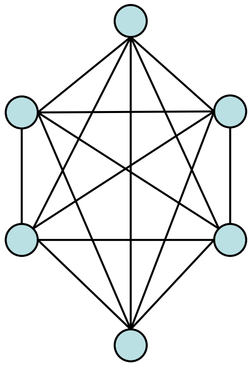
Goal: Distributing Keys

Using symmetric keys:

- Pair-wise distribution needs $\frac{n(n-1)}{2}$ keys total
- Backing them up?
- Maintaining them?

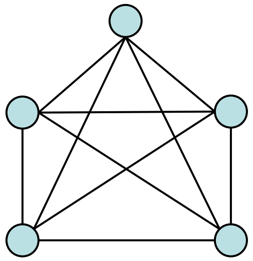
Better idea: Centralized key management

- Everyone has a shared key with the server
- The server helps set up communications

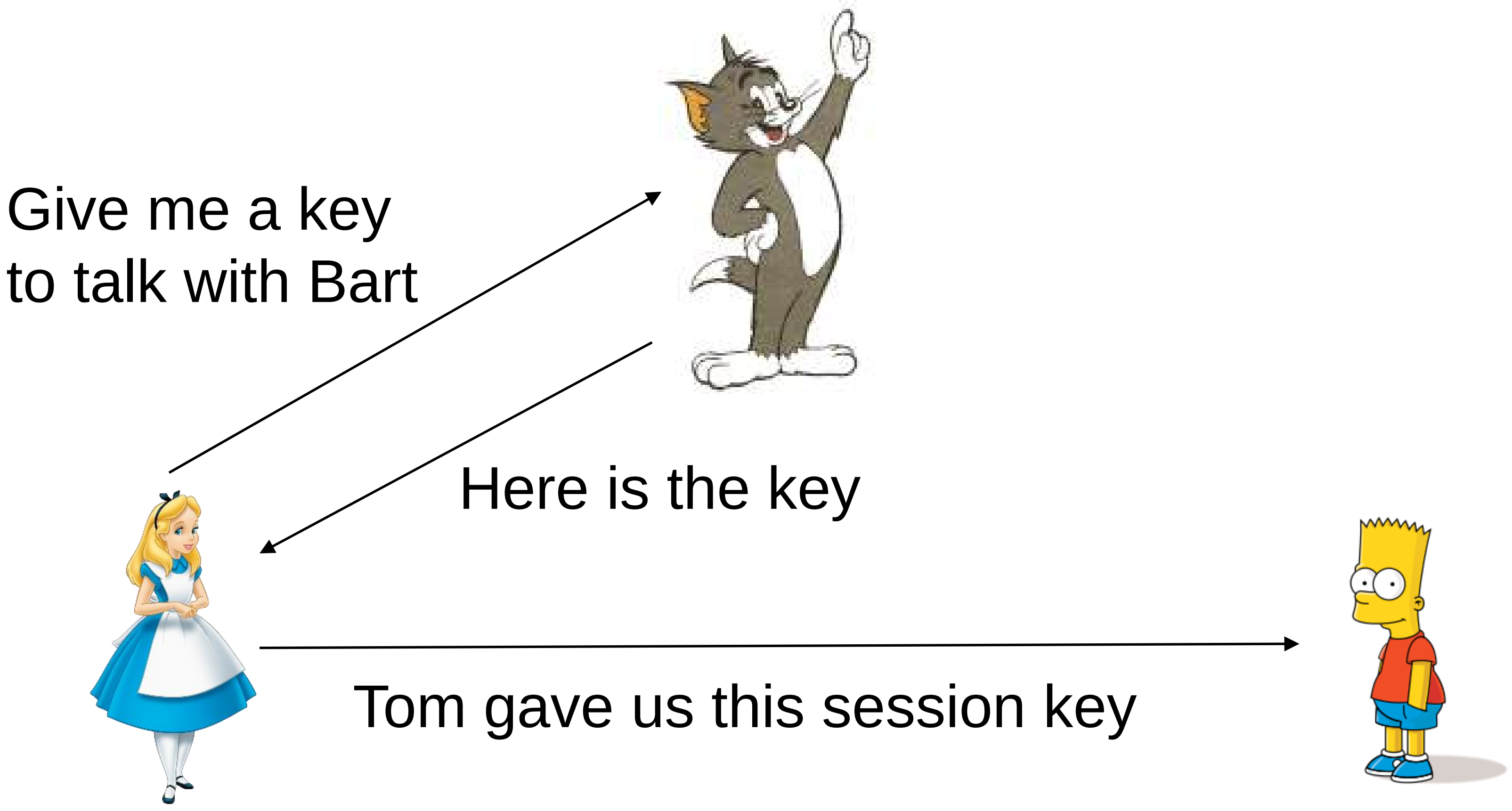


Alternatives:

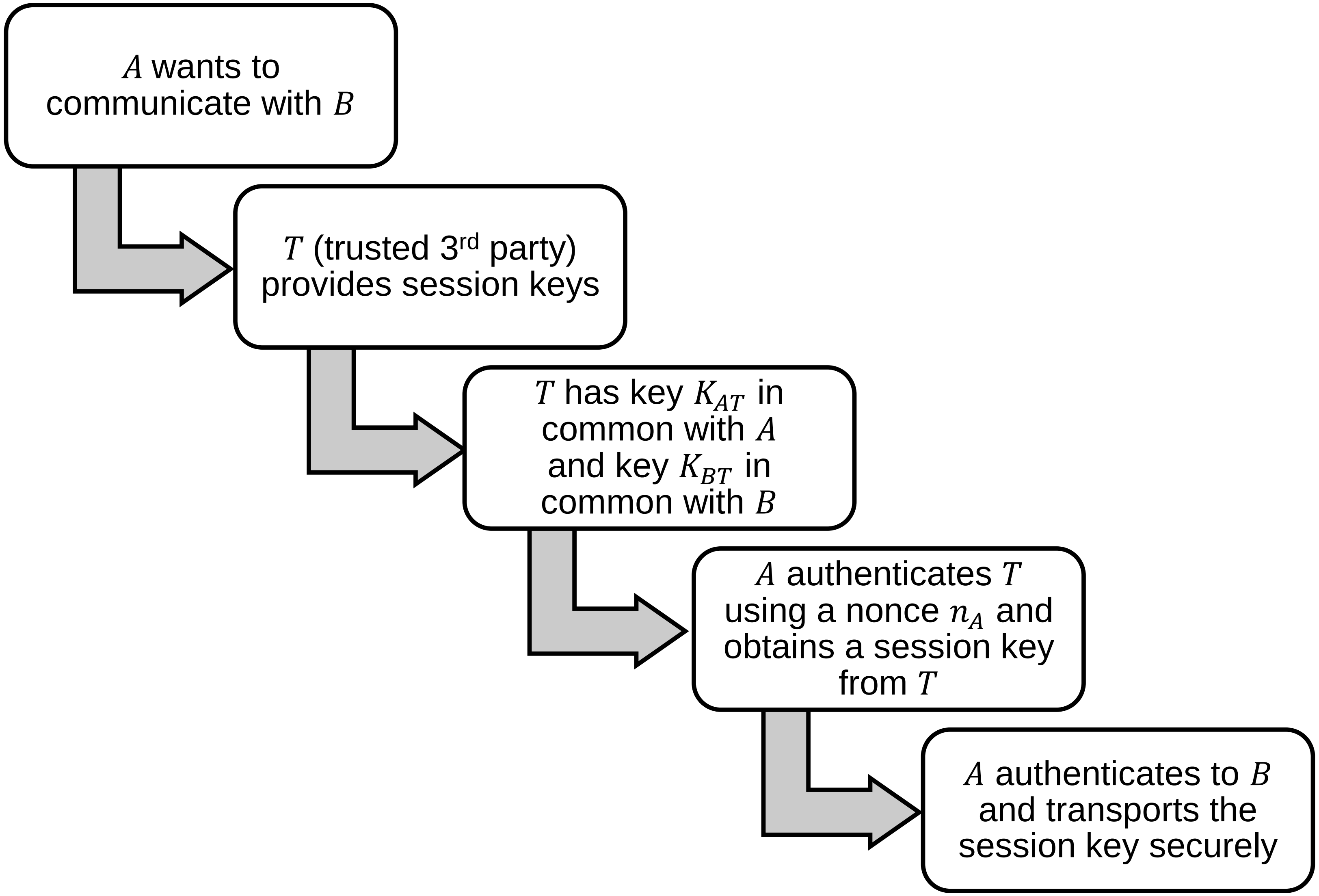
- Key Distribution Center
- Key Translation Center



Key Distribution Centers



Distribution Center Setup


A wants to communicate with B
T (trusted 3rd party) provides session keys
T has key K_{AT} in common with A and key K_{BT} in common with B
A authenticates T using a nonce n_A and obtains a session key from T
A authenticates to B and transports the session key securely

Needham-Schroeder Protocol

1. $A \to T$: A, B, n_A
2. $T \to A$: $K_{AT}\{K_S, n_A, B, K_{BT}\{K_S, A\}\}$

   A decrypts K_{AT} and checks n_A and B. Holds K_S for future correspondence with B.
3. $A \to B$: $K_{BT}\{K_S, A\}$

   B decrypts with K_{BT}
4. $B \to A$: $K_S\{n_B\}$

   A decrypts with K_S
5. $A \to B$: $K_S\{n_B - 1\}$

   B checks $n_B - 1$

There's a key reuse attack here – need to change 4 to fix it.

The inventors

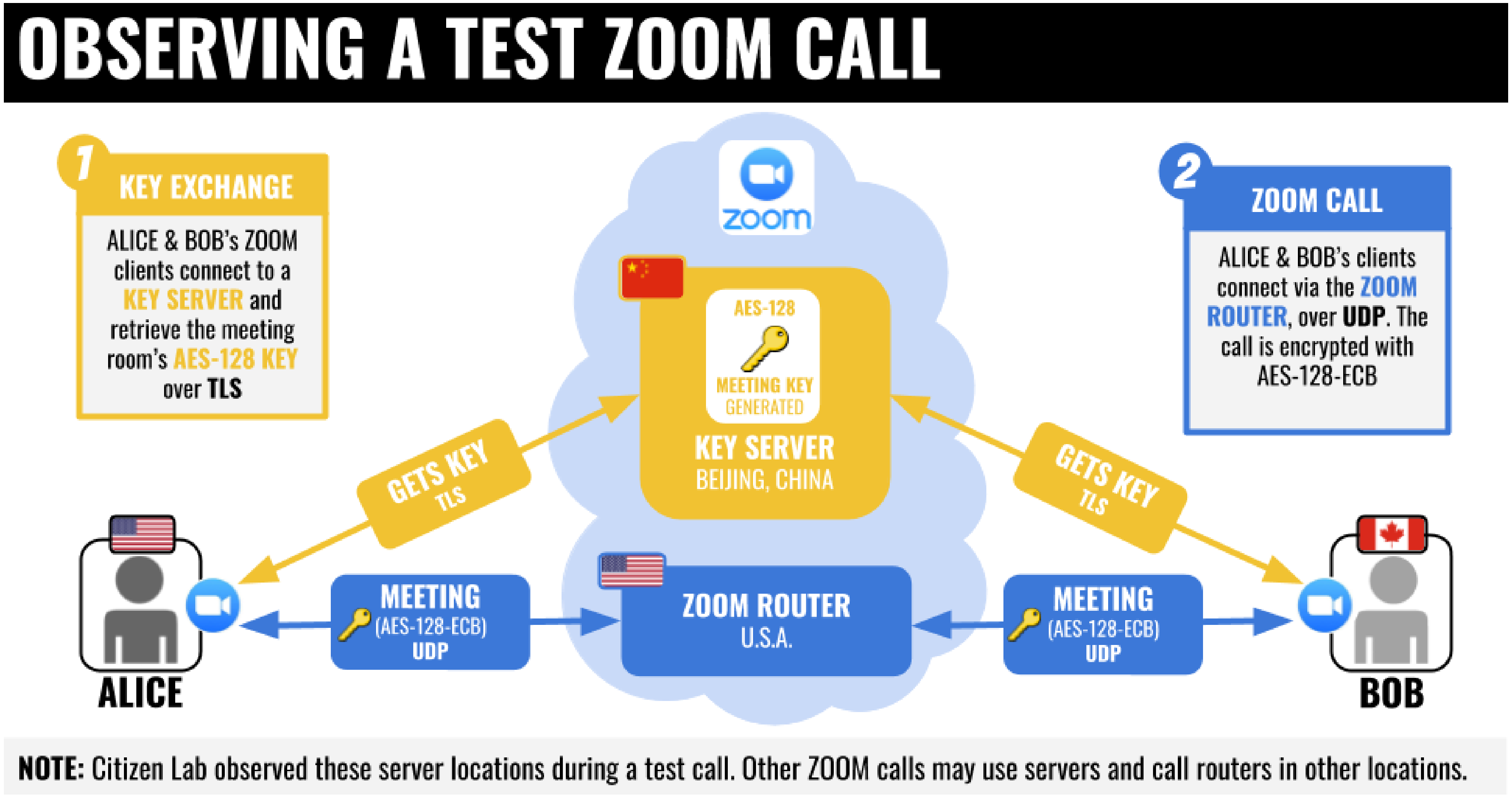
Roger Needham

Source: By Source, Fair use, https://en.wikipedia.org/w/index.php?curid=193399

Michael D. Schroeder

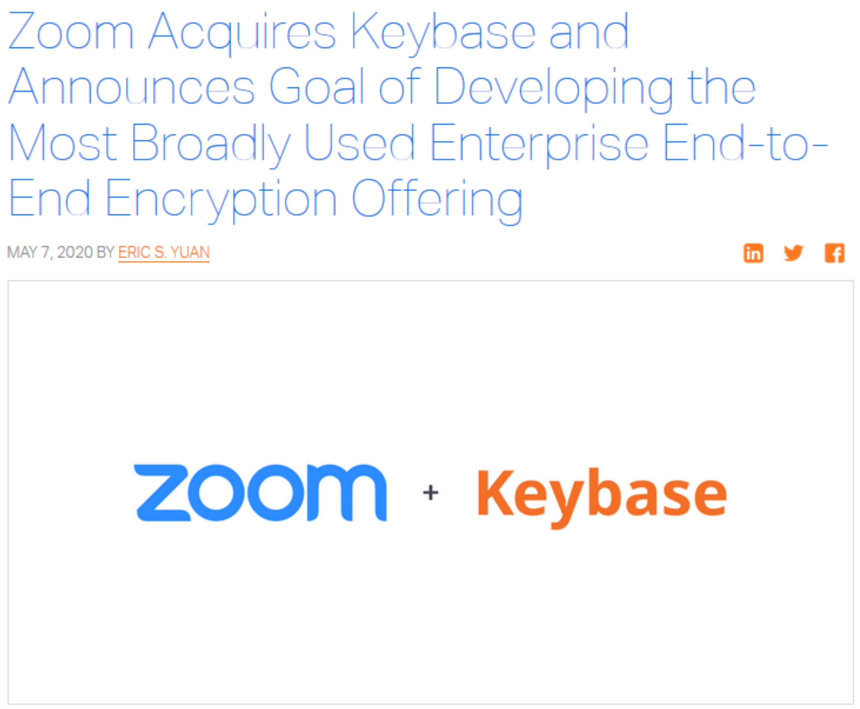
Source: https://www.d.umn.edu/tma/MungerSite/mike.jpg

It matters where Tom is


OBSERVING A TEST ZOOM CALL
1
KEY EXCHANGE
ALICE & BOB's ZOOM clients connect to a KEY SERVER and retrieve the meeting room's AES-128 KEY over TLS
2
ZOOM CALL
ALICE & BOB's clients connect via the ZOOM ROUTER, over UDP. The call is encrypted with AES-128-ECB
zoom
AES-128
MEETING KEY GENERATED
KEY SERVER
BEIJING, CHINA
GETS KEY
TLS
GETS KEY
TLS
MEETING
(AES-128-ECB)
UDP
ZOOM ROUTER
U.S.A.
MEETING
(AES-128-ECB)
UDP
ALICE
BOB
NOTE: Citizen Lab observed these server locations during a test call. Other ZOOM calls may use servers and call routers in other locations.

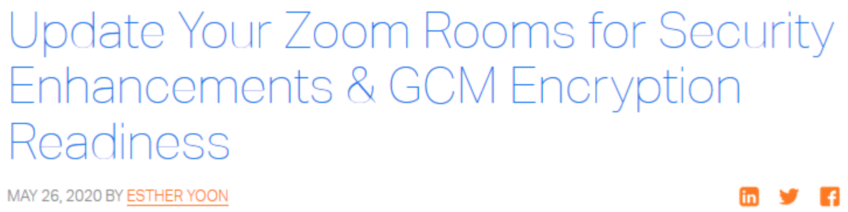
Source: https://citizenlab.ca/2020/04/move-fast-roll-your-own-crypto-a-quick-look-at-the-confidentiality-of-zoom-meetings/

Let's fix that



Zoom Acquires Keybase and Announces Goal of Developing the Most Broadly Used Enterprise End-to-End Encryption Offering

MAY 7, 2020 BY ERIC S. YUAN

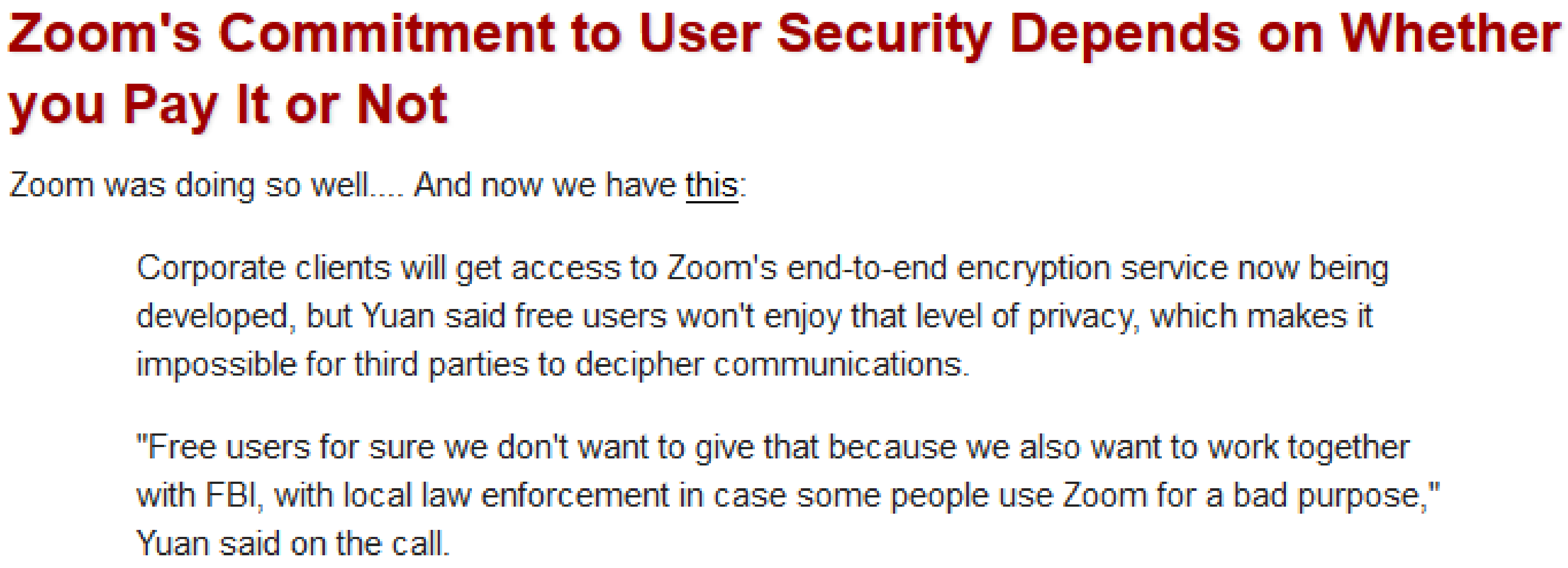


Update Your Zoom Rooms for Security Enhancements & GCM Encryption Readiness

MAY 26, 2020 BY ESTHER YOON

Sources: https://blog.zoom.us/wordpress/2020/05/26/update-your-zoom-rooms-security-enhancements-gcm-encryption-readiness/, https://blog.zoom.us/wordpress/2020/05/07/zoom-acquires-keybase-and-announces-goal-of-developing-the-most-broadly-used-enterprise-end-to-end-encryption-offering/

But not for everyone



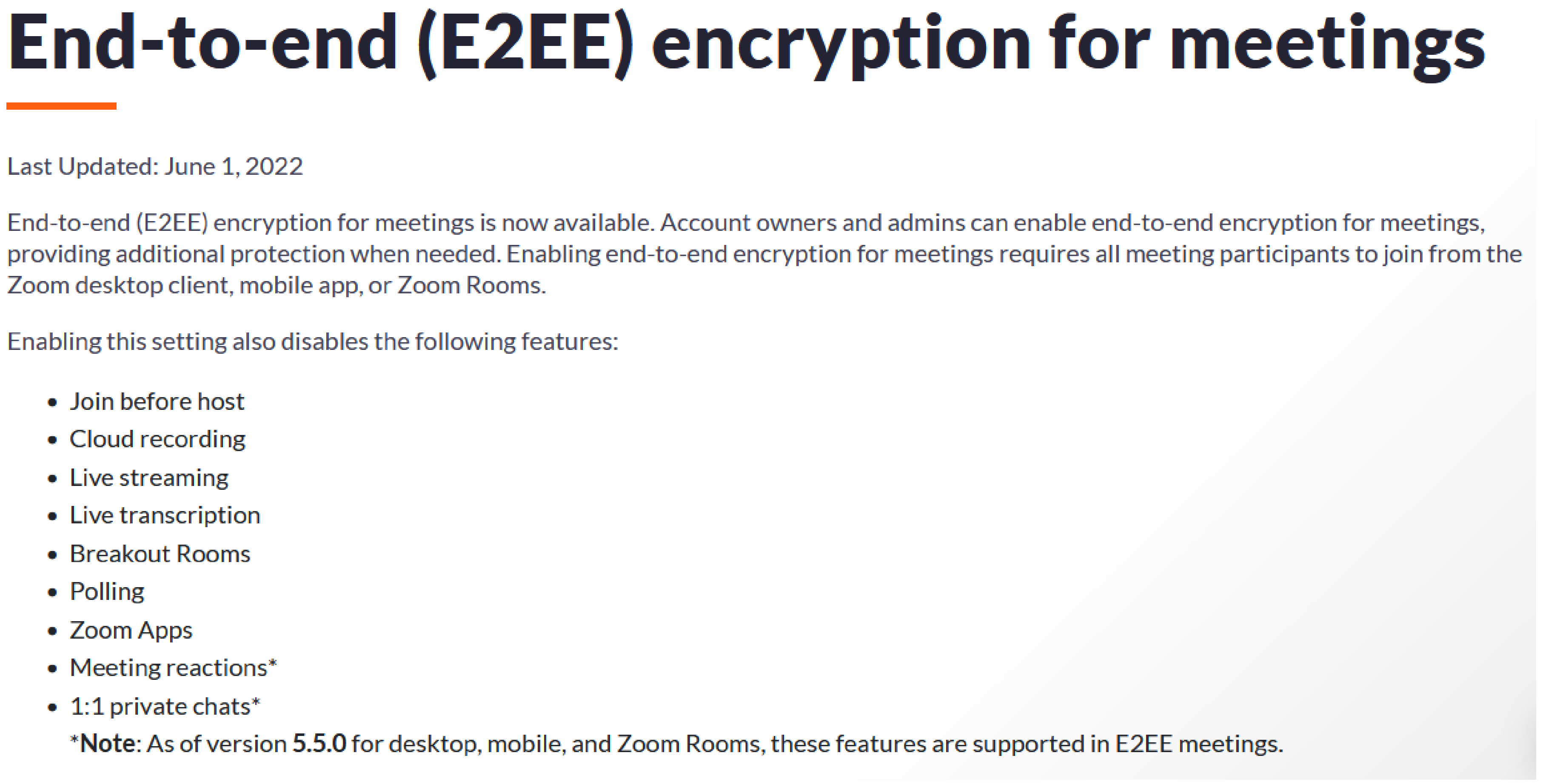
Zoom's Commitment to User Security Depends on Whether you Pay It or Not

Zoom was doing so well.... And now we have this:

> Corporate clients will get access to Zoom's end-to-end encryption service now being developed, but Yuan said free users won't enjoy that level of privacy, which makes it impossible for third parties to decipher communications.
>
> "Free users for sure we don't want to give that because we also want to work together with FBI, with local law enforcement in case some people use Zoom for a bad purpose," Yuan said on the call.

Source: https://www.schneier.com/blog/archives/2020/06/zooms_commitmen.html
https://www.bloomberg.com/news/articles/2020-06-02/zoom-transforms-hype-into-huge-jump-in-sales-customers

And not with many features



End-to-end (E2EE) encryption for meetings

Last Updated: June 1, 2022

End-to-end (E2EE) encryption for meetings is now available. Account owners and admins can enable end-to-end encryption for meetings, providing additional protection when needed. Enabling end-to-end encryption for meetings requires all meeting participants to join from the Zoom desktop client, mobile app, or Zoom Rooms.

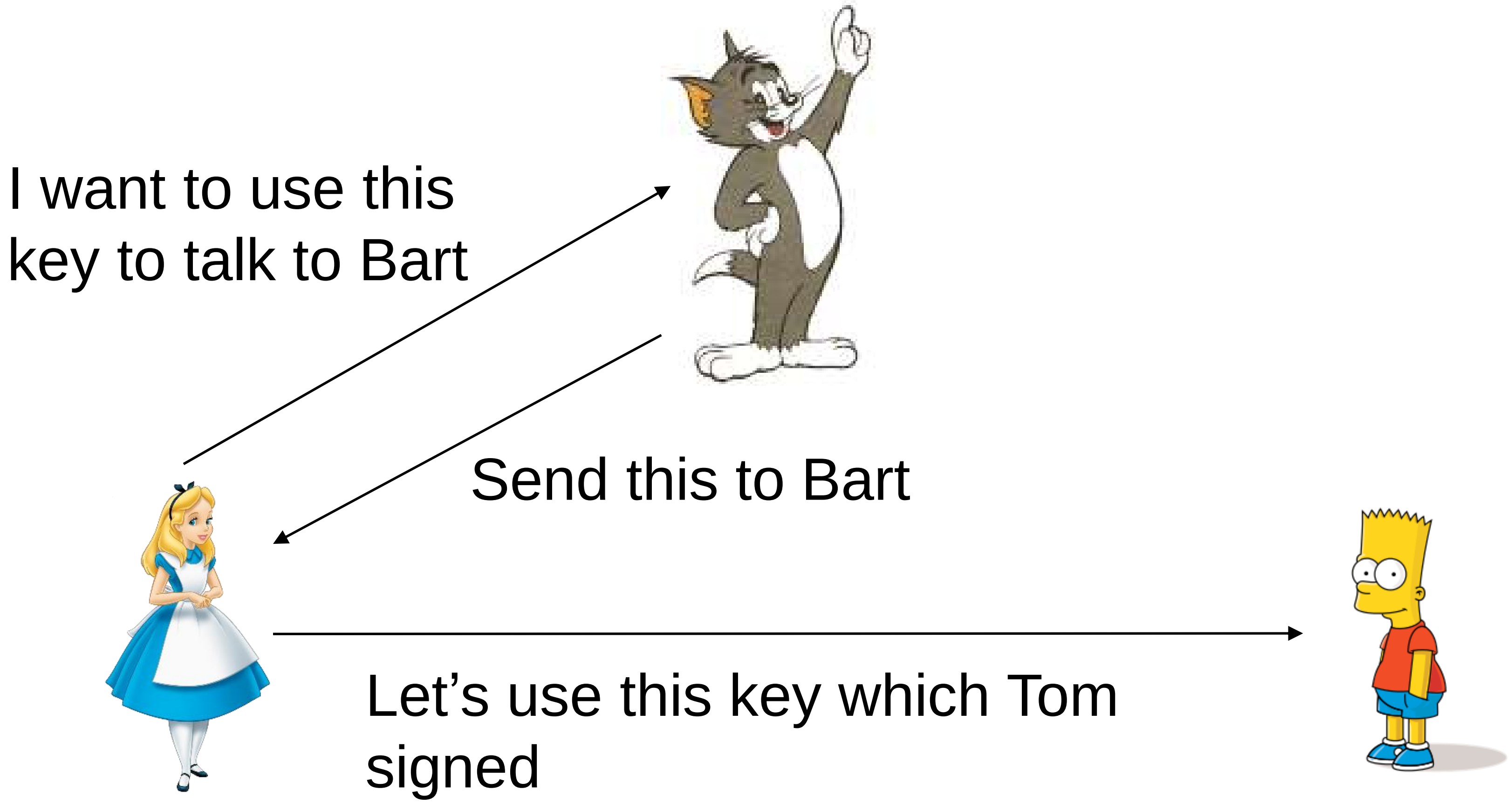
Enabling this setting also disables the following features:

- Join before host
- Cloud recording
- Live streaming
- Live transcription
- Breakout Rooms
- Polling
- Zoom Apps
- Meeting reactions*
- 1:1 private chats*

  ***Note**: As of version **5.5.0** for desktop, mobile, and Zoom Rooms, these features are supported in E2EE meetings.

https://support.zoom.us/hc/en-us/articles/360048660871-End-to-end-E2EE-encryption-for-meetings

Key Translation Centers


I want to use this key to talk to Bart
Send this to Bart
Let’s use this key which Tom signed

Key Translation Center

- Similar to Key Distribution Center except that Alice decides the session key to use
- Alice and Tom share a key
  - Tom authenticates to Alice
- Bart and Tom share a key
  - Alice authenticates to Bart
- Alice chooses a session key which Tom encrypts for Bart
  - Bart opens it up and forces Alice to prove she knows the key

Conclusion

- Digital Signatures
- Key Distribution