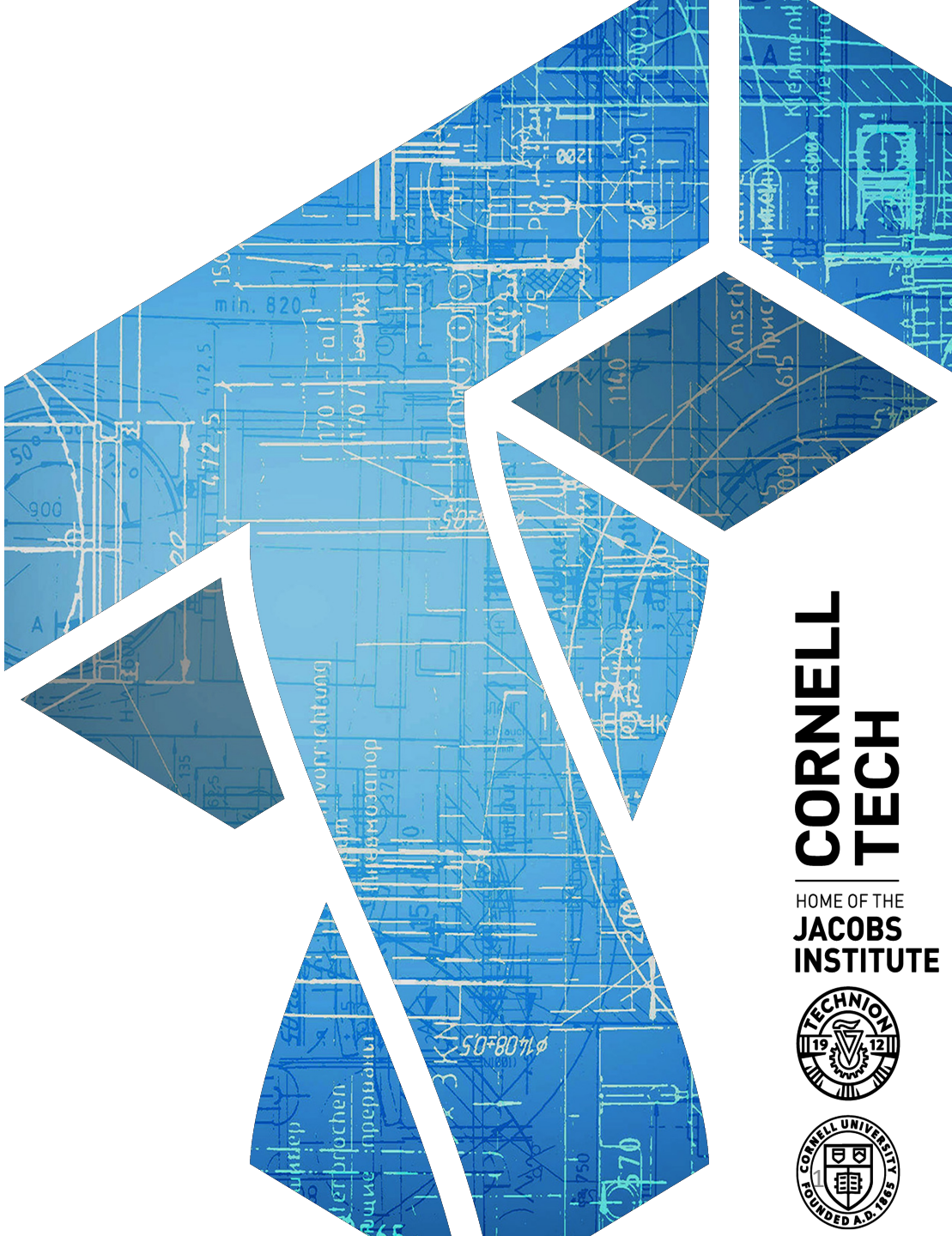
CS 5830
Cryptography

Instructor: Tom Ristenpart
TAs: Yan Ji, Sanketh Menda



CORNELL TECH
HOME OF THE JACOBS INSTITUTE

Crypto primitives we've seen

Symmetric primitives:

- Authenticated encryption (with associated data)
- Message authentication (rarely used standalone)
- Hash functions

Asymmetric primitives:

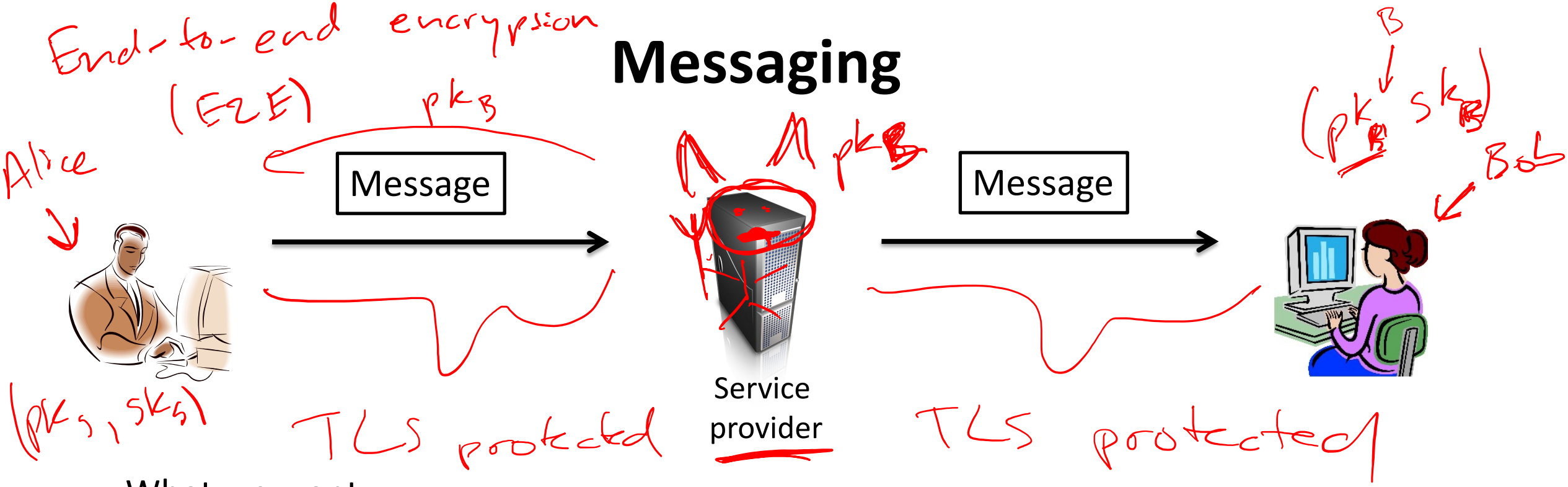
- Key exchange
- Public-key encryption
- Digital signatures

Application-layer crypto

- So far focused on TLS as running example
  - Transport Layer Security
  - Provides network socket style stream interface

- What about if an application wants to encrypt discrete messages (as opposed to stream)?
  - Email
  - Text messages
  - Etc.

Messaging



What we want:

- Message may be large (body of email, PDF, image, etc.)
- Ideally: asynchronous communications (both parties need not be online)
- Desire confidentiality and authenticity (including replay resistance)

How would we design this given primitives we know?

Public-key encryption (RSA or Diffie-Hellman), digital signatures, symmetric encryption

ElGamal public-key encryption

g is generator for group of order p

Kg outputs pk = $(g, X = g^x)$ and sk = (g, x)

Enc((g,X), M, R)

r <-$ $\mathbf{Z}_p$

$C1 = g^r$

$C2 = X^r * M$

Return C1 , C2

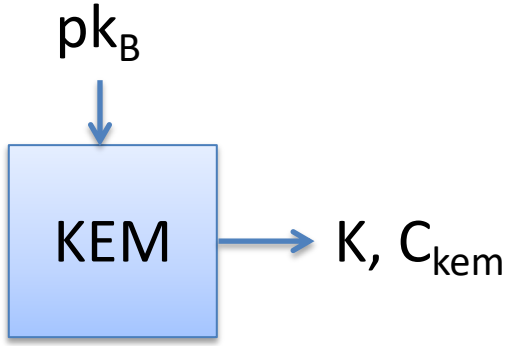
Dec((g,x), C1, C2):

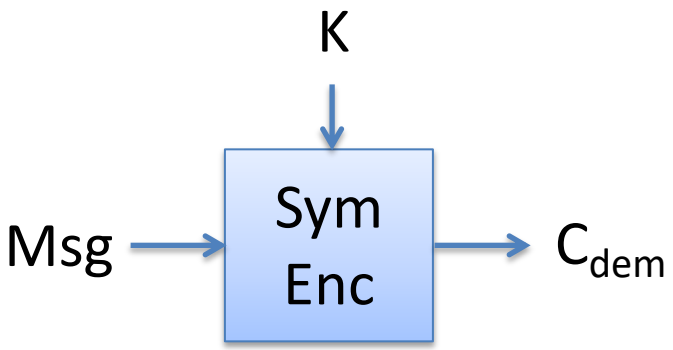
Return $C2 * C1^{-x}$

This is only at most chosen-plaintext attack secure. CCA attacks?

Only encrypts messages of size up to about log p bits

Hybrid encryption (KEM/DEM)





KEM = key encapsulation mechanism
Randomized public-key primitive

DEM = data encapsulation mechanism
One-time secure authenticated encryption

HybEnc(pk, M)
$K, C_{kem} \leftarrow\$ KEM(pk)$
$C_{dem} \leftarrow Enc(K,M)$
Return C_{kem} , C_{dem}

HybDec(sk, C_{kem} , C_{dem})
$K \leftarrow KEM^{-1}(sk, C_{kem})$
$M \leftarrow Dec(K, C_{dem})$
Return M

ElGamal KEM

Kg outputs pk = (g,X =g^x) and sk = (g,x)
g is generator for group of order prime p

Ephemeral DH

EG-KEM((g,X)):
r <-$ $\mathbf{Z}_p$
C_{kem} <- g^r
Return $H(X^r)$, C_{kem}

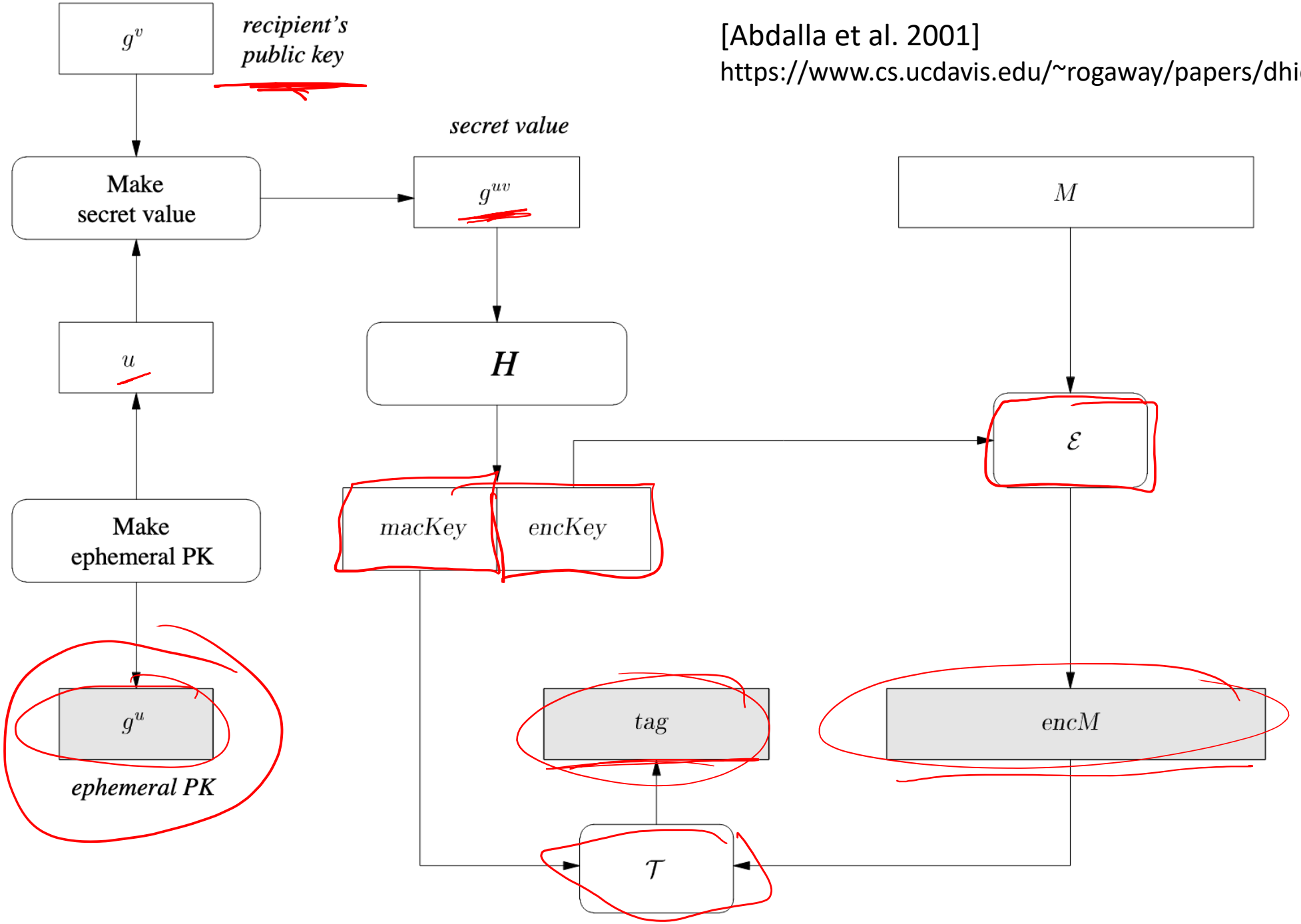
Dec((g,x), C_{kem}):

Return $H(C_{kem}^x)$

Secure if computational Diffie-Hellman assumption holds in group

DHIES (Diffie-Hellman Integrated Encryption Scheme) is an example of public-key encryption using this type of KEM

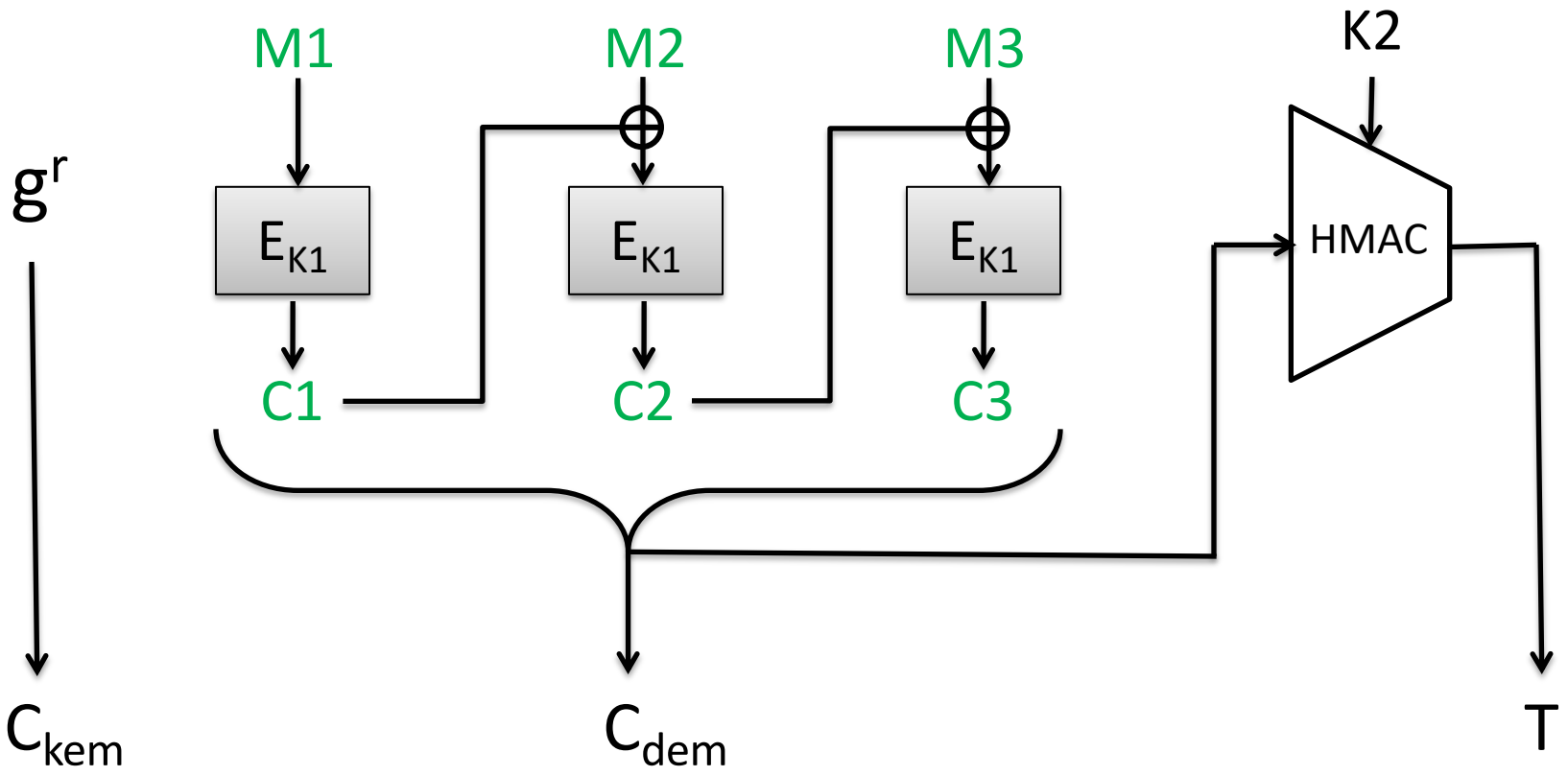
[Abdalla et al. 2001]
https://www.cs.ucdavis.edu/~rogaway/papers/dhies.pdf



DHIES with CBC and HMAC

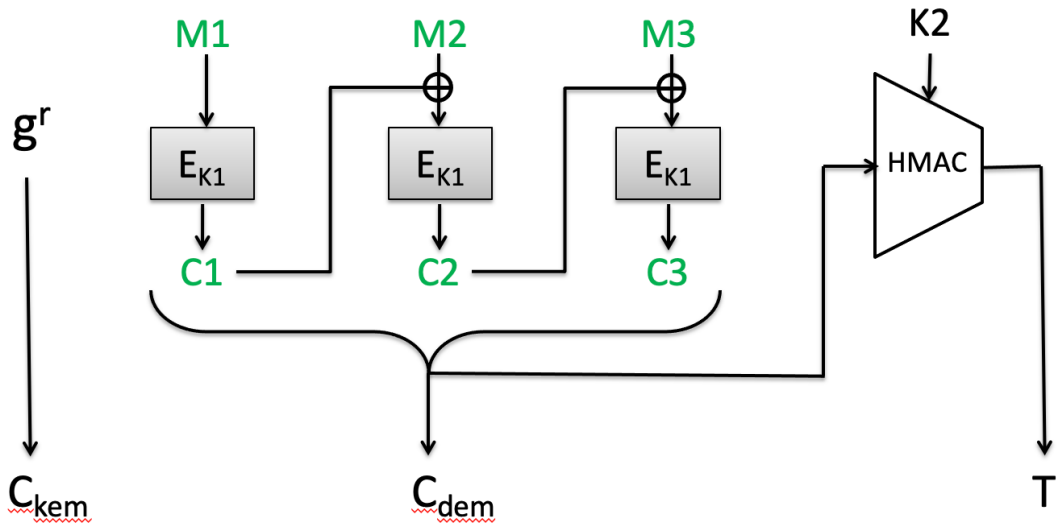
Recipient public key is g^x

K1 || K2 = SHA256(g^{xr})



DHIES security intuition

- Confidentiality for passive adversaries (IND-CPA)?
  - Symmetric encryption prevents

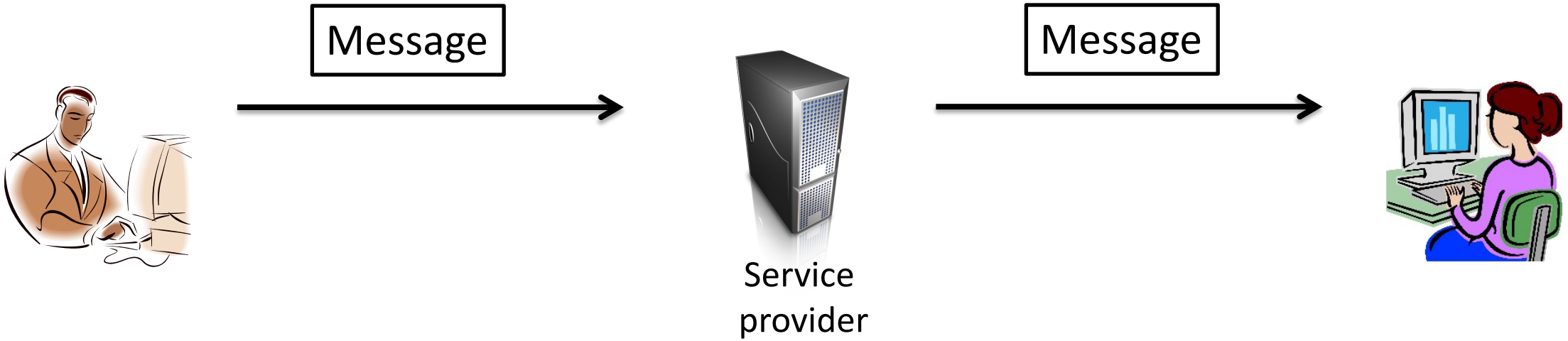


- Chosen-ciphertext attack security?
  - Change g^r => K1||K2 will be different
  - Change C_{dem} or T => tag check fails

- Can formally show security implied by:
  - IND-CPA security of symmetric encryption, message authenticity of tag, and (hashed) Diffie-Hellman assumption:

    $(g^r, g^x, H(g^{rx}))$ indistinguishable from (g^r, g^x, U) (U uniform random bits)

Messaging

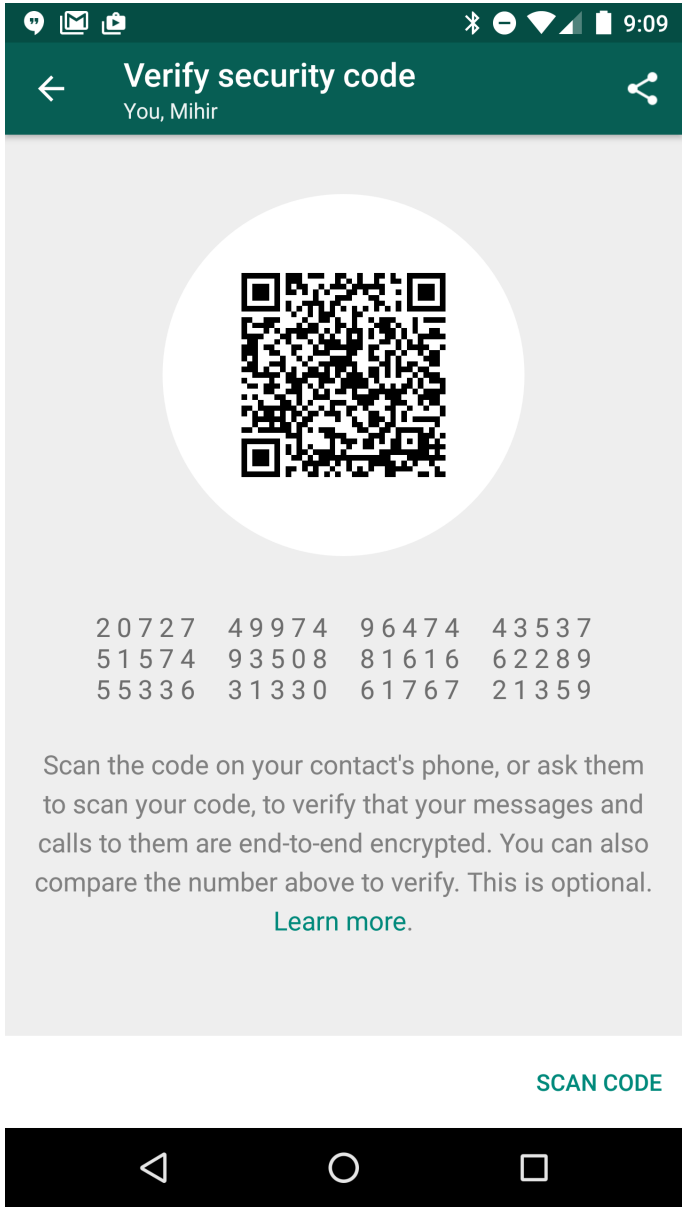


Want to use DHIES to encrypt message

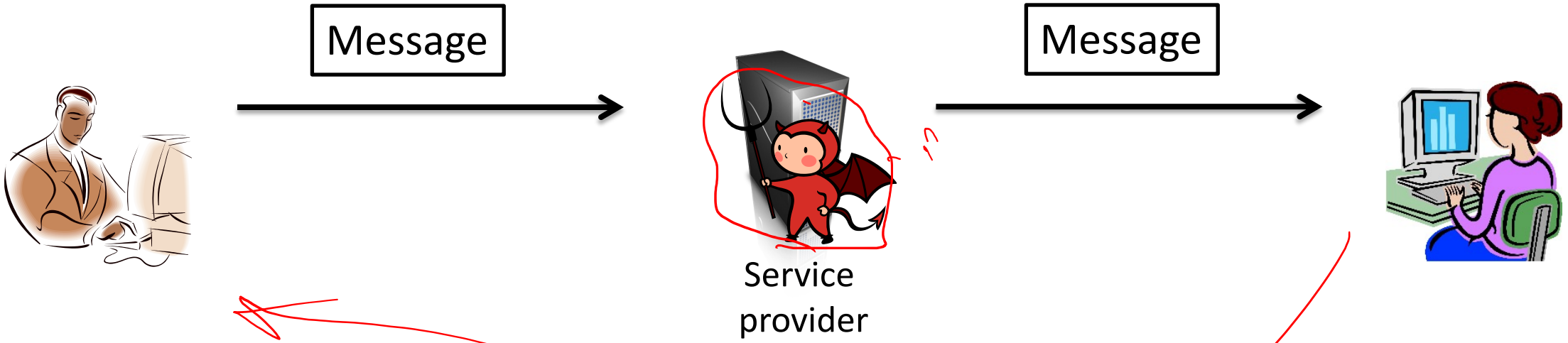
How do we get recipient's public key?

How do we trust recipient's public key?

Verifying public keys



Messaging



Say we verify recipient public key

What attacks does using just DHIES still allow?

No authenticity! Service provider can inject arbitrary messages
No repeat protection! Service provider can replay ciphertexts

Need digital signatures and some replay resistance mechanisms

PGP history

- Phil Zimmerman released “Pretty Good Privacy” in 1991 on a USENET post marked as “US only”

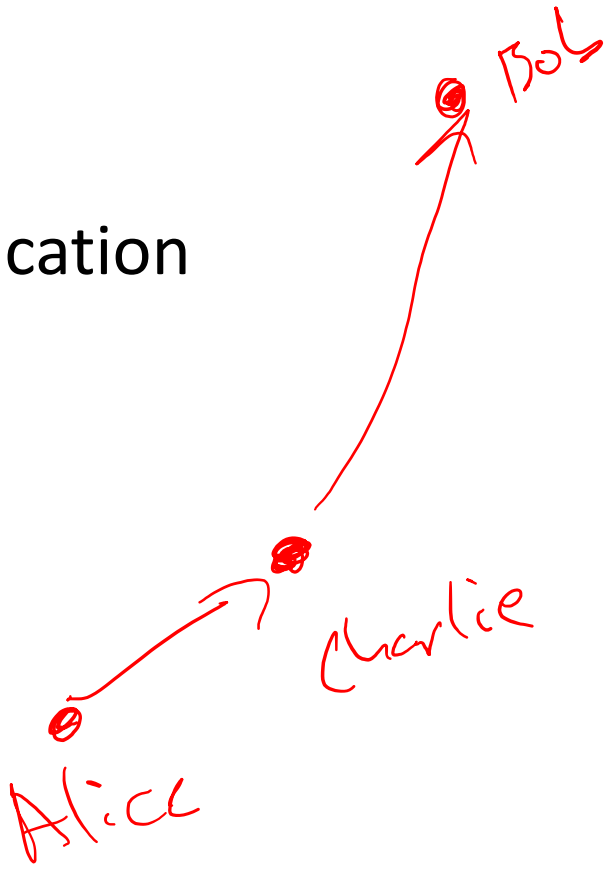
- 1993: Criminal investigation by US government for munitions export without a license.
  - Printed PGP source code into a book. First amendment gambit

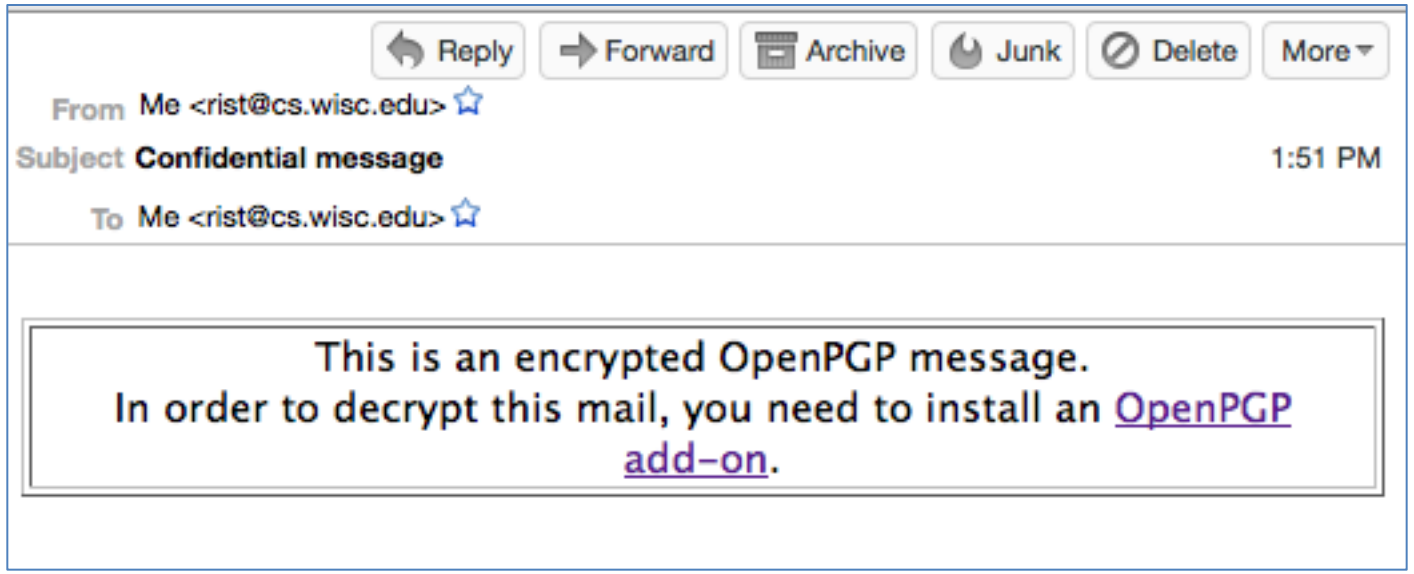
OpenPGP overview

- Standard for PGP is RFC 4880
- Key encapsulation mechanism:
  - RSA PKCS#1 v1.5 encryption
  - ElGamal over finite field or elliptic curve
- Digital signatures:
  - RSA PKCS#1 v1.5 signatures
  - DSA
- Symmetric encryption:
  - Password-based key derivations using iterated hashing (for protection of asymmetric secret keys, or for allowing ciphertext to be decrypted via passphrase)
  - CFB mode using block cipher (variant of CBC mode)

OpenPGP overview

- Security problems:
  - Padding oracle attacks against CFB & PKCS#1 v1.5
  - Attacks against home-brewed integrity checks (modification detection check, MDC)
  - Subject lines always in the clear
- Usability problems:
  - Users must manage their own keys
  - Copying private keys to each device
  - Checking validity of other recipient's public key





The Switch

Yahoo's plan to get Mail users to encrypt their e-mail: Make it simple

A | 14 | Save for Later | Reading List

End-to-end (E2E) Encrypted messaging

- TextSecure by Open Whisper Systems (2013)
  - Trevor Perrin and Moxie Marlinspike (Levchin award winners)
  - Later became Signal
  - Developed Signal Protocol, which was adopted elsewhere (most notably: WhatsApp)
- Protocol provides:
  - "… confidentiality, integrity, authentication, participant consistency, destination validation, forward secrecy, post-compromise security (aka future secrecy), causality preservation, message unlinkability, message repudiation, participation repudiation, and asynchronicity."

Signal Protocol (X3DH)

https://signal.org/docs/specifications/x3dh/

Identity key

Signed pre-key

Signature of SPK_B using secret key associated to IK_B

One-time pre-keys

Service provider

IK_B, SPK_B, σ , $OPK1_B$, … , $OPKn_B$

IK_B, SPK_B, $OPKi_B$ all public DH keys, recipient retains secret keys

IK_B, SPK_B, σ , $OPKi_B$

Verify σ

$r \leftarrow\$ \mathbf{Z}_p$; $EK_A \leftarrow g^r$

DH(PK1,PK2) is means output $g^{sk1*sk2}$

$DH1 = DH(IK_A, SPK_B)$

$DH2 = DH(EK_A, IK_B)$

Mutual authentication

$DH3 = DH(EK_A, SPK_B)$

$DH4 = DH(EK_A, OPK_B)$

Forward secrecy

$SK = KDF(DH1 \,||\, DH2 \,||\, DH3 \,||\, DH4)$

$C \leftarrow AEAD(SK, message)$

IK_A , EK_A, key identifiers, C

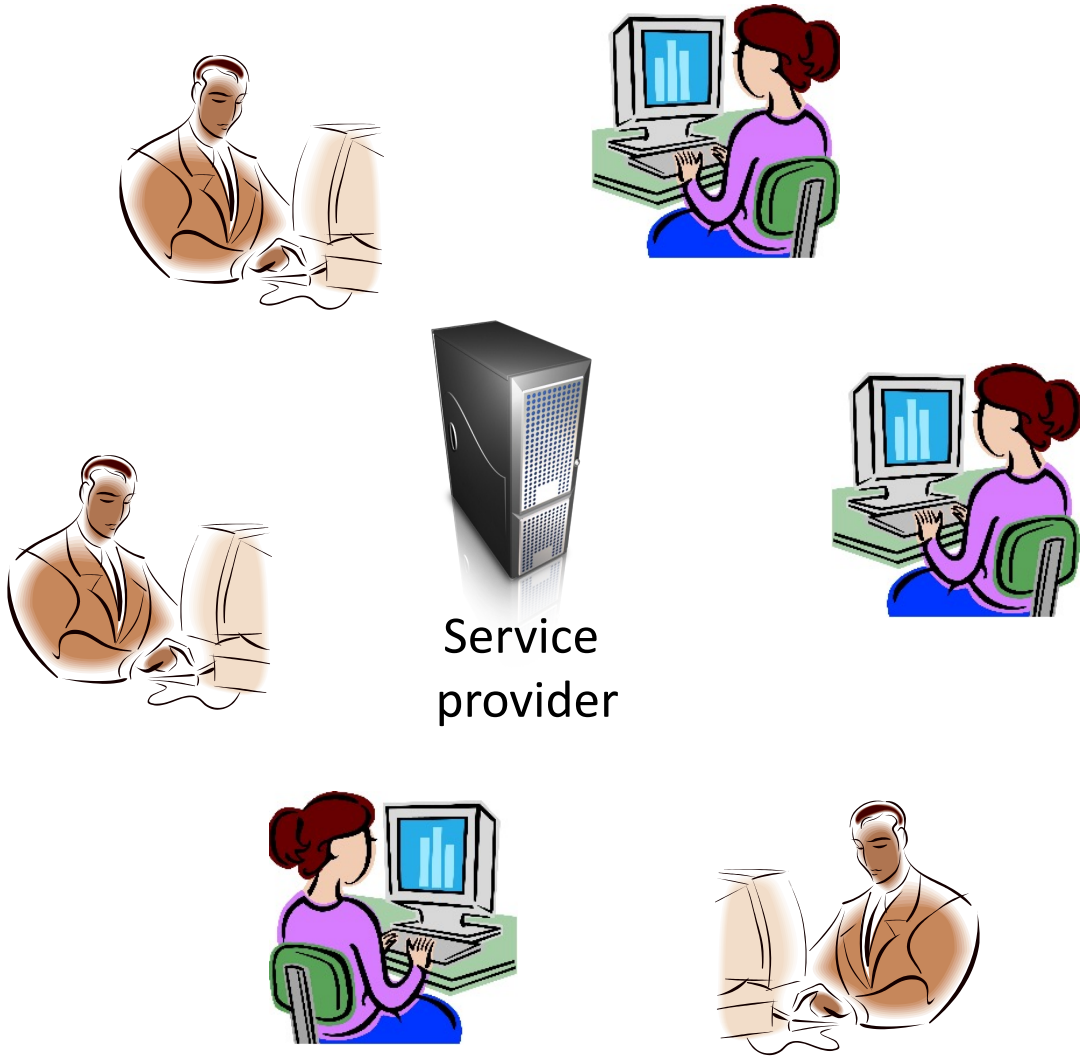
IK_A , EK_A, key identifiers, C

Signal Double Ratchet

https://signal.org/docs/specifications/doubleratchet/

- Use X3DH to establish shared secret SK
- To improve forward secrecy, shouldn’t use SK forever
- Ratcheting refers to deriving new key material to allow deleting old key material
- Asymmetric ratchet:
  - New ephemeral DH values sent along with messages
- Symmetric ratchet:
  - Use KDFs to generate new symmetric keys
- Key complexity: asynchronous communications

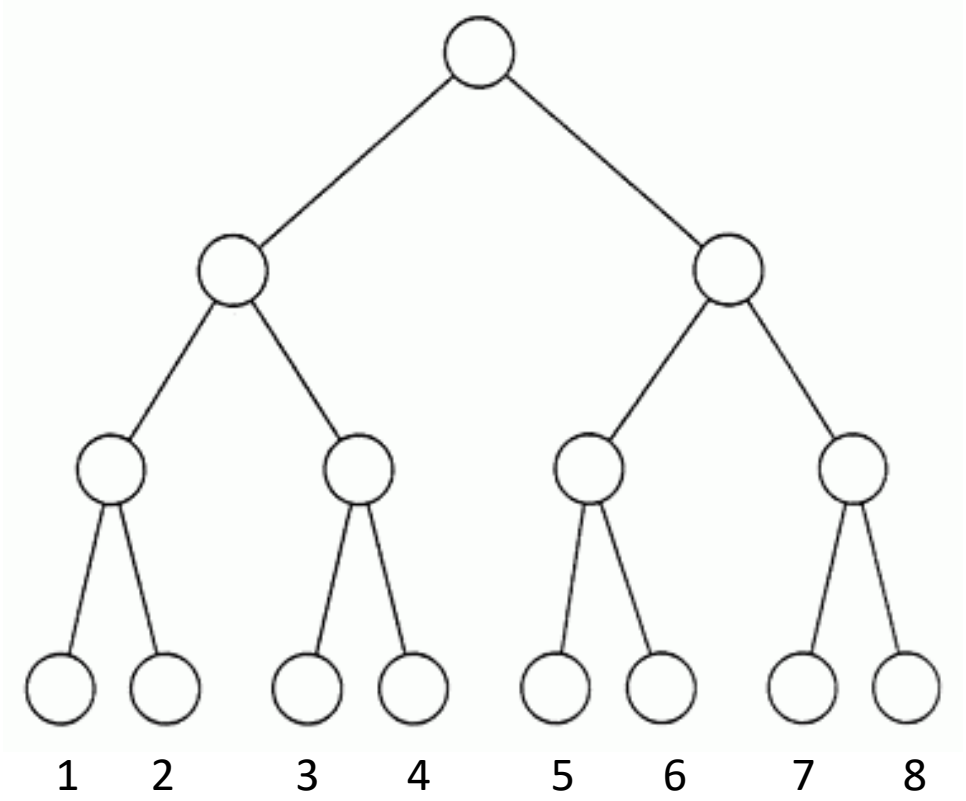
What about group messaging?



- Pairwise broadcast
  - Better: n KEMs and 1 DEM
- Sender keys
  - Each user generates symmetric sender key, sends it to all parties
  - Use sender key when sending
  - Bad post-compromise security: need to do O(n) work to change group membership
- Can we do better?

KEM trees

- Binary tree where each node has (pk,sk) associated to it for KEM
  - Leaves correspond to group members
- ***Invariant***: The private key for a node in the tree is known to a member of the group only if the node's subtree contains that member's leaf.
- ***Updating your own key*****:** Choose new keys for your direct path. Send update message that encrypts node secrets to co-path
  - log(n) ciphertexts
- ***Removing member***: “Blank out” (invalidate) keys along removed member’s direct path. Can encrypt to remaining group using at most log(n) encryptions



Direct path: leaf ancestors
Co-path: children of ancestors not on direct path

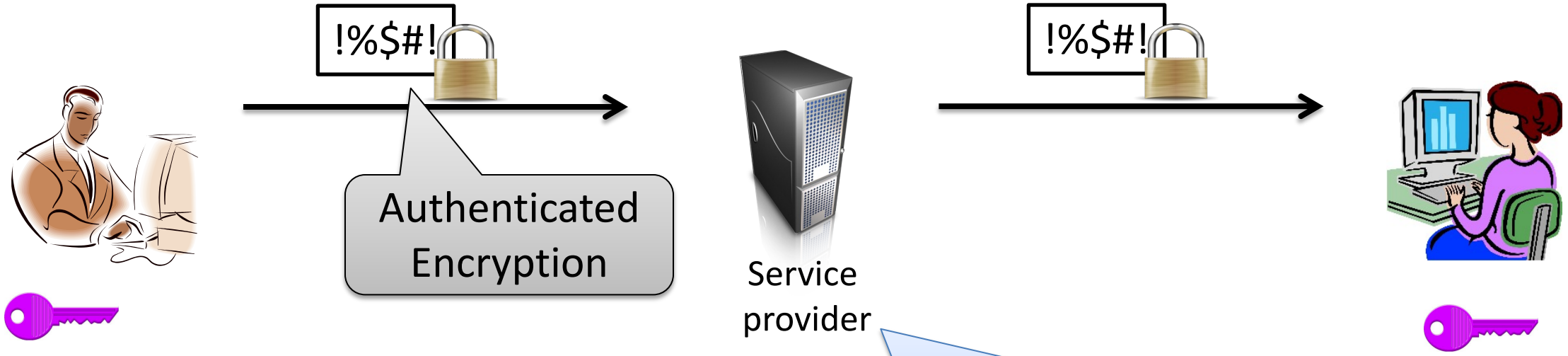
Message Layer Security (MLS)

- New IETF standard (https://datatracker.ietf.org/wg/mls/about/)
- Unified protocol for >=2 E2E encrypted messaging
- Uses KEM trees, symmetric ratcheting, 2-stage commit protocol for consistency of group state, etc.
- OpenMLS library implements it (https://openmls.tech/)

Encrypted messaging big topic

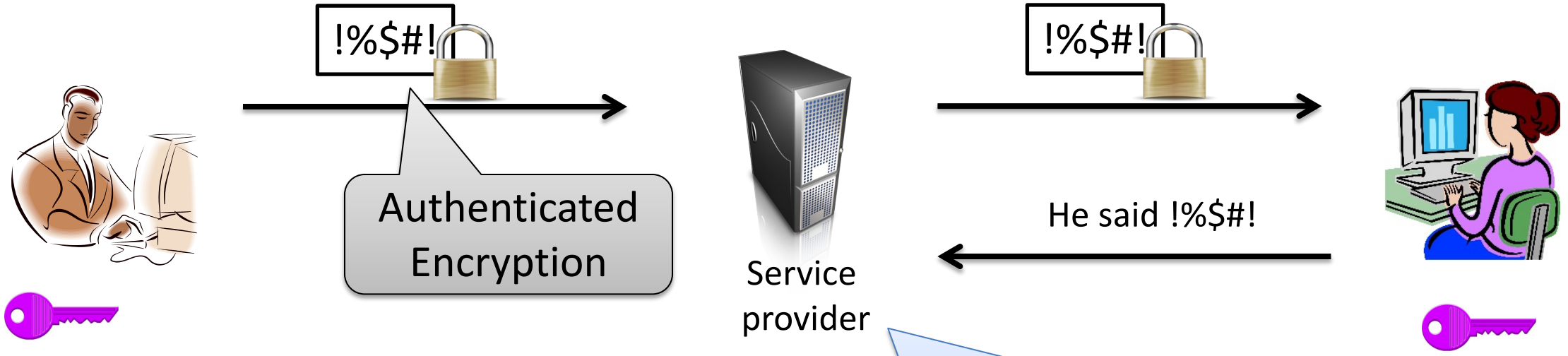
- Google, Facebook (Messenger & WhatsApp), Apple, Signal, Keybase, Skype, ... have implemented some E2E encryption
- Adds complexity to trust and safety features (e.g., abuse reporting)

End-to-end encrypted messaging and abuse



End-to-end security:
Provider cannot read or modify messages

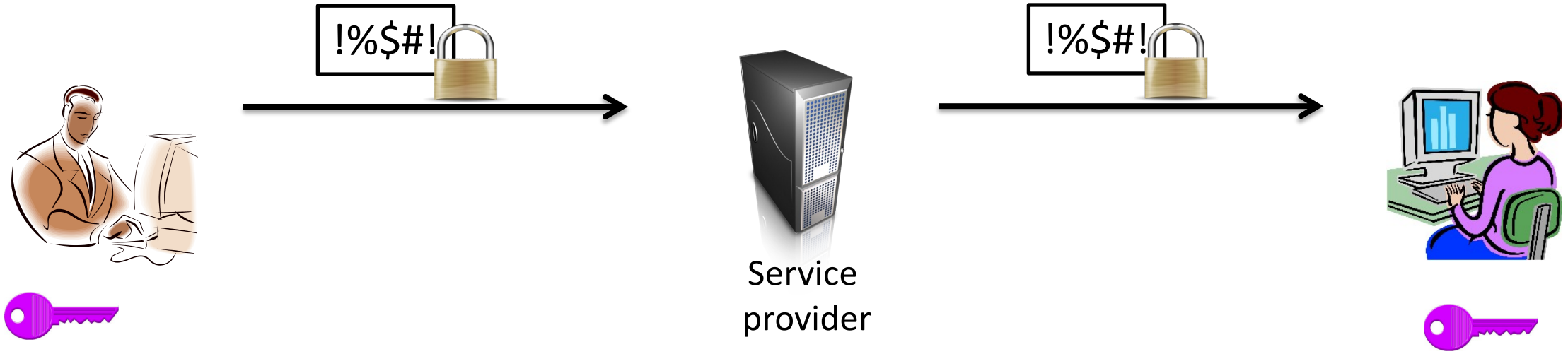
End-to-end encrypted messaging and abuse



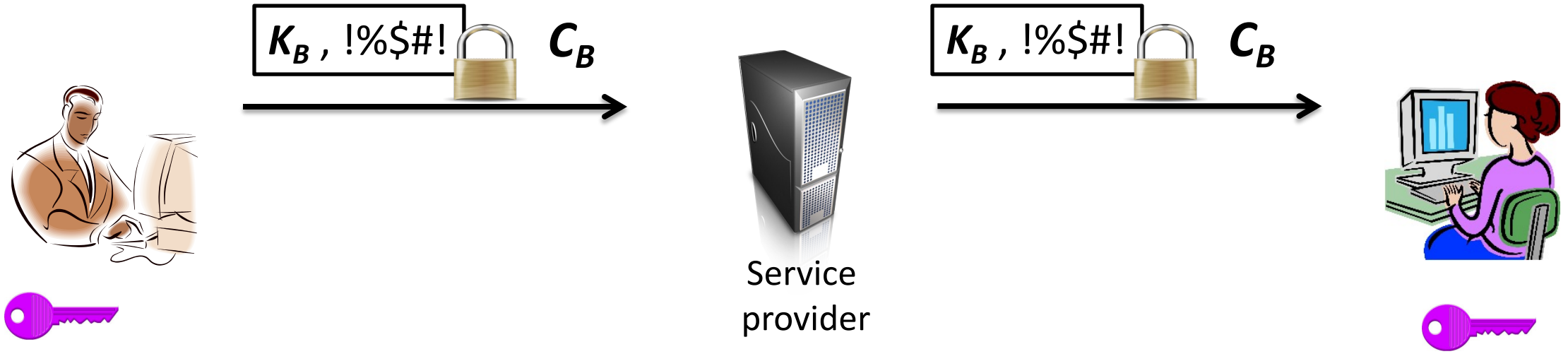
[Facebook 2016]:

- Provide cryptographic proof of message contents when reporting abuse
- Called technique ***message franking***

Facebook's message franking protocol



Facebook's message franking protocol

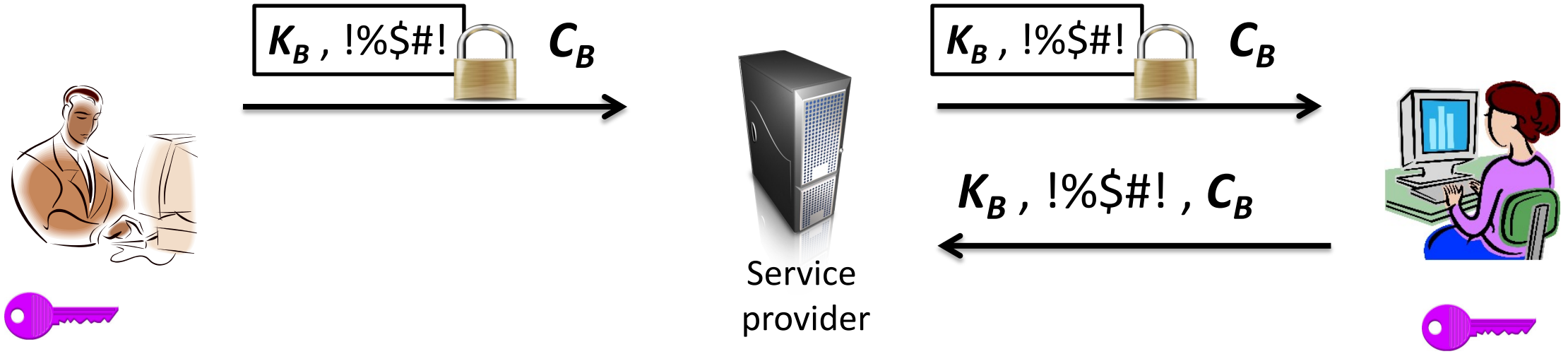


Sender ***cryptographically commits*** to message: $C_B = HMAC(K_B,$ "!%$#!"$)$

Encrypt message along with K_B (called the opening)

Receiver decrypts, retrieves K_B, and verifies C_B

Facebook's message franking protocol

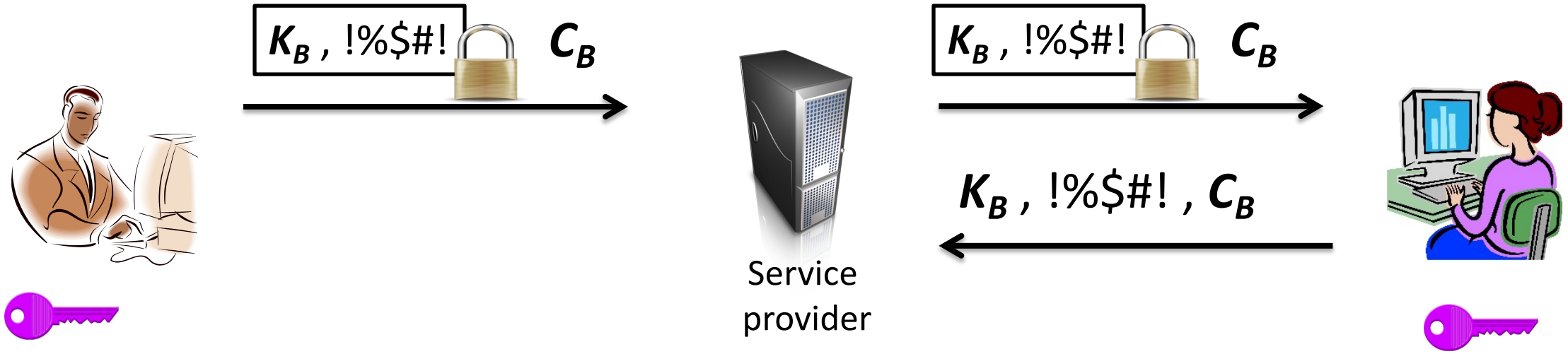


To report abuse, send message as well as K_B , C_B

Provider can verify that C_B was previously sent (and by whom)

Recompute $HMAC(K_B,$ "!%$#!"$)$ and check that it equals C_B

Security goals for message franking



1) ***Receiver binding***: receiver can't open a message not sent

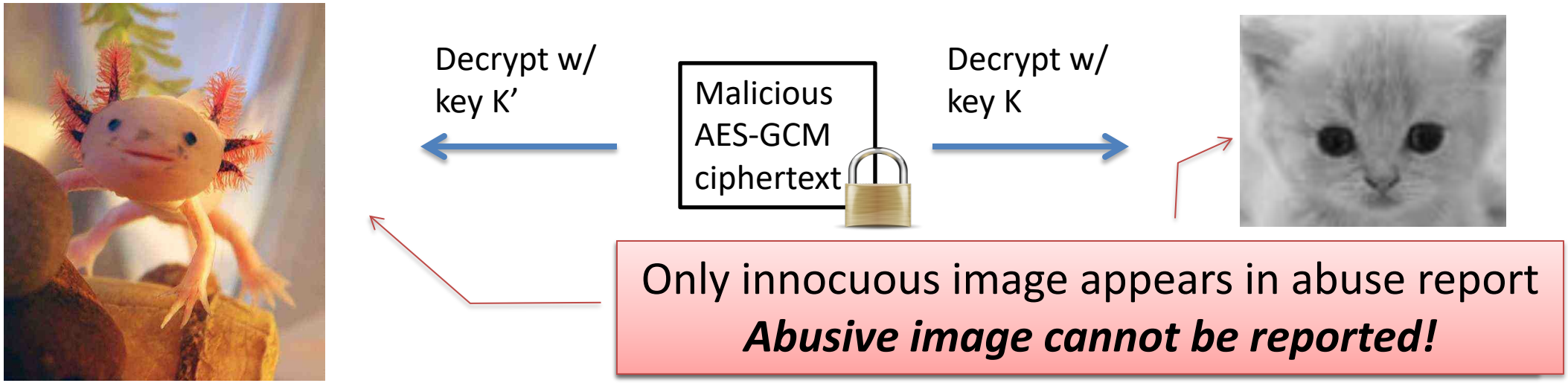
2) ***Sender binding***: can't send a message that can't be reported

3) End-to-end confidentiality/authenticity for messages not reported

Our analysis:

Facebook's scheme achieves this for messages, but not for image/video attachments

An attack against Facebook moderation



Vulnerability due to weakness in standard authenticated encryption (AES-GCM)

- Find two keys K,K' that decrypt ***same*** ciphertext to ***distinct*** images

More recently: exploit this weakness to recover secrets [Len et al. Security 2021]

Abuse mitigation motivated finding fundamental weakness in encryption tools

- New cryptographic theory to guide design of better encryption [Grubbs et al. Crypto 2017]
- New, secure algorithms [Dodis et al. Crypto 2018]

Summary

- Hybrid encryption uses combination of asymmetric and symmetric cryptography
  - Key encapsulation mechanisms (KEM) based on secure PKE, (elliptic curve) Diffie-Hellman
  - Use an authenticated encryption scheme for data encapsulation mechanism (DEM)
- PGP is historical example (and still somewhat widely used)
- End-to-end messaging for IM (chat) hot topic, now widely deployed

Signal Protocol

Identity key

Signed pre-key

Signature of SPK_B using secret key associated to IK_B

One-time pre-keys

Service provider

IK_B, SPK_B, σ , $OPK1_B$, … , $OPKn_B$

IK_B, SPK_B, $OPKi_B$ all public DH keys, recipient retains secret keys

IK_B, SPK_B, σ , $OPKi_B$

Verify σ

DH(PK1,PK2) is means output $g^{sk1*sk2}$

$r \leftarrow\$ \mathbf{Z}_p$; $EK_A \leftarrow g^r$

$DH1 = DH(IK_A, SPK_B)$
$DH2 = DH(EK_A, IK_B)$ — Mutual authentication
$DH3 = DH(EK_A, SPK_B)$
$DH4 = DH(EK_A, OPK_B)$ — Forward secrecy

$SK = KDF(DH1 \,||\, DH2 \,||\, DH3 \,||\, DH4)$

$C \leftarrow AEAD(SK, message)$

IK_A , EK_A, key identifiers, C

IK_A , EK_A, key identifiers, C

Summary

- Hybrid encryption uses combination of asymmetric and symmetric cryptography
  - Key encapsulation mechanisms (KEM) based on secure PKE, (elliptic curve) Diffie-Hellman
  - Use an authenticated encryption scheme for data encapsulation mechanism (DEM)
- PGP is historical example (and still somewhat widely used)
- End-to-end messaging for IM, chat hotter topic, now widely deployed

Summary

- Elliptic curves are specially constructed groups where DLP is conjectured to be hard
- These are faster than RSA or DLP over $\mathbf{Z}_p^*$
- Being used widely in practice
  - EC-DSA (bitcoin)
  - TLS EC-DHE (elliptic curve ephemeral DH)
- Post-quantum crypto studies new asymmetric primitives conjectured to resist quantum computers