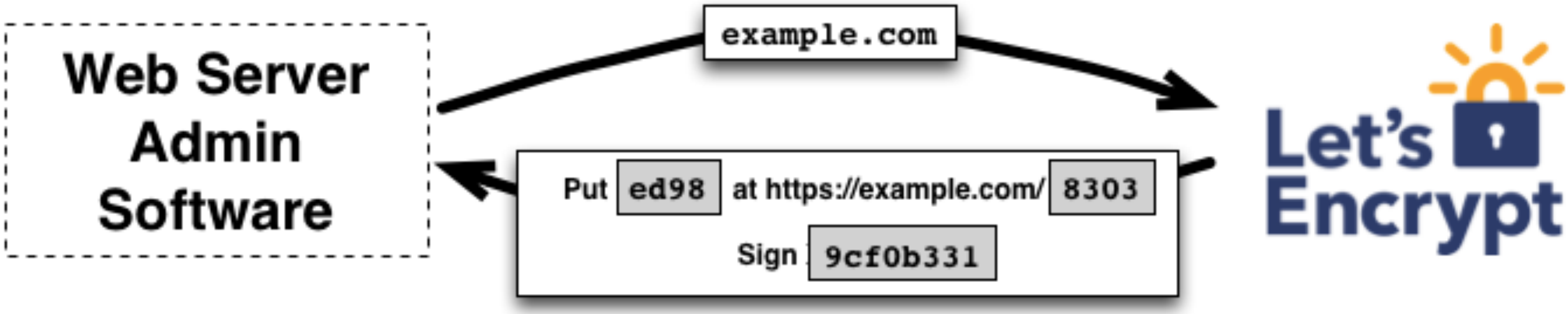
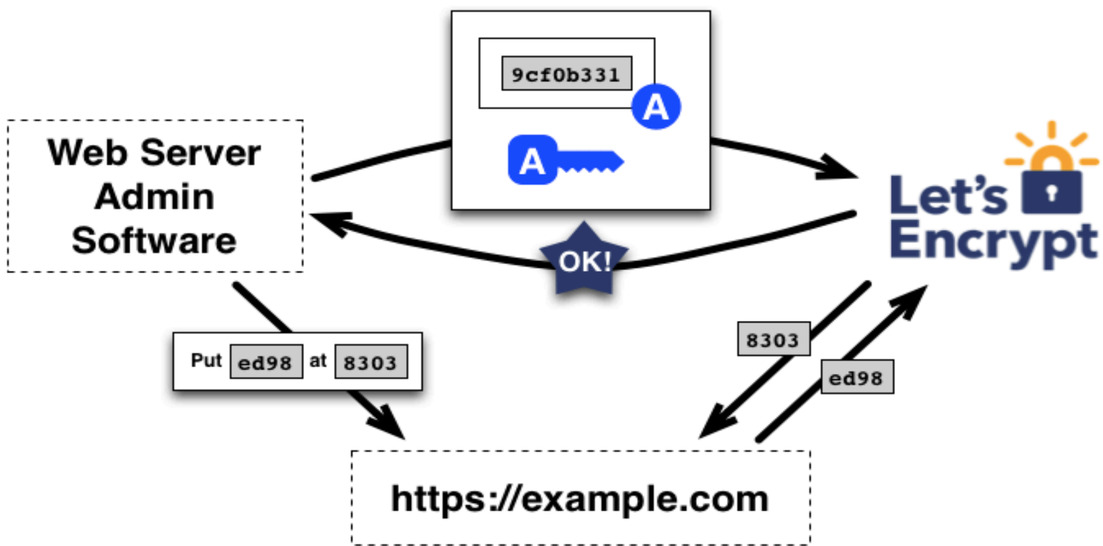
Today in Cryptography (5830)

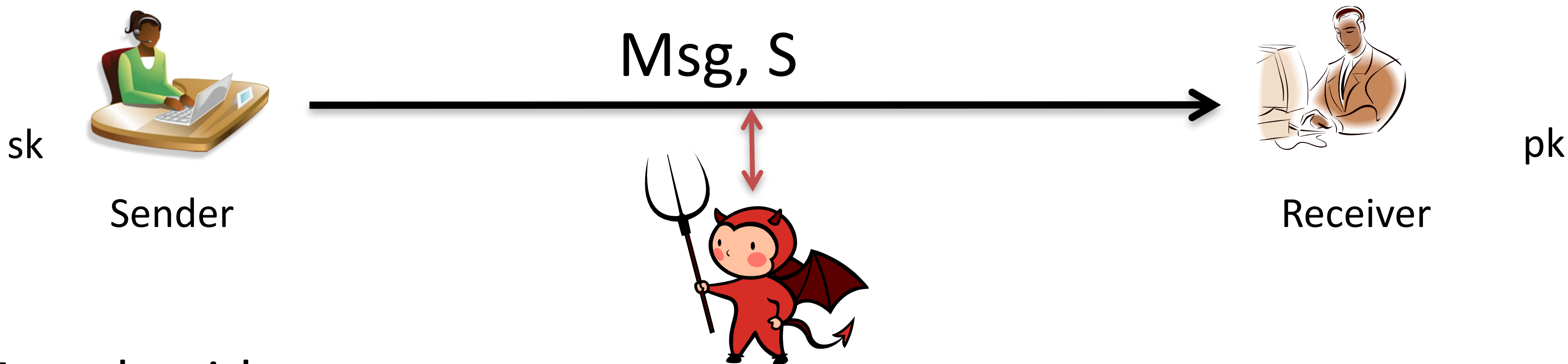
Digital signatures
Schnorr signatures, DSA
Encryption messaging

Free CAs


Web Server Admin Software
example.com
Put ed98 at https://example.com/ 8303
Sign 9cf0b331
Let's Encrypt


9cf0b331
A
A
Web Server Admin Software
OK!
Let's Encrypt
Put ed98 at 8303
8303
ed98
https://example.com

Digital signatures



<u>Two algorithms:</u>

(1) Key generation outputs (pk,sk)
(2) Sign (sk,Msg) outputs a signature S (may be randomized)
(3) Verify(pk,Msg,S) outputs 0/1 (invalid / valid)

Correctness: Verify(pk,Msg,Sign(sk,Msg)) = 1 always

Security: No computationally efficient attacker can forge signatures for a new message even when attacker gets

$$(\text{Msg}_1 , S_1) , (\text{Msg}_2 , S_2), \ldots , (\text{Msg}_q , S_q)$$

for messages of his choosing and reasonably large q.

Groups for Schnorr and DSA Signatures

Let p be a large prime number
Let q be a prime such that q divides p-1
Example: $p = 2q + 1$ (so-called safe prime p)

Fix the group $G = \mathbf{Z}_p^* = \{1,2,3,\ldots, p-1\}$
Let g be generator of sub-group of order q:

$$\{ g^0, g^1, g^2, \ldots , g^{q-1} \} \text{ subset of } G$$

How to pick g?
$g = h^{(p-1)/q} \bmod p$ for some h and check $g \neq 1 \bmod p$
If so, try repeat with another h. Usually start with $h = 2$

(Variant of) Schnorr signatures

p,q,g specified

sk = x chosen randomly from $\mathbf{Z}_q$ pk = X = g^x

<u>Sign(x, M)</u>

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

R = g^r ; c = H(M || R) ; z = r + cx mod q

Return (R,z)

<u>Ver(X, M, (R,z))</u>

c = H(M || R)

If $g^z = RX^c$ then Return 1

Return 0

Correctness? $g^z = g^{r + cx} = g^r g^{xc} = RX^c$

Security intuition

Assume an adversary that can output forgery (M,(R,z))
Then to be valid:

$$g^z = RX^c \quad \text{implies} \quad z = r + cx$$

for c = H(M || R) .
Assume c is random (H is random oracle)

Imagine we can run adversary twice but force forgery to be on same R, different c .
In second execution, getting (M',(R,z'))

Then success second time around gives:

$$g^{z'} = RX^{c'} \quad \text{implies} \quad z' = r + c'x$$

But now can compute z - z' / (c - c') = x the secret key

Fragility of signatures

Repeat randomness failure:

Sign two messages $M \neq M'$ and reuse randomness

$$\mathrm{Sign}(x,M) \to (R,z) = (R, r + cx \mod q)$$

$$\mathrm{Sign}(x,M') \to (R,z') = (R, r + c'x \mod q)$$

Then: $x = (z - z') / (H(M||R) - H(M'||R))$

If r is predictable/leaked, can recover secret from (R,z)

Can improve security by "hedging":

choose $r = H(x || M || \text{randomness})$

Actual Schnorr signatures

p,q,g specified

sk = x chosen randomly from $\mathbf{Z}_q$ pk = X = g^x

<u>Sign(x, M)</u>
r <-$ $\mathbf{Z}_q$
R = g^r ; c = H(M || R) ; z = r + cx mod q
Return (c,z)

<u>Ver(X, M, (c,z))</u>
R' = $g^s X^{-c}$
c' = H(M || R')
If c' = c then Return 1
Return 0

Correctness? $R' = g^s X^{-c} = g^{r+cx} g^{x/(H(M||R))} = g^r$

DSA (digital signature algorithm)

p,q,g specified

sk = x chosen randomly from $\mathbf{Z}_q$ pk = X = g^x

<u>Sign(x, M)</u>

r <-$ $\mathbf{Z}_q$; R = $(g^r \bmod p) \bmod q$

$z = r^{-1}(H(M) + x R) \bmod q$

Return (R,z)

<u>Ver(X, M, (R,z))</u>

$w = z^{-1} \bmod q$

$u1 = H(m) * w \bmod q$

$u2 = R*w \bmod q$

If $R = (g^{u1} X^{u2} \bmod p) \bmod q$

then Return 1

Else Return 0

Correctness?

$$g^{u1} X^{u2} = g^{H(M)\, w}\, g^{x\, R\, w} = g^{(H(M)+xR)\, w}$$

$$= g^{(H(M)+xR)\, (H(M)+xR)^{-1}\, r} = g^r$$

Fragility of DSA

Repeat randomness failure:

Sign two messages $M \neq M'$ and reuse random

$$\mathrm{Sign}(x,M) \to (R,z) \quad = \quad (R,\ r^{-1}(\ H(M) + x\,R\ ) \bmod q\ )$$

$$\mathrm{Sign}(x,M') \to (R,z') \quad = \quad (R,\ r^{-1}(\ H(M') + x\,R\ ) \bmod q\ )$$

Then: Solve for r^{-1}, solve for x

If r is predictable/leaked, can recover secret from (R,z)

Again, can improve security by "hedging":

choose $r = H(\ x \,||\, M \,||\, \text{randomness}\ )$

Hackers Describe PS3 Security As Epic Fail, Gain Unrestricted Access

BY MIKE BENDEL DECEMBER 29, 2010 @ 11:19 AM

fail0verflow

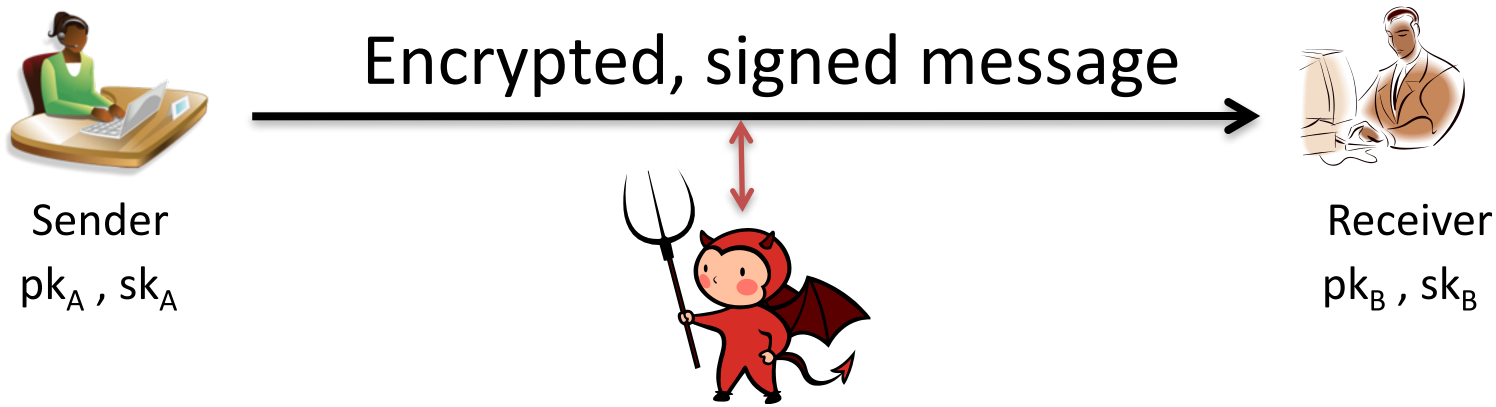
http://psx-scene.com/forums/content/sony-s-ps3-security-epic-fail-videos-within-581/

Application-layer crypto

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

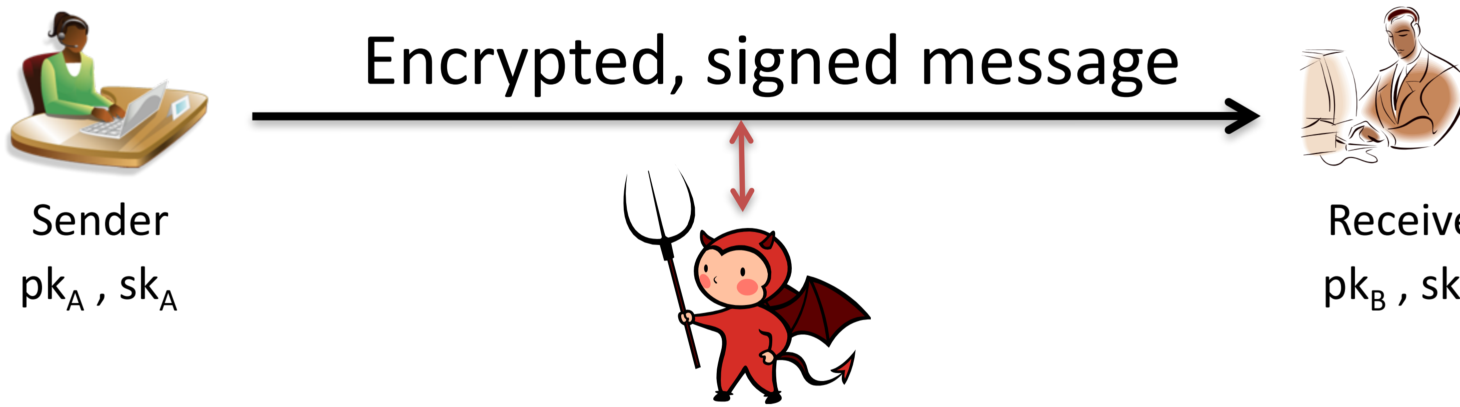
- What about if an application wants to encrypt discrete messages (as opposed to stream)?
  - Email
  - Text messages
  - Etc.

Email encryption



- Message may be large (body of email, PDF of attachments)
- Desire authenticity and confidentiality
- Public-keys delivered out-of-band
    - Websites, key parties, key directory servers

Email encryption



How should we design a solution?

Public-key encryption

Digital signatures

Symmetric authenticated encryption with associated data

ElGamal public-key encryption

g is generator for group of order p

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

<u>Enc((g,X), M, R)</u>

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

$C1 = g^r$

$C2 = X^r * M$

Return C1 , C2

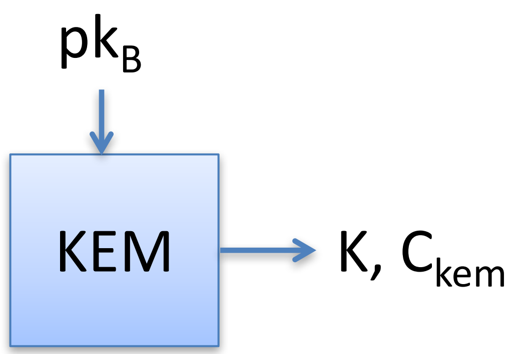
<u>Dec((g,x), C1, C2):</u>

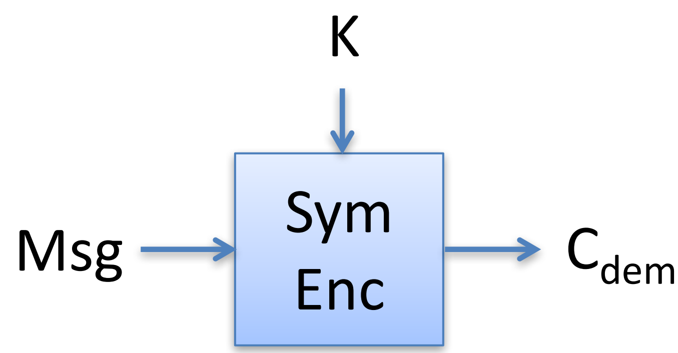
Return $C2 * C1^{-x}$

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)





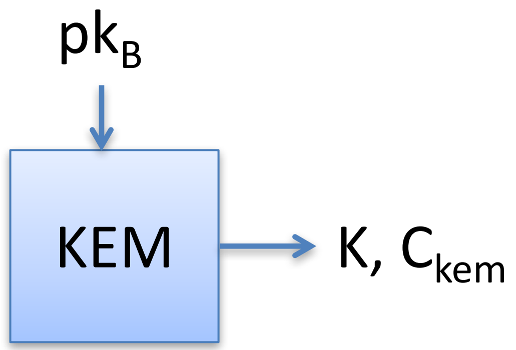
KEM = key encapsulation mechanism
Randomized public-key primitive

DEM = data encapsulation mechanism
One-time secure authenticated encryption

<u>HybEnc(pk, M)</u>
K, C_{kem} <-\$ KEM(pk)
C_{dem} <- Enc(K,M)
Return C_{kem} , C_{dem}

<u>HybDec(sk, C_{kem} , C_{dem})</u>
K <- KEM^{-1}(sk, C_{kem})
M <- Dec(K, C_{dem})
Return M

KEM from PKE



KEM = key encapsulation mechanism
Public-key primitive

<u>KEM(pk)</u>
Choose randomness R
C_{kem} <- PKE-Enc(pk,R)
Return H(R), C_{kem}

ElGamal KEM

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

g is generator for group of order prime p

<u>EG-KEM((g,X), R)</u>

$r = R \bmod p$

$C_{kem} = g^r$

$K = X^r$

Return $H(K)$, C_{kem}

<u>Dec((g,x), C_{kem}):</u>

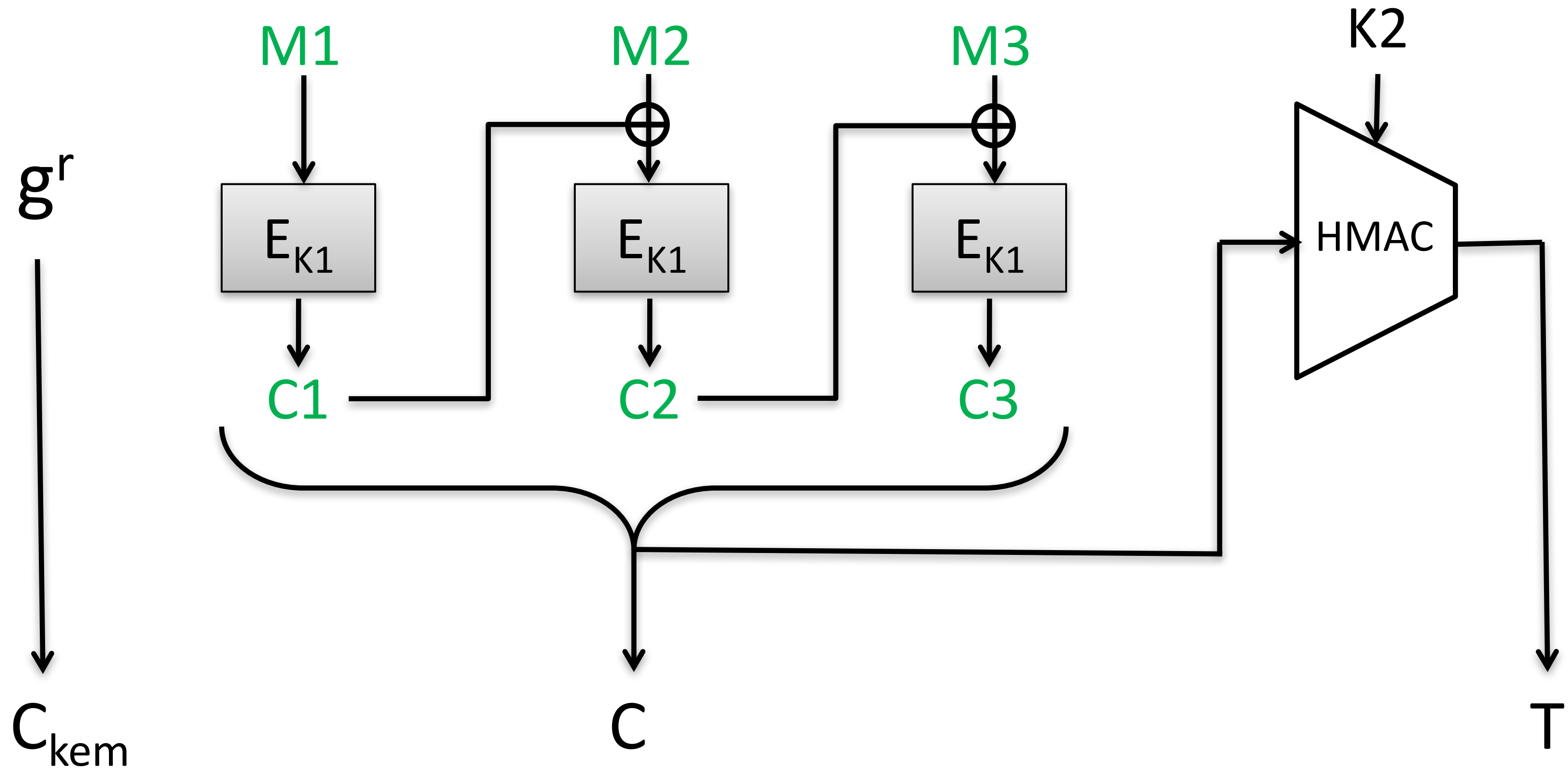
Return $H(C_{kem}^x)$

Secure if computational Diffie-Hellman assumption holds in group

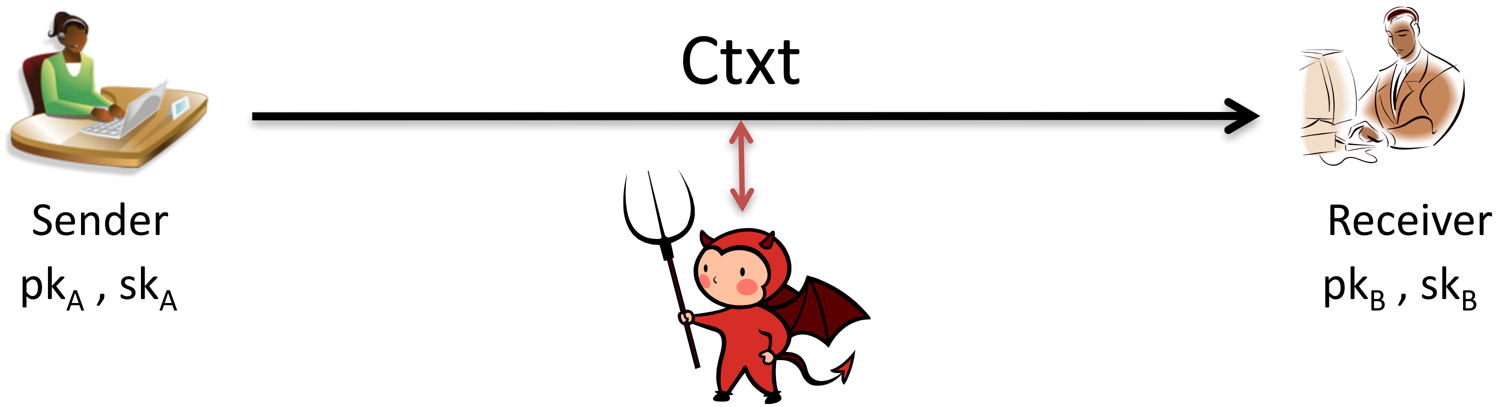
Example hybrid encryption

Enc(X,M):

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



Email encryption



- To digitally sign, let M = Msg || Sign(sk_A , Msg)
- Ctxt = Encrypt(pk_B , M)

PGP history

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

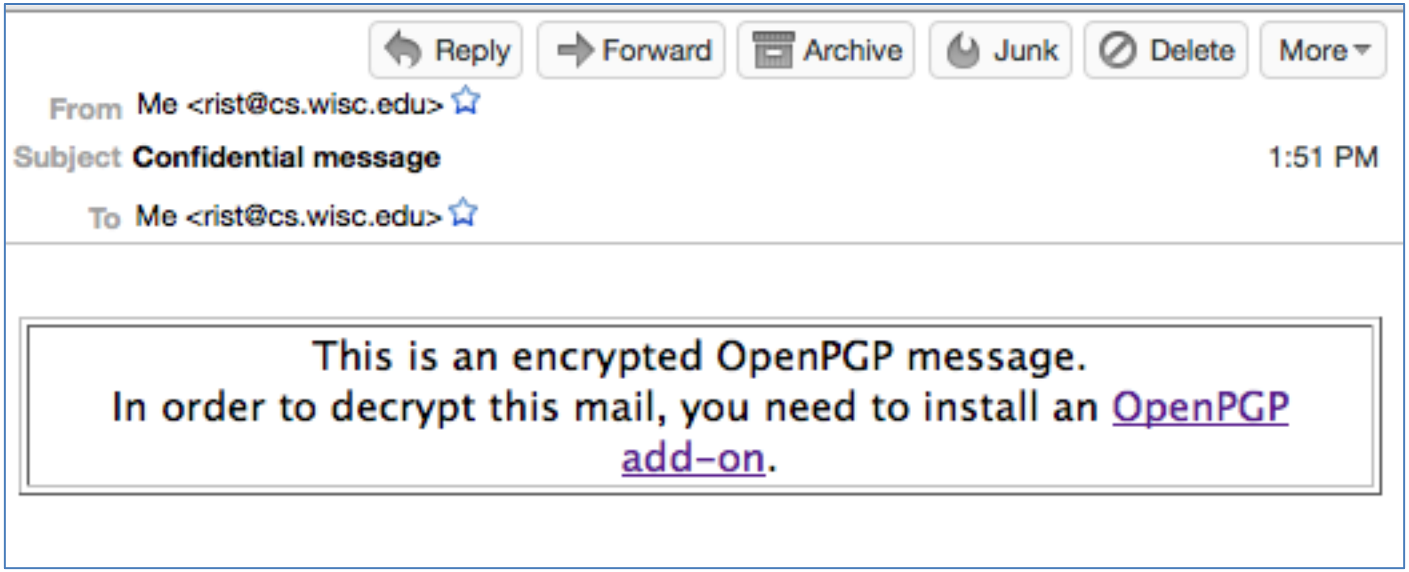
- 1993: Criminal investigation by US government for munitions export without a license.
  - Printed PGP source code into a book. First amendment gambit

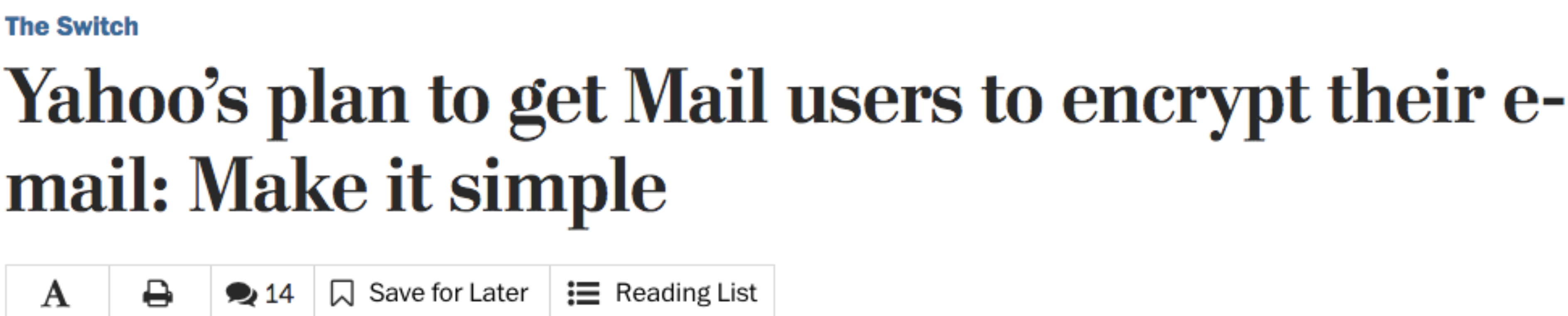
OpenPGP overview

- 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
  - 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


Reply
Forward
Archive
Junk
Delete
More
From Me <rist@cs.wisc.edu>
Subject Confidential message
1:51 PM
To Me <rist@cs.wisc.edu>
This is an encrypted OpenPGP message.
In order to decrypt this mail, you need to install an OpenPGP add-on.


The Switch
Yahoo's plan to get Mail users to encrypt their e-mail: Make it simple
14
Save for Later
Reading List

Messaging encryption

Encrypted/Signed SMS or IM

Sender

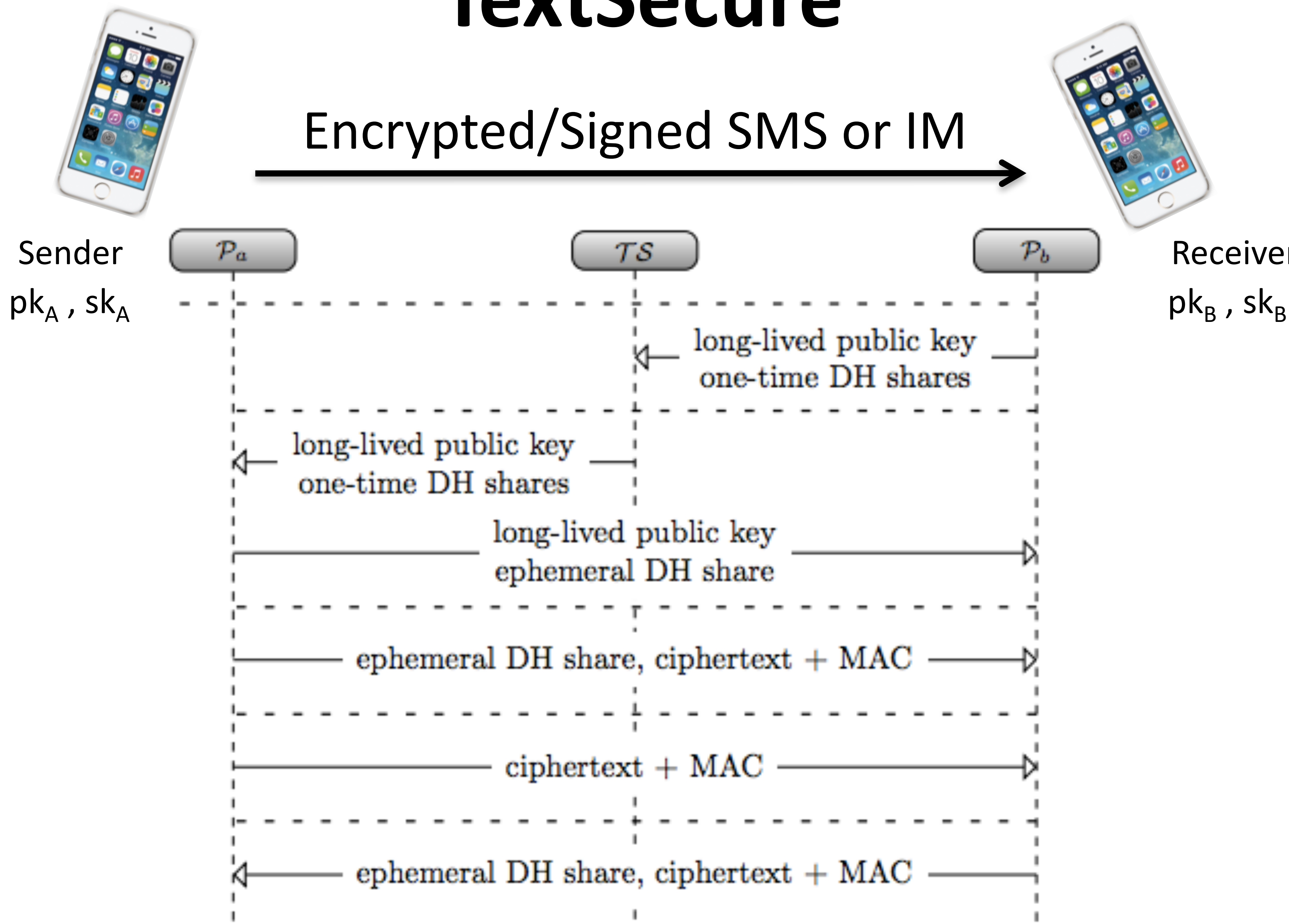
pk_A , sk_A

Receiver

pk_B , sk_B

- End-to-end encrypted messaging is a big topic
- TextSecure is protocol adopted by WhatsApp (~1 billion users)

TextSecure



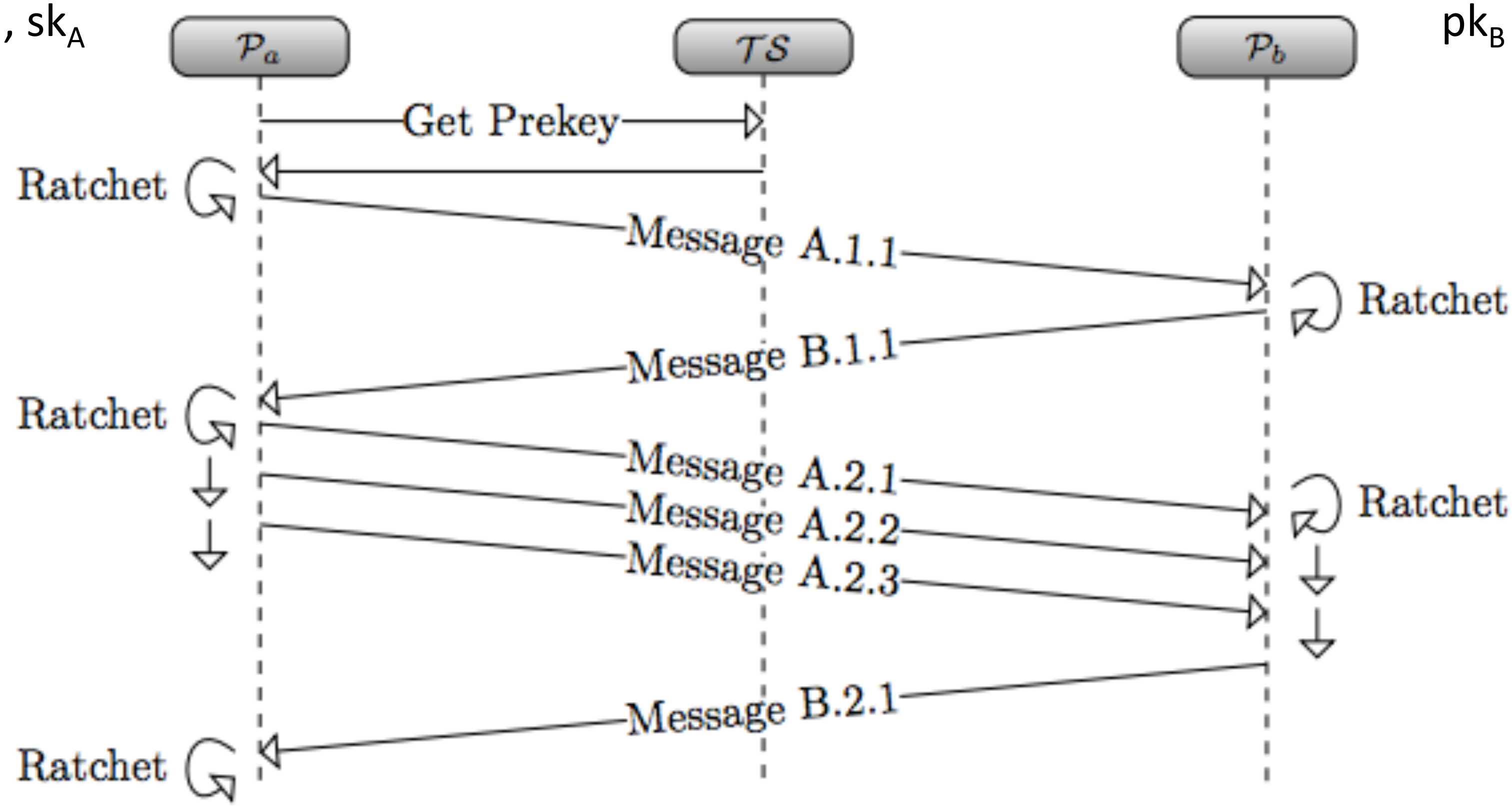
https://eprint.iacr.org/2014/904.pdf

TextSecure

Encrypted/Signed SMS or IM

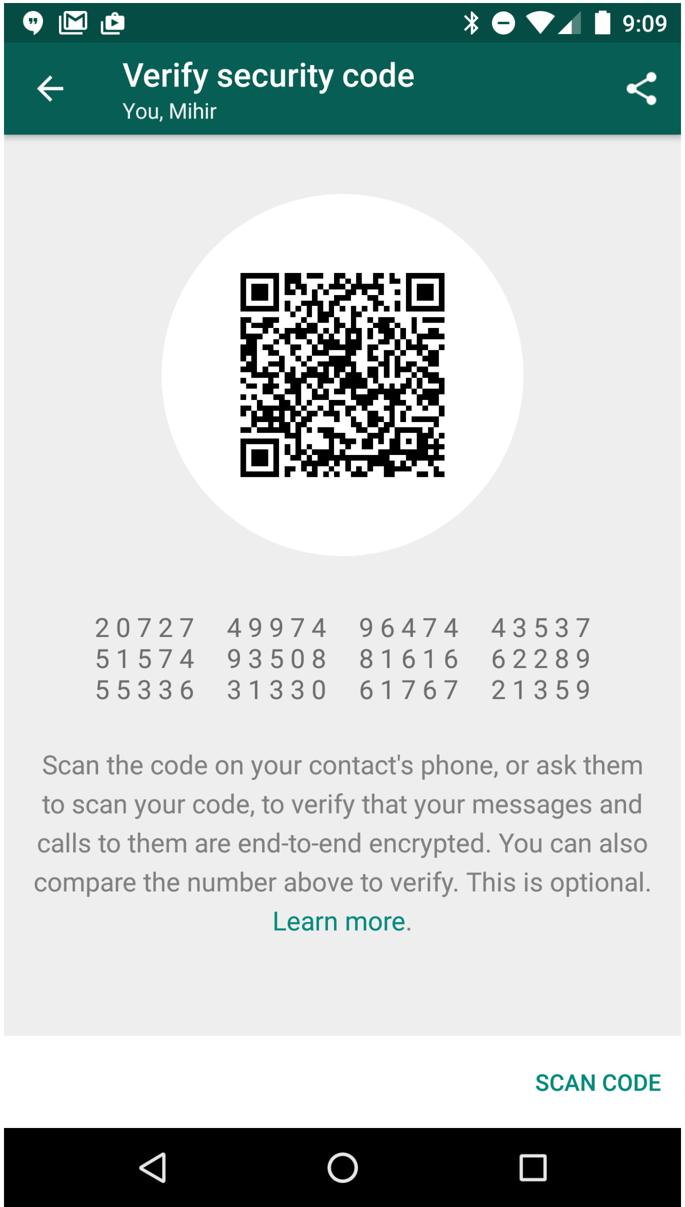
Sender
pk_A , sk_A

Receiver
pk_B , sk_B



https://eprint.iacr.org/2014/904.pdf

Verifying public keys



Summary

- Schnorr and DSA allow discrete-log based digital signatures, but are fragile without hedging
- 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