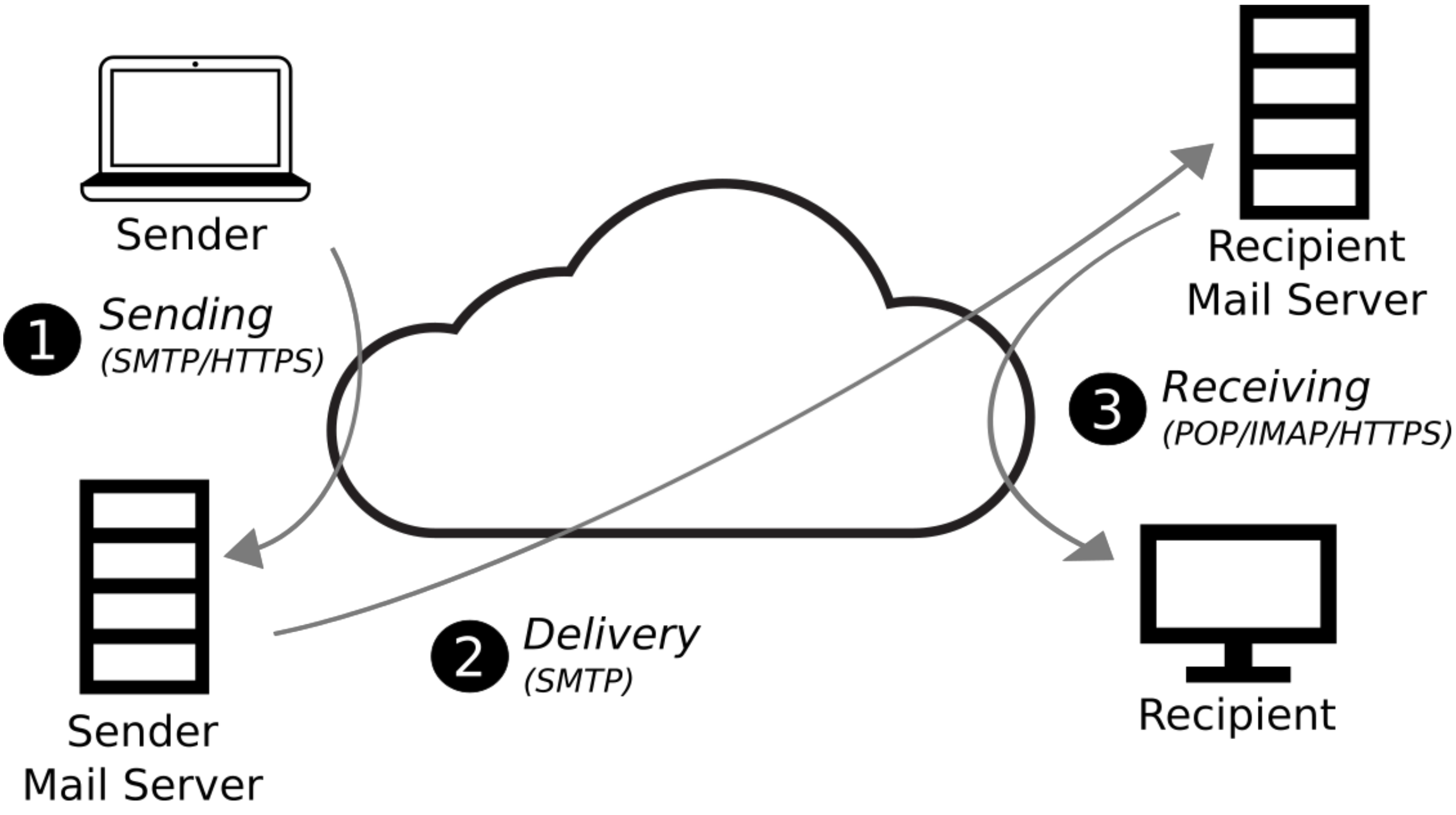
# CS 465
# Computer Security

## Secure Email

# Email

- Has been used for 50+ years!
- Seamless interoperability among all email users, regardless of client device or software
  - compare to instant messaging, which is often a *walled garden*
- As of 2017, 6.3 billion email accounts, owned by 3.7 billion users (almost half of the earth!)

# Email Protocols



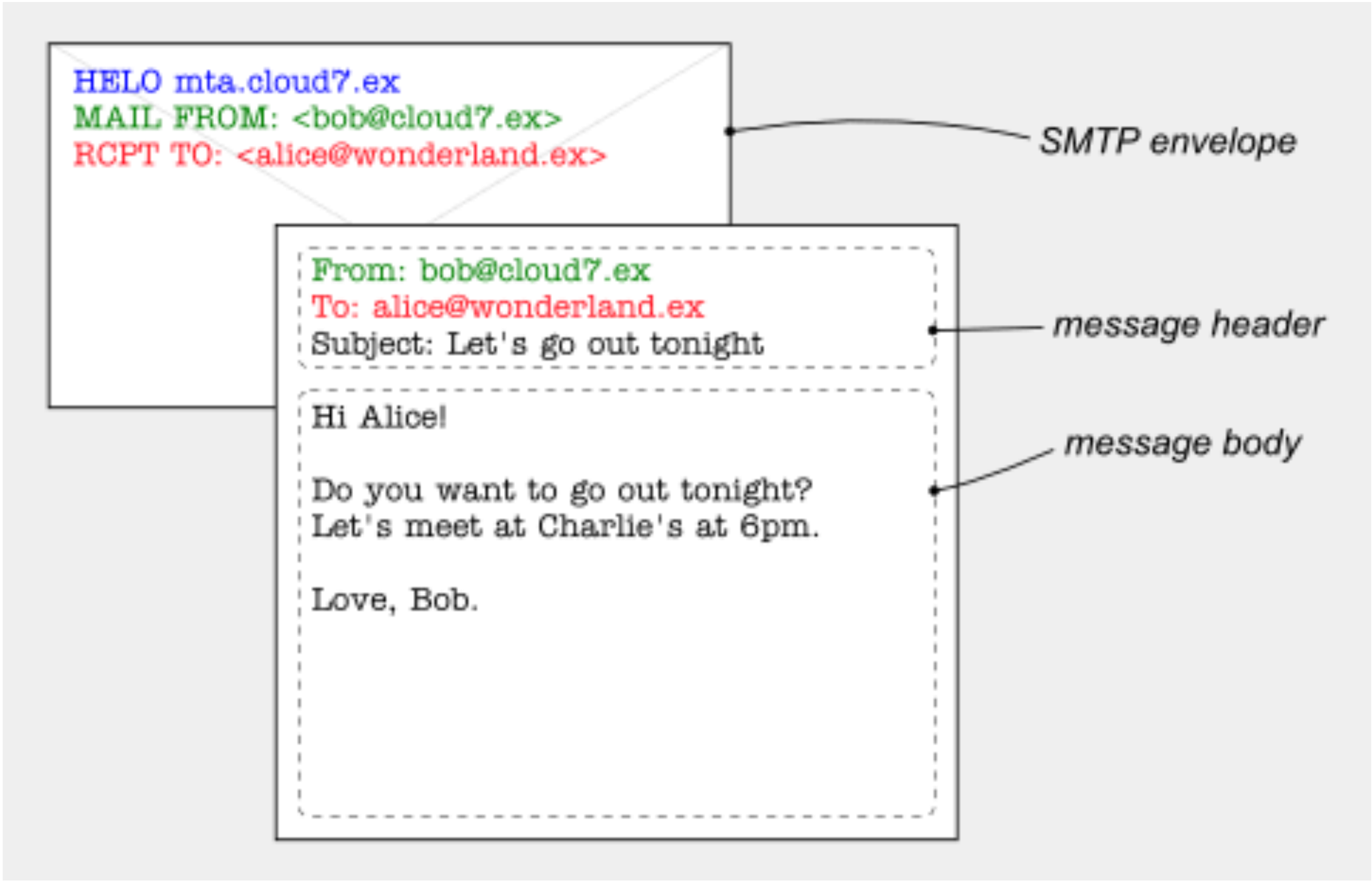
# Steps

1. Sending client submits email to server — HTTPS or SMTP

2. Originating server transfers to destination server — SMTP, may take multiple hops

3. Receiving client receives email from server — HTTPS or IMAP or POP

# SMTP


HELO mta.cloud7.ex
MAIL FROM: <bob@cloud7.ex>
RCPT TO: <alice@wonderland.ex>
SMTP envelope
From: bob@cloud7.ex
To: alice@wonderland.ex
Subject: Let's go out tonight
message header
Hi Alice!
Do you want to go out tonight?
Let's meet at Charlie's at 6pm.
Love, Bob.
message body

# Why is Email Insecure

- No authentication in SMTP
  - Can list any MAIL FROM address in the SMTP envelope
  - Can list any FROM address in the message header
- Mail servers were originally open relays — would forward messages from anyone to anyone
- No confidentiality or integrity
- No anonymity, deniability, freedom from tracing, or ephemerality

# Problems

- Spam
  - In the 2010s, a typical campaign involved using a botnet of 3,000 compromised machines to send 25 billion emails to 10 million email addresses.
  - Pipeline of compromised machines, email list, software tools owned and sold by specialists, campaign run to generate pay-per-click revenue
  - Storefronts for unregulated pharmaceuticals are over a third of spam
  - https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6957289

# Problems

- Malware
  - trick users into downloading (e.g. a spreadsheet) or clicking (e.g. malware website)
    - https://www.rsa.com/en-us/blog/2017-12/anatomy-of-an-attack-carbanak
- Phishing and spear-phishing
  - $100 million bank transfer fraud from 2013 — 2015
  - DNC and Podesta hacks in 2016 Presidential election
  - http://people.cs.vt.edu/danfeng/papers/Phishing-IMC-2018.pdf

# Protection by email providers

# Link Encryption

- Can upgrade SMTP to use TLS using the STARTTLS command
  - works only between pairs of mail servers
  - trivial for an adversary to conduct a downgrade attack by stripping this command
- Strict Transport Security (STS) — DNS records that advertise STARTTLS requirement

# Domain Keys Identified Mail (DKIM)

- sending mail server signs outgoing email (message body, not the envelope)
- receiving mail server authenticates signature using public key of domain
  - find the public key via DNS
- what to do if DKIM fails is up to the receiving mail server — can mark or drop

# Sender Policy Framework (SPF)

- organization publishes DNS record indicating which IP addresses can be used to send email for their domain
- receiving mail server validates that IP address of sending mail server is on the list for the sending domain (as listed in the SMTP envelope)
- what to do if SPF fails is up to the receiving mail server — can mark or drop

# Domain Message Authentication, Reporting, and Conformance (DMARC)

- organization publishes a DNS record indicating which authentication services (DKIM, SPF) it supports, plus a policy indicating action to be taken if authentication fails
- requires identifier alignment
  - DKIM: domain used for signing must match domain in the header *From* address seen by the user
  - SPF: domain in the SMTP envelope must match domain in the header *From* address seen by the user
- allows receiving mail server to send reports of failure back to sending mail server, to identify misconfiguration or abuse

# Authenticated Receive Chain (ARC)

- extends authentication to handle cases when messages are forwarded (e.g. to a mailing list)

- forwarding mail server adds headers indicating status of authentication checks from originator or any forwarder (e.g. DKIM, SPF, DMARC)

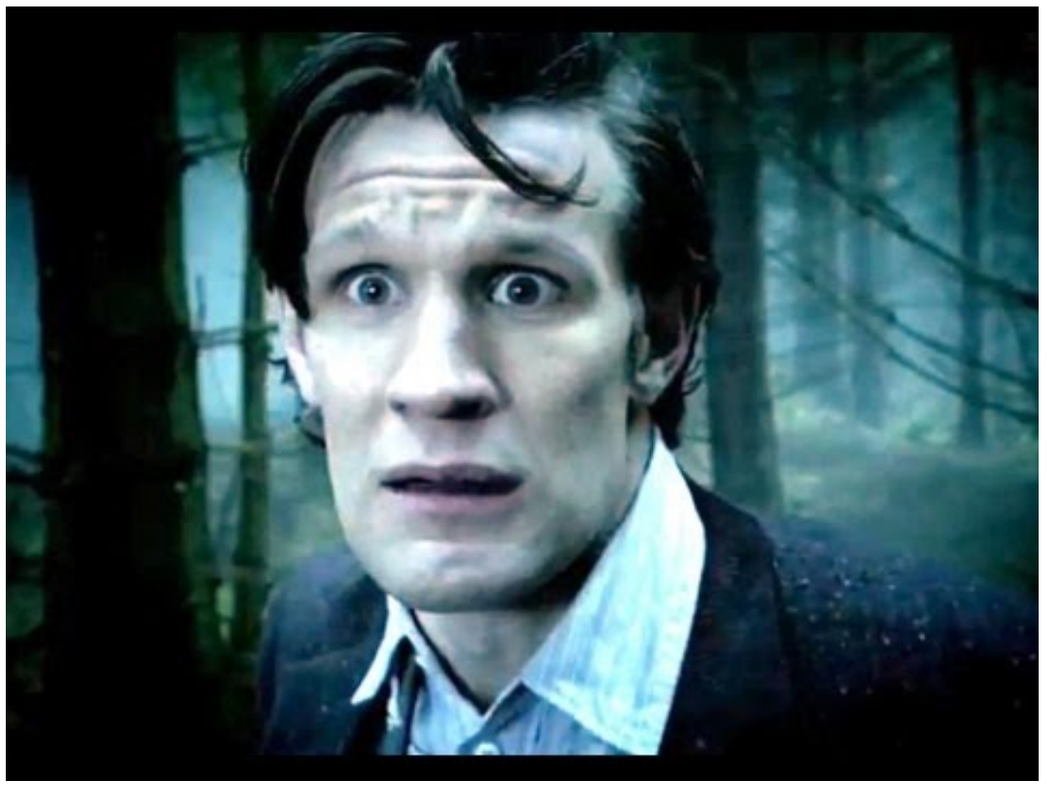
- forwarding mail server signs the message header and body as it was received, plus signs additional ARC headers it adds

- with multiple forwarders, can authenticate entire chain

# End-to-end encryption

# End-to-end Encryption

- encryption from the sending client to the receiving client
- provides both confidentiality and integrity
- end-to-end encryption + seamless global interoperability = disaster



# S/MIME

- developed in early 1990s
- end users obtain certificates from a CA and email clients look up certificates in a certificate directory
  - encrypt with recipient's public key
  - verify signature with sender's public key
- typically used by a company or government department with a locally trusted root, a local certificate directory, and private key escrow
  - provide email scanning (spam/malware), comply with regulations that require access to plaintext, provide recovery if client loses key

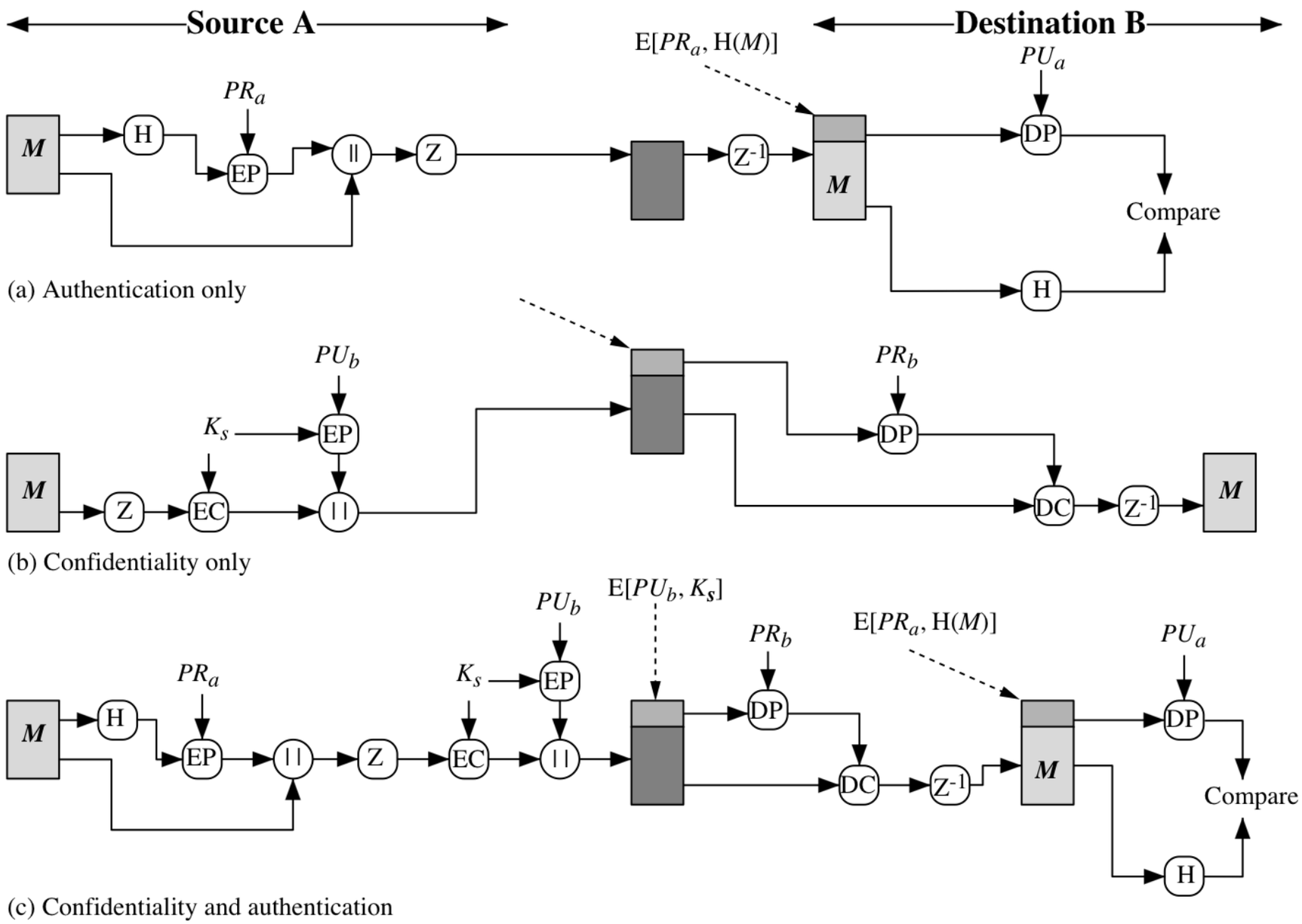
# Pretty Good Privacy (PGP)

- developed in the early 1990s by Phil Zimmerman
- avoids certificate hierarchy in favor of web of trust — each users independently decides which public keys to associate with a given email address
  - user associates name and email address with a key
  - other users can add signatures attesting to this binding — was typically done at key signing parties
  - can receive signed keys from other users you trust — forming a web of trust
  - can publish keys on key servers, e.g. https://pgp.mit.edu/

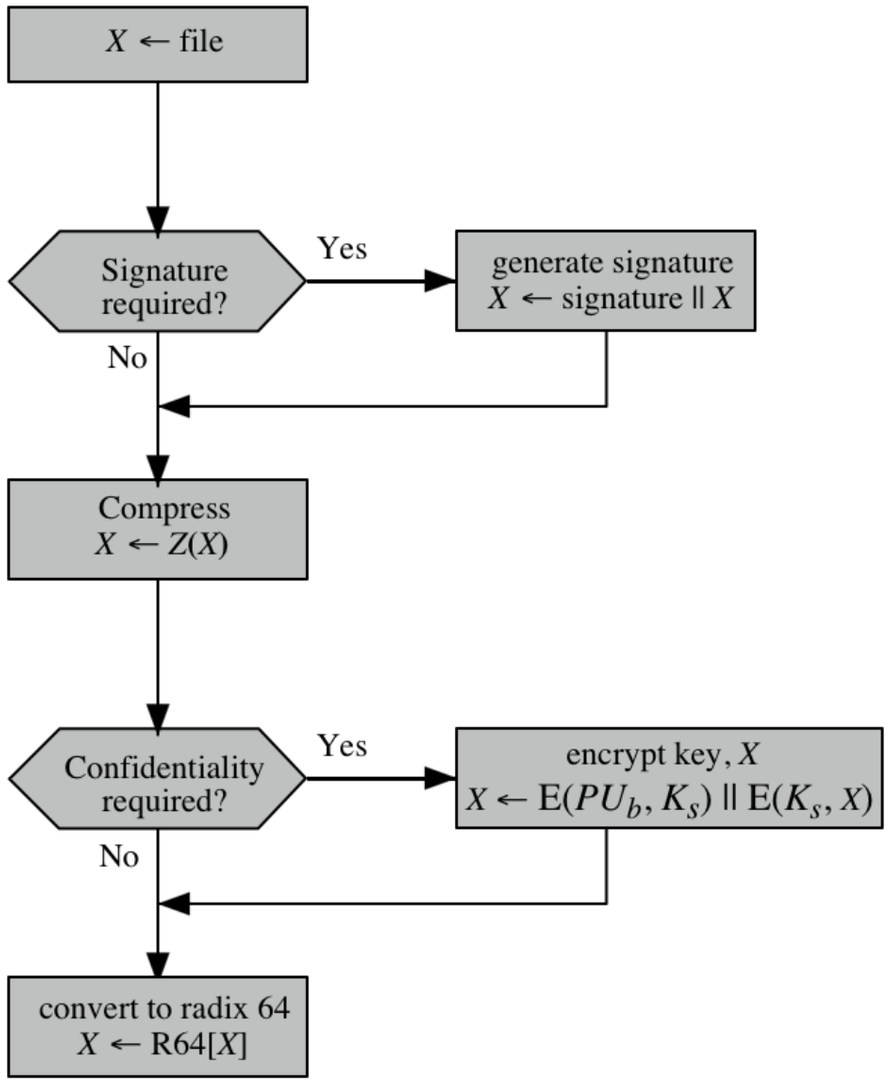
# Pretty Good Privacy (PGP)

- distributed as freeware on the Internet in 1991, leading to an investigation of Zimmermann by the United States Customs Office for allegedly violating U.S. export laws
- he published the PGP source code in book form in 1995, and the case was subsequently dropped in 1996
- in the 1990's, one way to skirt federal export controls was to publish the source code in book form (1st Amendment), ship the books to Europe, have people retype or scan the source code using OCR
- https://heinonline.org/HOL/Page?collection=journals&handle=hein.journals/syrlr48&id=1331&men_tab=srchresults

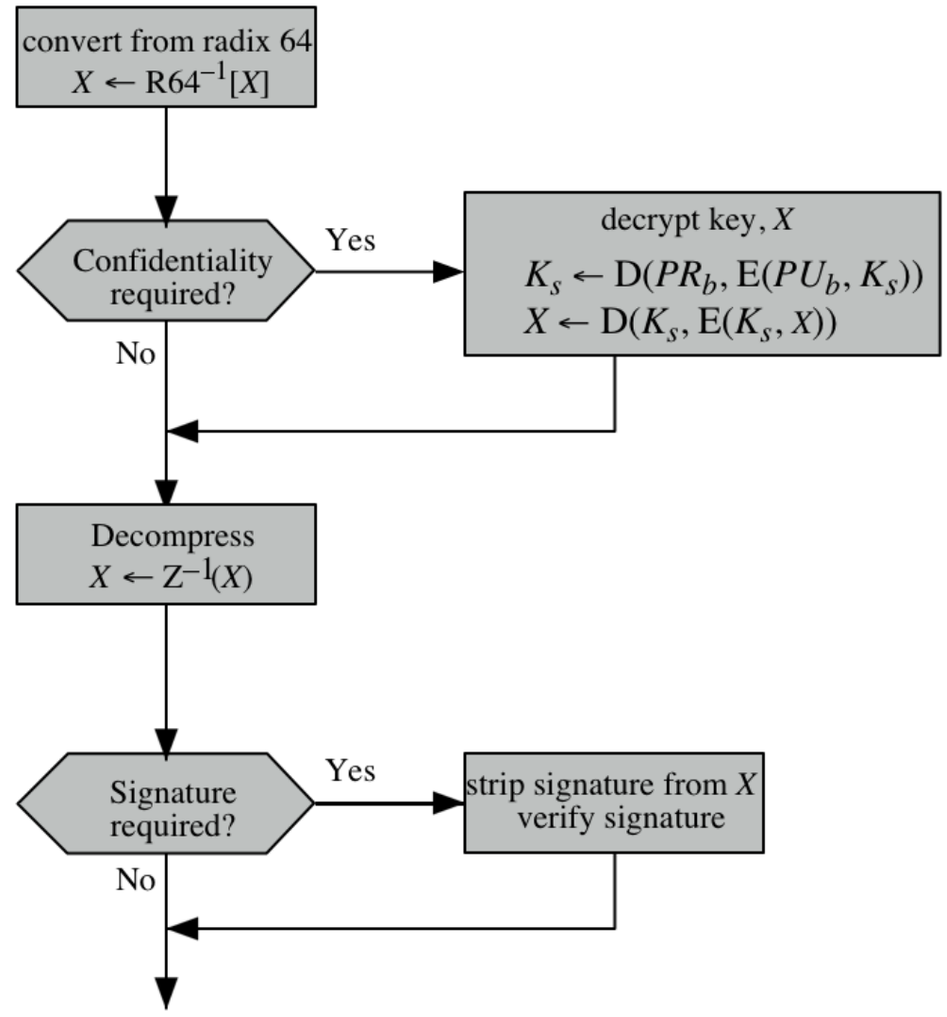
# PGP Cryptographic Functions



# PGP Encryption and Decryption




(a) Generic Transmission Diagram (from A)




(b) Generic Reception Diagram (to B)

# Difficulties with PGP

- difficult to scale beyond small groups due to web of trust an manual key management
- difficult to revoke compromised keys
- difficult for non-technical users — and plagued by poor implementations
    - Why Johnny Can't Encrypt (1999): https://people.eecs.berkeley.edu/~tygar/papers/Why_Johnny_Cant_Encrypt/USENIX.pdf
    - Why Johnny Still Still Can't Encrypt (2016): https://arxiv.org/pdf/1510.08555.pdf
- People are giving up on PGP
    - Filippo Valsorda, long-time user, dislikes the long-term key model and lack of forward secrecy: https://blog.filippo.io/giving-up-on-long-term-pgp/
    - Moxie Marlinspike considers it to be too complex and outdated: https://moxie.org/blog/gpg-and-me/

# Next generation PGP

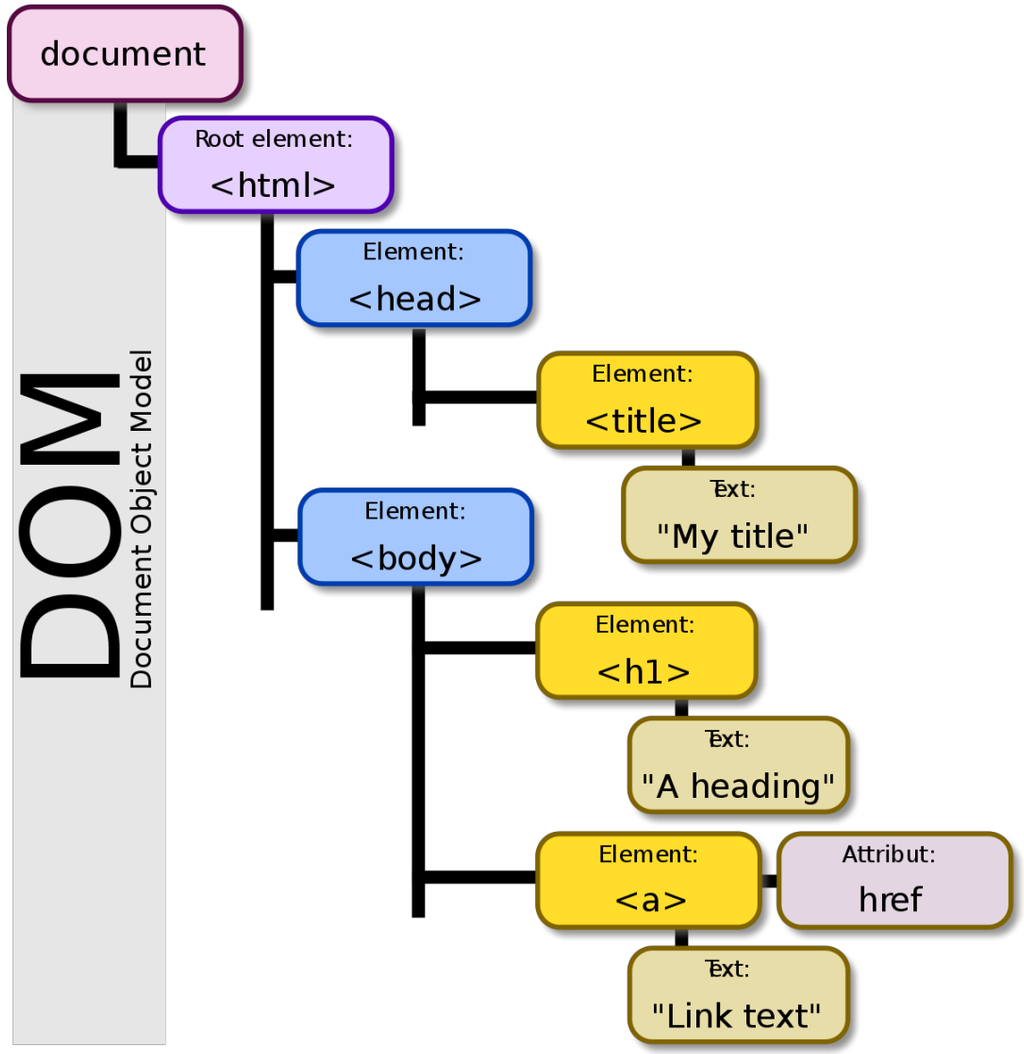
- trusted public key servers (e.g. gmail.com would run one for its users)
- audited key servers — use a public ledger that allows monitors to examine a history of advertised public keys and detect rogue certificates and misbehaving servers
- trust-on-first-use key exchange
- forward secrecy
- see the autocrypt standard (TOFU) and mailipile for examples of where the PGP community is working on now
  - https://github.com/mailpile/Mailpile/wiki/Security-roadmap

# Secure Webmail

- provides secure email (usually based on PGP) for users of the same service
  - email server stores public keys and (password-protected) private keys for all users
  - sending: JavaScript automatically looks up public key on the server and encrypts email for the user you send to, JavaScript also retrieves your private key and signs outgoing emails
  - receiving: JavaScript downloads email, retrieves your private key, and decrypts it
  - if sending to a user of another system, can choose a password for encrypting the email, and the recipient gets a link to the password-protected email on the email server's website
- examples: ProtonMail, Tutanota

# Secure Webmail

- security issues
  - must trust the JavaScript (long term, maybe it is signed and verified free of bugs)
  - third-party JavaScript can access the DOM and thus get the plaintext (long term, maybe browsers will implement privilege separation between JavaScript)
  - alternatively, use a non-browser client or a browser extension



# State of Secure Email

- Enterprises / governments : S/MIME with a local certificate directory and CA, plus private key escrow
- Individuals: Secure webmail (e.g. ProtonMail)
- Journalists: Small PGP deployments
    - When the Weakest Link is Strong: Secure Collaboration in the Case of the Panama Papers: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/mcgregor
- No one-size-fits-all solution and little interoperability between similar solutions or between different solutions
- End-to-end encryption without escrow means your provider can’t scan emails for spam, phishing, and malware, and can’t provide search over your archives
- Phishing is a huge problem
- Most users store extensive, unencrypted email archives with their providers — based on the 1986 Electronic Privacy Communications Act, any email stored over 180 days in the US may be read by federal government without a warrant — see https://epic.org/privacy/ecpa/

# Why is Secure Email Not More Widely Used?

- lack of perceived risk
  - info being sent is not valuable
  - they are not a valuable target
- poor awareness of effective security practices
- routine encryption of email considered paranoid
- requires cooperation from people you send email to (must use a compatible system), so it can’t be unilaterally adopted
- no system today provides universal compatibility

# Encryption in the US

- Longstanding debate between privacy advocates and federal law enforcement regarding end-to-end encryption
    - federal government would like some way to get access to encrypted data (sometimes called an “encryption backdoor”)
    - privacy advocates say this is both not desirable and not technically feasible
- Originally started in the 1990s — NSA developed the “Clipper chip” designed to perform encryption with a backdoor
    - each chip programmed with a key and key also put into escrow (with NIST and Treasury)
    - government can get the decryption key with a warrant
    - plagued by technical flaws, stiff opposition from privacy groups, resistance from industry
- Rekindled recently with widespread adoption of smartphone encryption and secure messaging apps (e.g. Signal)

# Encryption in the US

- Based on the fear that if there is strong end-to-end encryption, then criminals and terrorists will be able to use it to communicate securely and federal government will lose its ability to wiretap
- Privacy advocates point to a history of wiretapping abuses, suggest there will be plenty of unencrypted data and metadata available for authorities
- Security community worries about reversing progress in deploying forward secrecy, substantial increases in complexity that makes it harder to build secure systems, concentration of value for targeted attacks (e.g. on an escrow system)
- Jurisdictional issues on a global Internet are a significant complication
- Whenever service providers have access to keys that can decrypt customer email, this allows plaintext to be revealed due to incompetent or untrustworthy service providers, by disillusioned employees, by government subpoena, or by regulatory coercion
- The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption (1997): https://academiccommons.columbia.edu/doi/10.7916/D8GM8F2W
- Keys under doormats: Mandating insecurity by requiring government access to all data and communications (2015): https://dspace.mit.edu/handle/1721.1/97690