OPES and E2E Encryption

- Should OPES be compatible with end-to-end encryption?
  - Define “compatible”
  - Define the trust model
  - Discuss pro and con
  - Decide, spec, implement
- Goal: combine confidentiality with services, if possible

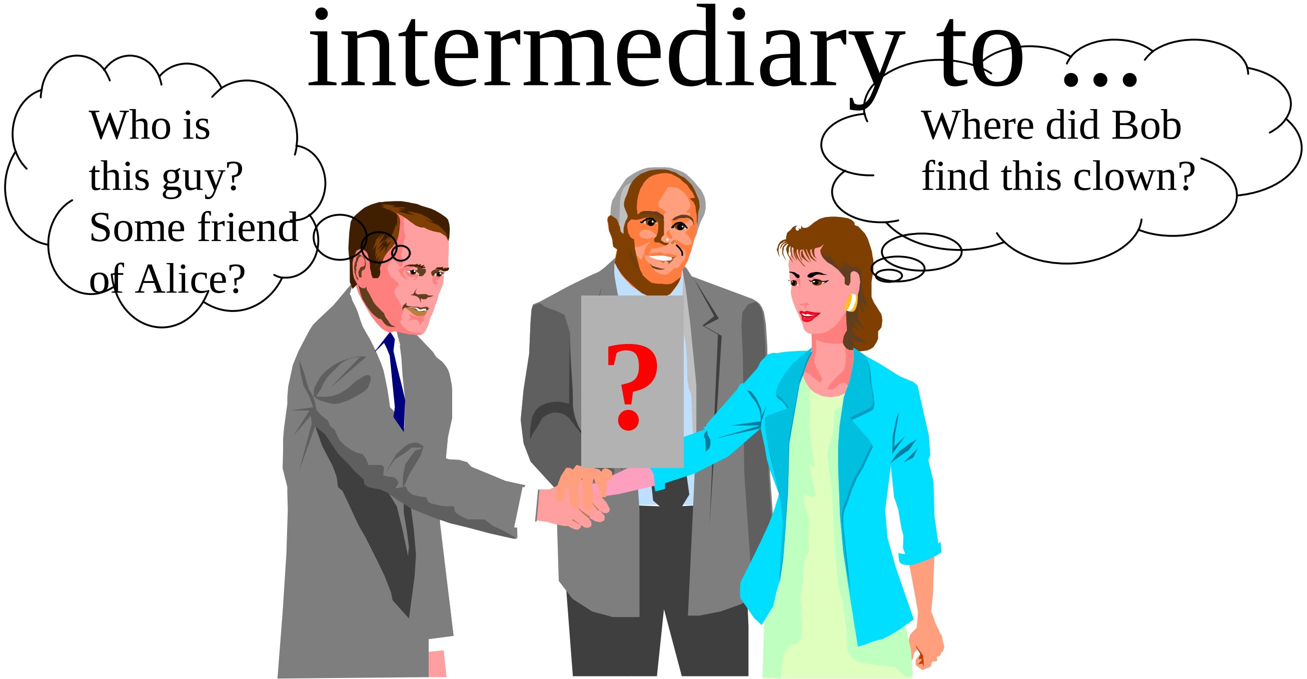
What is E2E Encryption?

- Alice and Bob have a *mutual* interest in keeping their communication ***confidential***
- Alice and Bob open a communication channel with
  - Mutual authentication
  - Encrypted data
  - Reason to believe that *only* Alice and Bob hold the symmetric keys
- Resolved, OPES will not compromise E2EE

If It’s Not E2E, What is It?

- Alice to Carol to Bob to Carol to Alice
- Alice and Bob trust Carol to keep their communication confidential
- Alice has an encrypted channel to Carol, Bob has an encrypted channel to Carol
- ***Hop-by-hop or link-level*** confidentiality
- Advantage: If Alice and Bob value Carol’s help, they can utilize it by trusting only her

Would you trust your OPES intermediary to ...


Who is this guy? Some friend of Alice?
?
Where did Bob find this clown?

- Question: is it sufficient for Alice to trust Carl? For Bob to trust Carl?
- Suppose Carl trusts Earl?
- Fact: The more parties, the less security

To Be Resolved

- Should OPES support concatenated confidential links?
- Must co-administered callout servers use encryption with an OPES intermediary?
- How to signal confidentiality requirements?
- How is delegation policy negotiated?
- Must all links be visible to and approved by Bob and Alice?

If Linked E2EE is Allowed...

- Need policy requirements
- Policy representation
- Policy configuration
- Signaling
- Prior art in hop-by-hop setup?
- Or … ?

And what about the callouts?

I've got no idea what to do here; I'm sending everything to Earl, my callout server
Who are Bob and Alice??
?

Multi-party Integrity

- Integrity is easier
  - You can delay the checks
  - With digital signatures, anyone can do the verification
  - No necessity to share secrets
- Channel integrity - SSL or Ipsec
- Message integrity
  - Complex policies with multiple delegations
  - Fine-grained control

Message Manifests

- Table of contents for a multi-part message
- Access control per part
  - Right: delete, replace, append, delegate
  - Allowed parties: identify by name, by key, etc.
- Modification actions appended to the manifest
- Signature over original message + mods
- Monotonic delegation (can only limit rights)

Policy Expression via Manifests

- Message addressed to principal
- No message content
- Describes messages to be subjected to policy
  - URL with wildcards
  - Modified by name principals
  - Containing delegation
  - Etc.

Manifests with OPES

- OPES intermediary can tell if message originator allows callout server action
  - Before sending a message or message part
  - After modification has occurred
- Callout server can determine if another organization can modify a message
  - Even if the callout server cannot!
- Receiver or agent can validate all changes