6.857 Rivest

**Admin:** Projects overviews due today  4/1/09 L15.1

Quiz on monday

(in class; coverage through today, your own notes & class notes (printed) are allowed, & handouts.)

**Outline:**

- Key establishment
- Direct
- Single Server – Symm
  – Asymm
- large-scale: X.509
  SPKI/SDSI
- cert revocation

6.857 Rivest
4/1/09 L15.2

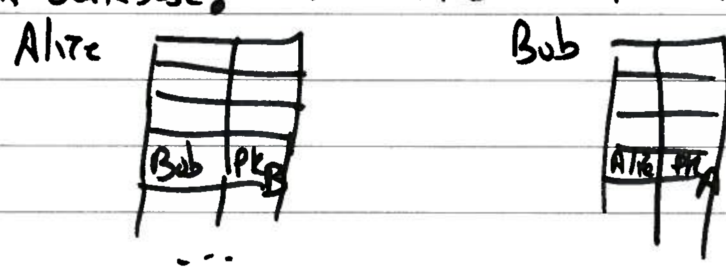
## Key management / key distribution

themes: crypto, keys, names, individuals, trust, identity, scaling, usability, certificates, PKI, trusted intermediaries

- keys need to be shared to be useful (at least PK part for PK crypto)
- how is such sharing to be arranged?

- Directly (by physical mtg)


meet in private (no eavesdroppers)
recognize each other (authentication)
share PK's, or symmetric keys
save in database: (when Alice has > 1 contact...)



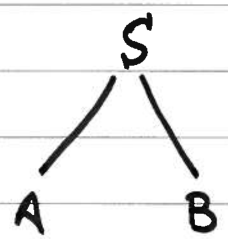

note appearance of names tied to entries...

privacy not needed if PK's are exchanged (as opposed to symm. keys)

Alice or Bob could be a computer
(e.g. Alice installs key in computer Bob,
or computer Alice gives user Bob her public key)

Such direct meetings are <u>necessary</u> <u>foundation</u> of key mgt, as we'll see...

## <u>Indirect / Two-link / TTP (trusted third party) or server</u>




Alice & Bob can't meet in person, but they have each met with trusted third party S & exchanged keys.

They can then request S to broker a "key-setup" operation so that A & B end up sharing a key, more-or-less as if they had actually met.

However, as we'll see, they need to trust S to behave properly & setup protocol (aka "key exchange" needs to be well designed).

Needham & Schroeder proposed 2 such protocols: one for symmetric keys, and one for public keys.

# NS Symmetric Protocol

$K_{AS}$ = key shared between A & S
$K_{BS}$ similarly for B & S
} already set up

"nonce" means a "use once" value

$N_A$ nonce generated by Alice
$N_B$ " " " Bob

could be from a counter, or a long random value...
Used to protect against certain forms of "replay attack"...

$K_{AB}$ : key that gets setup here between A, B (and S)

$\{M\}_K$ : message M encrypted & authenticated with key K
(e.g. encrypt M using AES in suitable mode & key $K_1$, append $MAC_{K_2}(M)$, where $K_1$ & $K_2$ derived from K)
(literature often is vague about properties of $\{\}$...)

**Protocol:**

① $A \rightarrow S: A, B, N_A$ — hi, I'm Alice, & I want to talk with Bob. $N_A$ is my "request nonce"...

② $S \rightarrow A: \{N_A, K_{AB}, B, \underbrace{\{K_{AB}, A\}_{K_{BS}}}_{\text{blob}}\}_{K_{AS}}$ — Oh here's key $K_{AB}$ & blob to give B

③ $A \rightarrow B: \{K_{AB}, A\}_{K_{BS}}$ — A knocks on B's door. B decrypts & checks blob

④ $B \rightarrow A: \{N_B\}_{K_{AB}}$ — B challenges A with $K_{AB}$

⑤ $A \rightarrow B: \{N_B - 1\}_{K_{AB}}$ — A responds

- who knows $K_{AB}$? A, B, S
- S must be <u>trusted</u>: can pretend to be A to B or vice versa...
- note roles of names: handles by which to identify parties
  addresses to which msgs can be sent
  text strings that can be included in messages
- if no nonces in ①, ② & cut out ④, ⑤: could replay earlier session to A or B
  (can do same if $K_{AB}$ later compromised...) → fix with <u>timestamps</u>
  (Kerberos)

## PK protocol

[$K_{PX}$, $K_{SX}$
public & secret keys of X

S has $K_{PA}$, $K_{PB}$
A & B have $K_{PS}$

6.857 Rivest
4/1/09 L15.6

① $A \to S: A, B$ — PK req

② $S \to A: \{K_{PB}, B\}_{K_{SS}}$ — signed "cert" for B's PK

③ $A \to B: \{N_A, A\}_{K_{PB}}$ — knock
encrypted, bound together
"psst! I'm A, and here's $N_A$ ... tell tone"

new inserted:

③' $B \to S: B, A$ — PK req

③'' $S \to B: \{K_{PA}, A\}_{K_{SS}}$ — signed "cert" for A's PK

④ $B \to A: \{N_A, N_B\}_{K_{PA}}$ — encrypted, bound together (non malleable...)

⑤ $A \to B: \{N_B\}_{K_{PB}}$ — yep, I'm really here...

- ③' & ③'' could be replaced by including blob/cert $\{K_{PA}, A\}_{K_{SS}}$ in ②
- at end only A & B know $N_A$ & $N_B$; eavesdroppers don't...

**Attack!** (Gavin Lowe) 17 yrs later!
automated analysis

intruder I gets A to initiate communication with I
- then passes knock $\{N_A, A\}$ on to B (after re-encrypting with $K_{PB}$)
- B responds with $\{N_A, N_B\}_{K_{PA}}$, which I sends to A
- A sends $\{N_B\}_{K_{PI}}$ to I. I decrypts it & gets $N_B$
- I sends $\{N_B\}_{K_{PB}}$ to B

Now B thinks he is sharing $N_A$ & $N_B$ only with A, but I knows $N_B$. WRONG

Fix: (4) $B \rightarrow A: \{N_A, N_B, B\}$

**Moral:** Be explicit in protocols!

(e.g. give session id both ways, & identities;

give hash of shared transcript in each message
(i.e. of all previous messages...)

Huge literature on such key-establishment protocols...