Berkeley CS276 & MIT 6.875

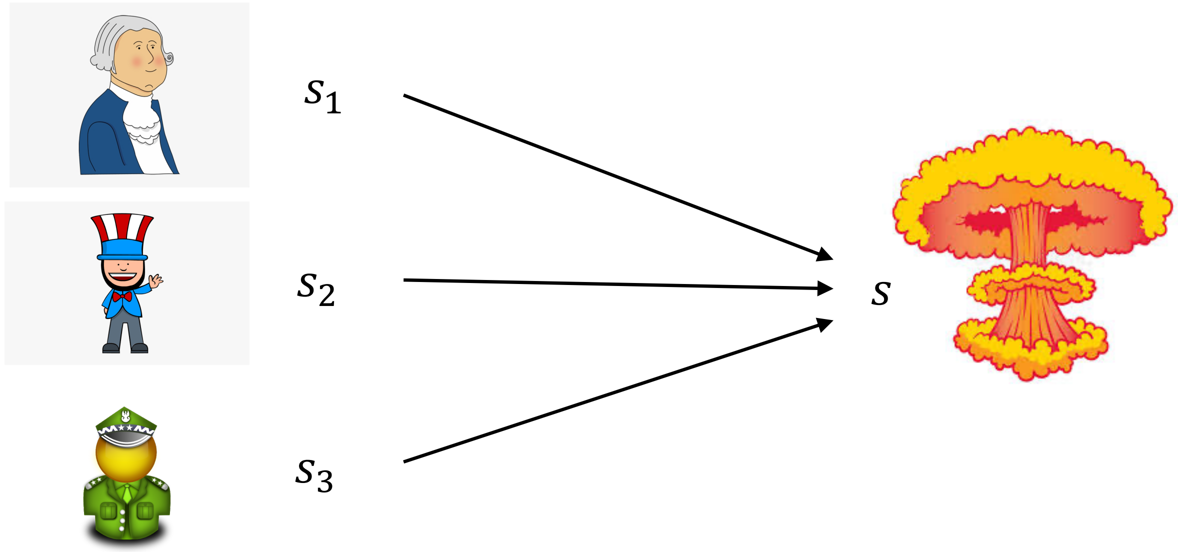
Secret sharing and applications

Lecturer: Raluca Ada Popa

- Starting to record

Nuclear launch codes

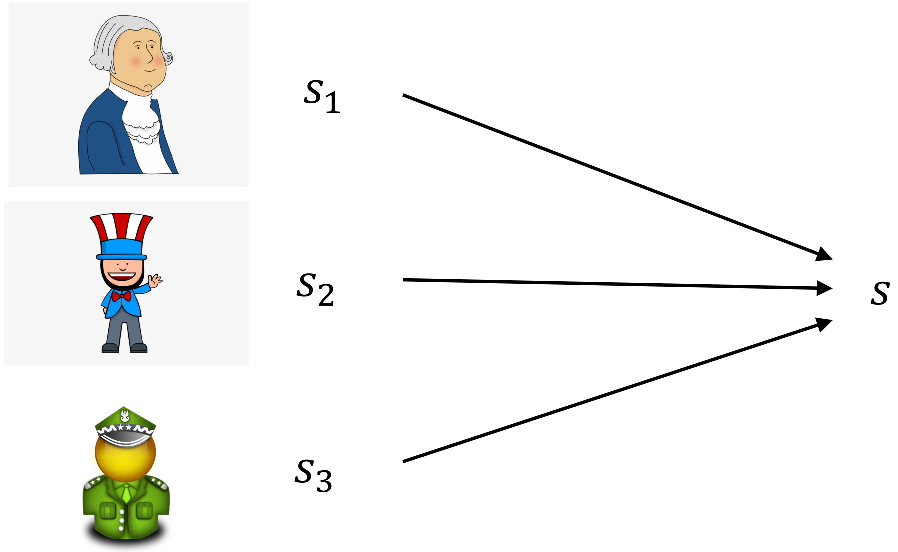
- A judge, president and general receive shares s_1, s_2, s_3 of a secret s from a **dealer.** If the nuke station receives s, it launches the nuke
- If any two come together, their shares together should reveal no info about s



Ideas?

Nuclear launch codes: xor secret sharing

- A trusted dealer chooses s_1 and s_2 randomly in $\{0,1\}^k$, and computes $s_3 = s \oplus s_1 \oplus s_2$
- The parties recover s by computing $s_1 \oplus s_2 \oplus s_3$



Nuclear launch codes: xor secret sharing

- A trusted dealer chooses s_1 and s_2 randomly in $\{0,1\}^k$, and computes $s_3 = s \oplus s_1 \oplus s_2$
- The parties recover s by computing $s_1 \oplus s_2 \oplus s_3$

Claim: nothing is learned about s from any one or two shares

$\Pr[s \mid \text{shares}\ ] = \Pr[s]$

(information theoretic security)

Nuclear launch codes: xor secret sharing

- A trusted dealer chooses s_1 and s_2 randomly in $\{0,1\}^k$, and computes $s_3 = s \oplus s_1 \oplus s_2$
- The parties recover s by computing $s_1 \oplus s_2 \oplus s_3$

$\Rightarrow$ 3-out-of-3 secret sharing

How about n-out-of-n xor secret sharing?

Similarly, choose $s_1, \dots, s_{n-1} \leftarrow \{0,1\}^k$, and set $s_n = s \oplus s_1 \oplus \cdots \oplus s_{n-1}$.

How about 1-out-of-3 xor secret sharing?

- Trivial: $s_1 = s_2 = s_3 = s$

Shamir secret sharing

- t-out-of-n: any t shares out of n shares can recover the secret
- Shamir, Adi (1979), "How to share a secret"

Syntax

Let F be a finite field of size P a prime number. Let 0<$t \leq n < P$.

A t-out-of-n secret sharing scheme for F is a pair of PPT algorithms (Share, Recover):

- Share($s \in \mathrm{F}$) outputs $s_1, s_2, \ldots, s_n$
- Recover ($s_{i_1}, s_{i_2}, \ldots, s_{i_t}$) outputs s

Correctness: $\forall s \in F, \forall\ s_1, s_2, \ldots, s_n$ ←Share(s), for any subset of t distinct indices $\{i_1, \ldots, i_t\}$ of size t:

Recover($s_{i_1}, \ldots, s_{i_t}$) $= s$.

Security

Given any $< t$ shares, absolutely nothing is learned about s.

Namely, the conditional distribution given the known shares for s should be the a priori distribution for s:

How would you formalize this?

$\forall v \in F, \ \ \forall$ distinct $i_1, \dots, i_{t-1} \in [1, n]$

$\Pr[s = v \mid s_{i_1}, \dots, s_{i_{t-1}};\ s_1, s_2, \dots, s_n \leftarrow \text{Share}(s)] = \Pr[s = v]$

Shamir's intuition

- t distinct points in F determine precisely one polynomial of degree $t-1$
- $t-1$ points could belong to an exponential number of polynomials of degree t in F

How would you design it?

Shamir's t-out-of-n scheme

Let $\alpha_1 \dots \alpha_n \in F$ be distinct non-zero elements known to all parties

Share(s):

- sample $a_1, \dots a_{t-1} \leftarrow F$ independently and uniformly at random.

- let $P(x) = s + \sum_{i=1}^{t-1} a_i x^i$

- for each i, set share $s_i = (i,\ P(\alpha_i))$

How to recover?

Shamir's t-out-of-n scheme

Let $\alpha_1 \dots \alpha_n \in F$ be distinct non-zero elements known to all parties

Share(s): $s_i = P(\alpha_i)$

Recover($s_{i_1}, \dots, s_{i_t}$):

- find a polynomial q of degree $t-1$ such that $q\left(\alpha_{i_j}\right) = s_{i_j}, \forall j$

- output s to be $q(0)$

Theorem: Shamir's scheme is correct and secure

Why?

Lagrange interpolation

Theorem: $\forall F$, $\forall t$ distinct values $x_1, \ldots, x_t$, and every t values $y_1, \ldots, y_t$, there exists a unique polynomial Q of degree at most $t-1$ s.t. $Q(x_j) = y_j \; \forall j$

Lagrange interpolation in our case:

$$Q(x) = \sum_{\ell=1}^{t} s_{i_\ell} \prod_{\substack{1 \le j \le t \\ j \ne \ell}} \frac{\alpha_{i_j} - x}{\alpha_{i_j} - \alpha_{i_\ell}}$$

Homomorphism of shares

Share(s): $s_i = P(\alpha_i)$

homomorphic?

Additively homomorphic, $s_i + s'_i$ is the i-th share for $s + s'$ because the Lagrange interpolation of the sum of the shares is the sum of $P(x) + P'(x)$ which evaluates to $s + s'$ for $x = 0$

$$P(x) + P'(x) = \sum_{\ell=1}^{t} (s_{i_\ell} + s'_{i_\ell}) \prod_{\substack{1 \le j \le t \\ j \ne \ell}} \frac{\alpha_{i_j} - x}{\alpha_{i_j} - \alpha_{i_\ell}}$$

How about xor secret sharing?

Shares of s are $s_1, \dots, s_n$ s.t. $s_1 \oplus \cdots \oplus s_n = s$

Homomorphic for XOR:

$s_i \oplus s_i'$ for $i \in [1, n]$ are shares of $s \oplus s'$

What are some problems with using Shamir secret sharing with malicious parties?

- If a participants cheats during recover, the wrong secret is recovered. The other participants cannot even tell this is the case.
- There is total trust in the dealer of the shares.
- The scheme is one time.
- The scheme only allows revealing the secret and not computing on it without revealing.

Verifiable Secret Sharing (VSS)

- The players can verify that their shares are consistent to some committed value
- Concept first introduced in 1985 by Benny Chor, Shafi Goldwasser, and Silvio Micali
- We will look at Feldman scheme based on Pedersen commitments

Setup

- Dealer publishes a commitment to s and to the polynomial used for the shares $P(x)$
- The dealer signs all messages it sends to the parties
- Each party receiving a share s_i can check their share against the commitment

Construction

Recall Pedersen commitments:

$comm(x) = g^x h^r \in G$ for r random, g, h public

Dealer publishes a commitment to s and to the polynomial used for the shares $P(x) = s + \mathrm{a}_1\mathrm{x} + \cdots + a_{t-1}x^{t-1}$

$comm(s) \ = g^s h^{r_0}, comm(a_1), \ldots, comm(a_{t-1})$, signed by the dealer

Let $R(x)$ be the polynomial with the randomness from these commitments

$R(x) = r_0 + r_1 x + \cdots + r_{t-1}\mathrm{x}^{\mathrm{t}-1}$

Construction

The dealer gives to each party i: its share $s_i = P(\alpha_i)$ and $R(\alpha_i)$.

Party i checks that

$$comm(s) * comm(a_1)^{\alpha_i} * \cdots * \mathrm{comm}(\mathrm{a}_{\mathrm{t}-1})^{\alpha_i^{t-1}}$$

equals

$$g^{P(\alpha_i)} h^{R(\alpha_i)}$$

Security properties

- What malicious behavior of a dealer would this prevent?
- What malicious behavior of the parties would this prevent?
- What malicious behavior would it not prevent?

Security properties (informally)

- Parties know their shares are consistent (different subsets of t shares will reveal the same value s), or prove misbehavior of the dealer.
- Party i can check that it indeed received share i
- Upon reveal, a party can check the share of another party, so a malicious party cannot affect a reveal
- Dealer can still commit to s of its choice
- Still assumes trusted setup of public parameters?

Why do the security properties hold? (informally)

- Parties know their shares are consistent (different subsets of t shares will reveal the same value s), or prove misbehavior of the dealer
  - Because the commitment is binding
- Upon reveal, a party can check the share of another party
  - Because the commitment is binding

What is the hiding property used for?

Secrecy of the secret sharing scheme

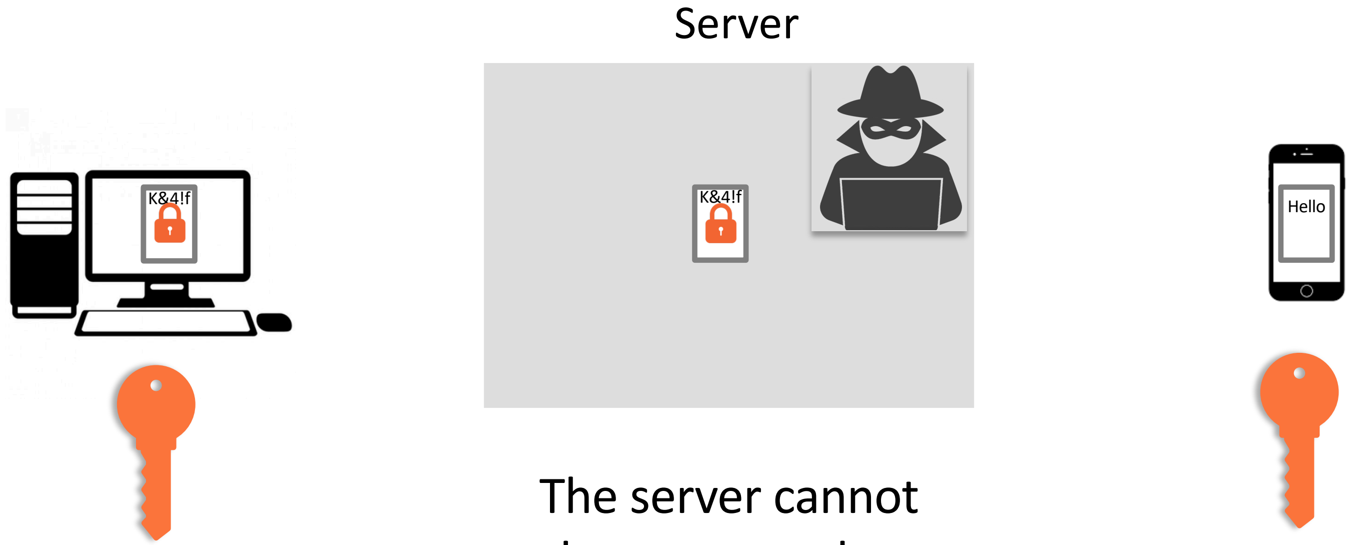
Applications

What applications come to mind?

Applications

- Key recovery for end-to-end encryption
- Custody of secrets for cryptocurrencies

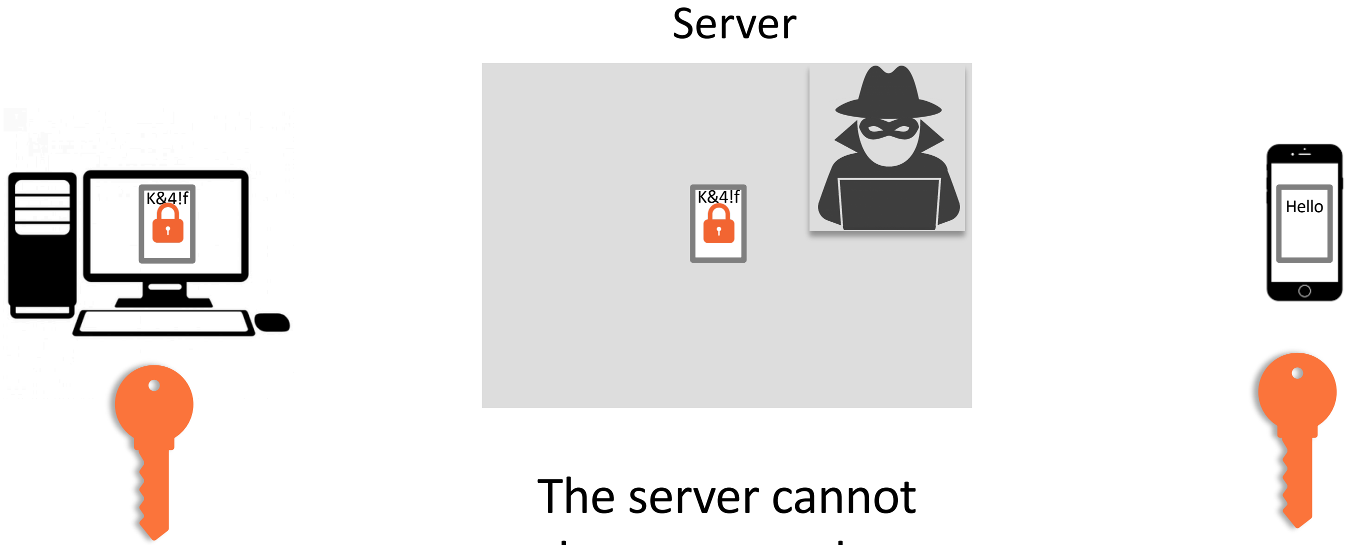
End-to-end encryption



The server cannot decrypt user data

The server is not a central point of attack, does not have the decryption key

End-to-end encryption

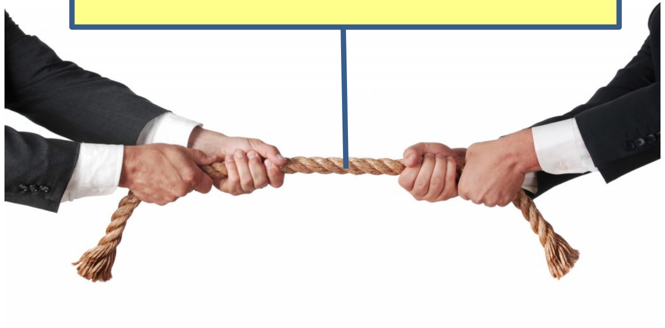


The server cannot decrypt user data

Key recovery is challenging

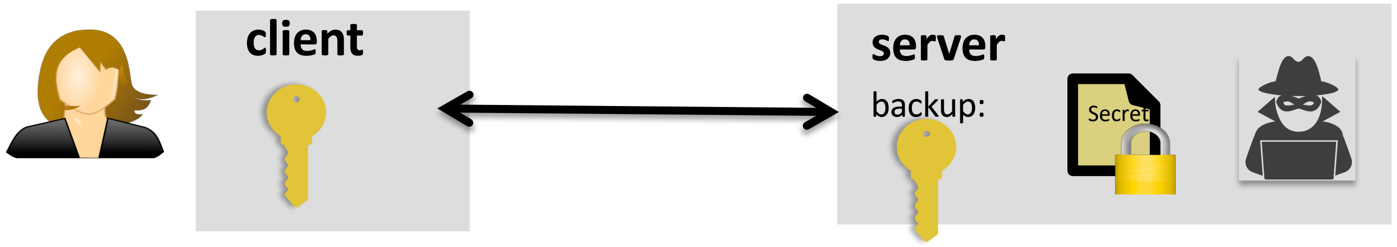
key recovery

usability



end-to-end security

Key recovery challenge



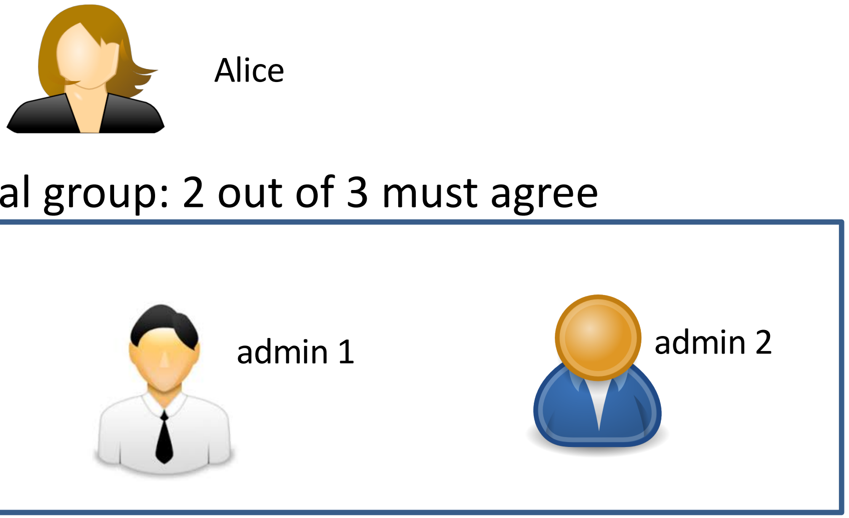
Usability issue: If Alice loses her key, she loses access to her data (e.g., PGP)

Security issue: Existing solutions prefer to compromise on security: save key at the server!

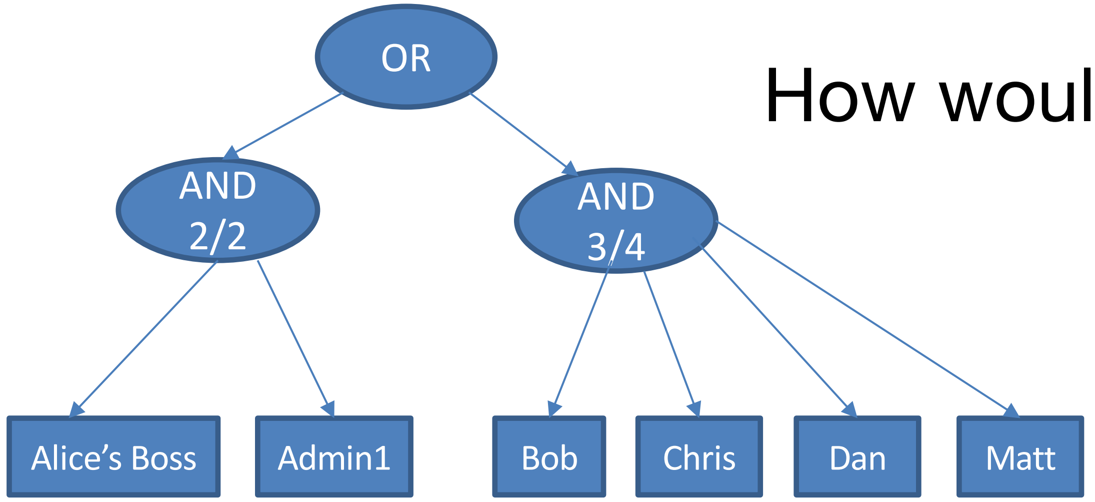
Ideas?

Secret sharing keys

- Each user chooses a set of trusted users who can reconstruct her lost key
- None of the users in the group can reconstruct the key by itself



Richer policies for organizations



How would you implement this?

Cryptocurrency application

- We saw that secret keys control assets in Bitcoin, Zcash, etc.
- Need to back these secret keys to prevent asset loss
- Ideas?
  – Some crypto custodians offer secret sharing

TREZOR

Ledger

Cryptocurrency application

Even storing the secret on the user device that performs payments is worrisome, so newer technologies store the secret key secret shared and sign transactions using it by recovering the secret from shares “under encryption” – we will learn how to do this in secure-multi party computation

curv

Another huge application of secret sharing

Summary

t-out-of-n secret sharing enables splitting a secret into n shares such that any less than t cannot reconstruct any information about the secret, but t shares can determine the secret

Secret sharing schemes can be homomorphic

They have many real-world applications