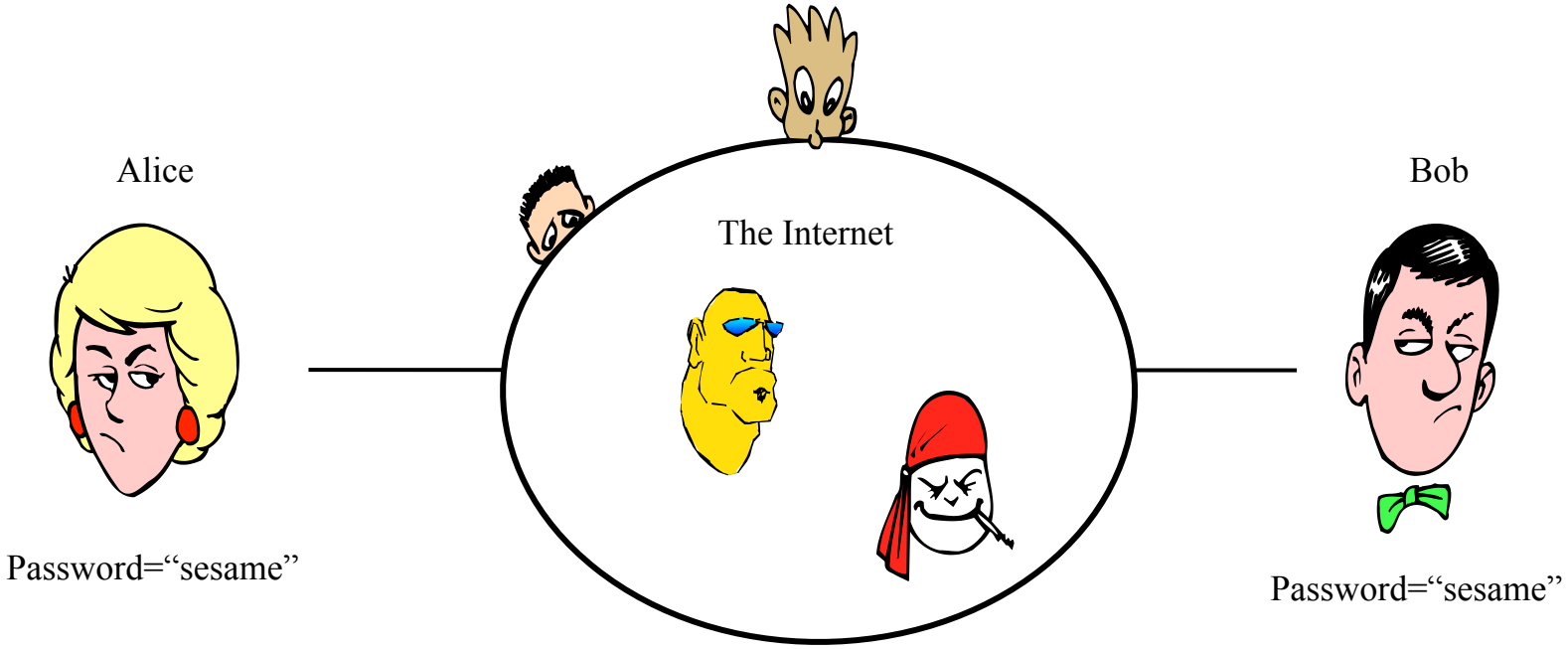
Authenticating People and Machines over Insecure Networks

EECE 571B “Computer Security”

Konstantin Beznosov

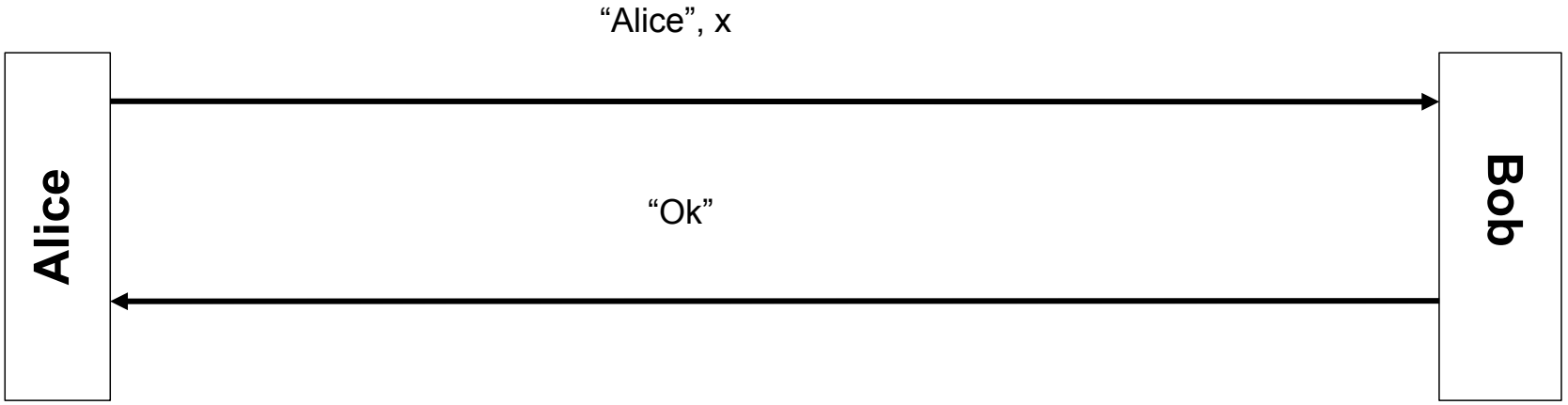
authenticating people

objective

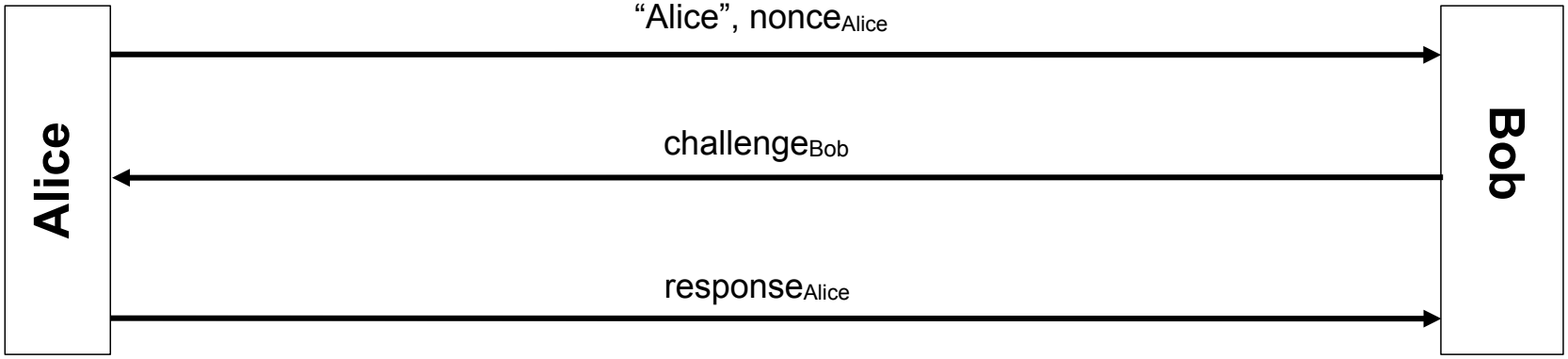


- authenticate Alice to Bob over insecure network

simplistic approach (attempt #1)



general challenge-response protocol



password password

$response_{Alice} = f(challenge_{Bob}, password)$

How can it be attacked? offline dictionary attack on eavesdropped messages!

What else? plaintext-equivalent!

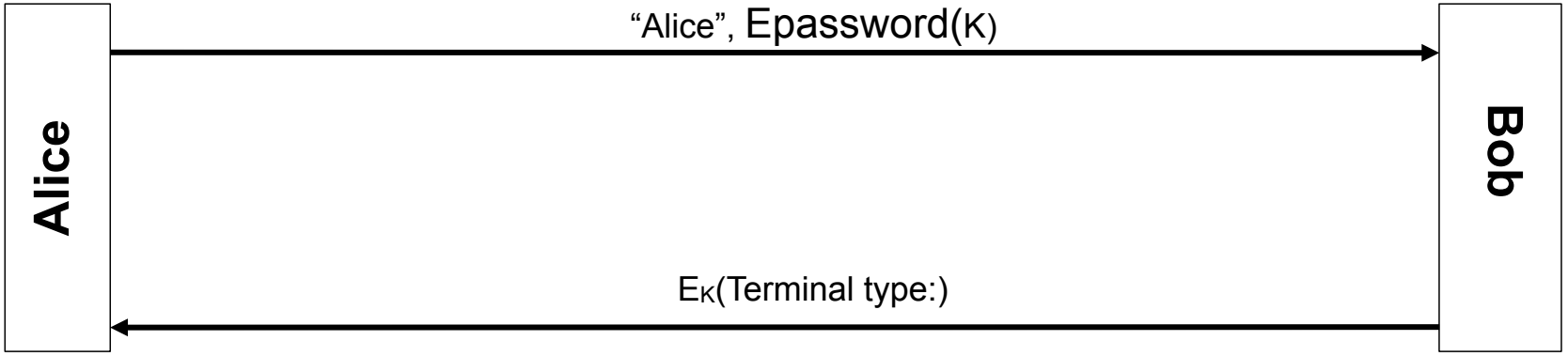
desirable properties

- mutual authentication
- session key
- resistant to dictionary attacks
- server compromise does not make it easy to find password
- password compromise does not lead to revealing past session keys (forward secrecy)
- session key compromise does not lead to password compromise
- does not take long

another view of PAKE

"a means of "bootstraping" a common cryptographic key from the (essentially) minimal set up assumption of a low-entropy, shared secret"

attempt #2



password

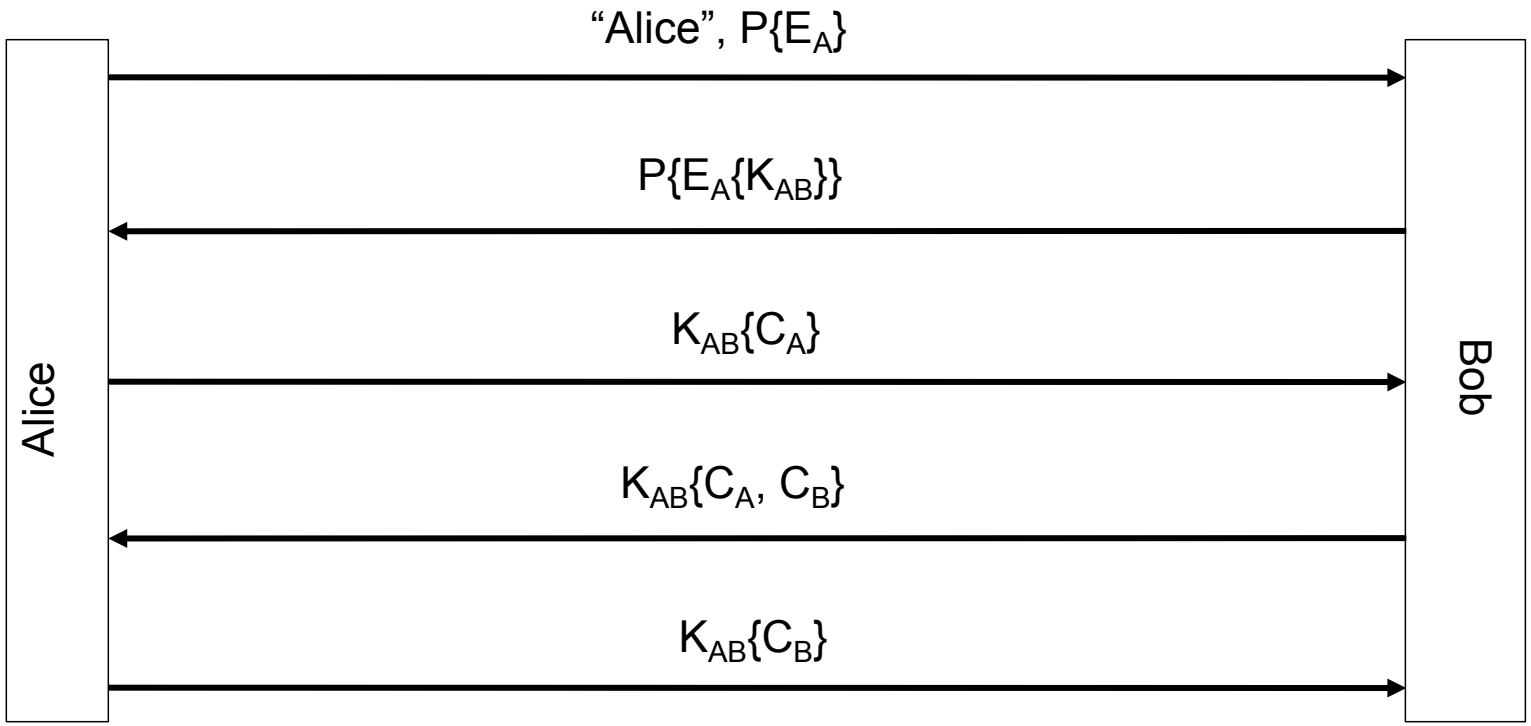
password

K -- random session key generated by Alice

How can it be attacked? offline dictionary attack on eavesdropped message from Bob!

What else? replay attacks

Encrypted Key Exchange (EKE)

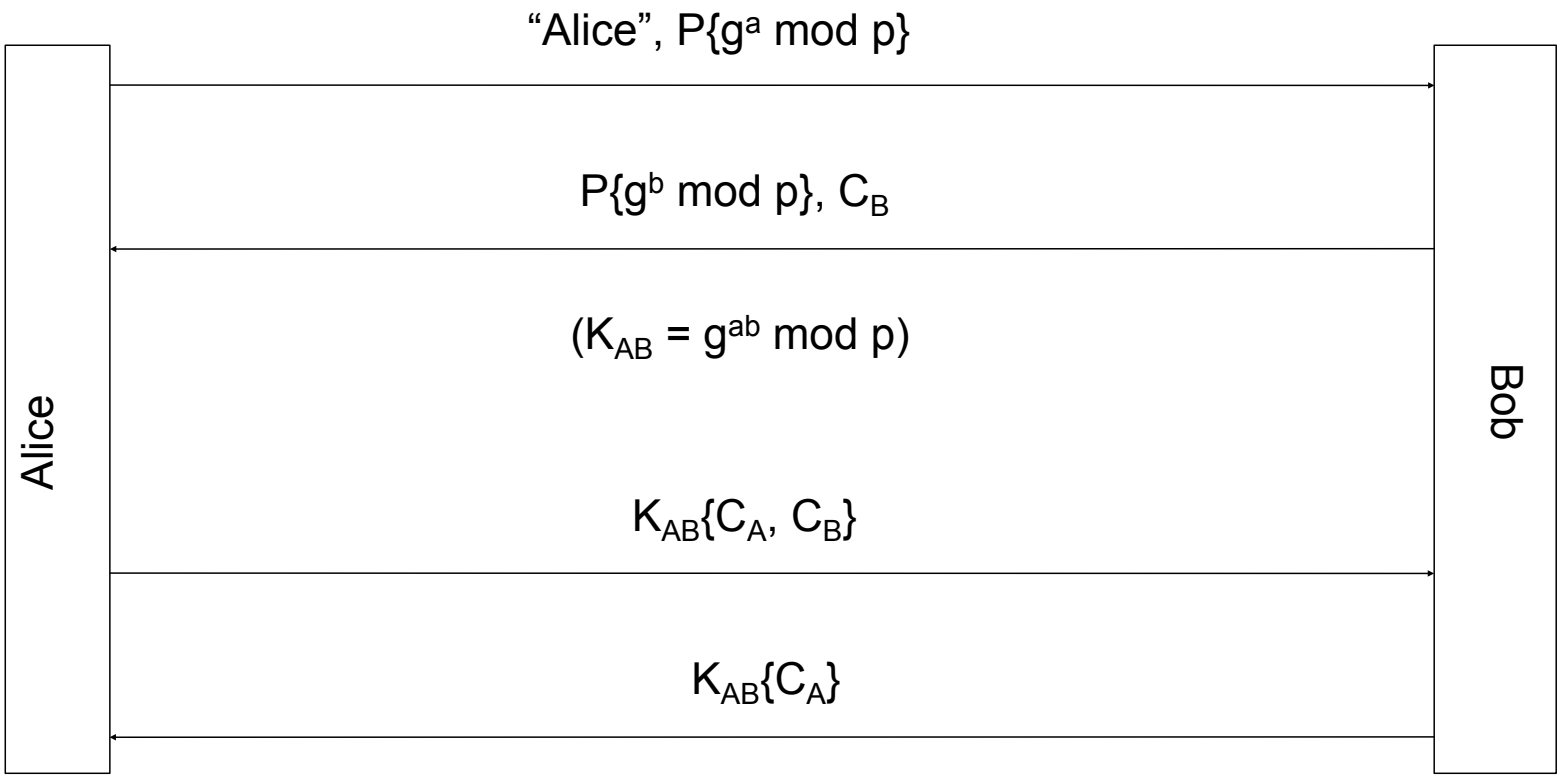


plain text for encryption with password P must look random

more on EKE

- assumptions
  - encryption must not leak any useful information
  - for all P', $P'^{-1}\{P\{E_A\}\}$ must appear a valid public key
- strengthening EKE
  - what if a session key K_{AB} has been recovered?
  - $S_{AB} = f(S_A, S_B)$

EKE with Diffie-Hellman



Why are g^a and g^b encrypted?

desirable properties

- mutual authentication
- session key
- resistant to dictionary attacks
- server compromise does not make it easy to find password
- password compromise does not lead to revealing past session keys (forward secrecy)
- session key compromise does not lead to password compromise
- does not take long

EKE properties

- ✓ mutual authentication
- ✓ session key
- ✓ resistant to dictionary attacks
- - server compromise does not make it easy to find password
- - password compromise does not lead to revealing past session keys (forward secrecy)
- ✓ session key compromise does not lead to password compromise
- - does not take long
  - - public key crypto is expensive

Asymmetric Key Exchange (AKE)
&
Secure Remote Password (SRP) Protocol

AKE/SRP features and idea

- generalized form of a class of verifier-based protocols
  - no plaintext-equivalence
- does not encrypt protocol flows

idea

each party

- computes a secret
- applies one-way function to it to generate a verifier
- sends its verifier to the other party
- both parties generate session key from secrets and verifies

SRP notation

n : A large prime number. All computations are performed modulo n.

g : primitive root modulo n (often called a *generator*).

s : A random string used as the user's salt.

P : The user's password.

x : A private key derived from the password and *salt*.

v : The host's password verifier.

u : Random scrambling parameter, publicly revealed.

a, b : Ephermeral private keys, generated randomly and not publicly revealed.

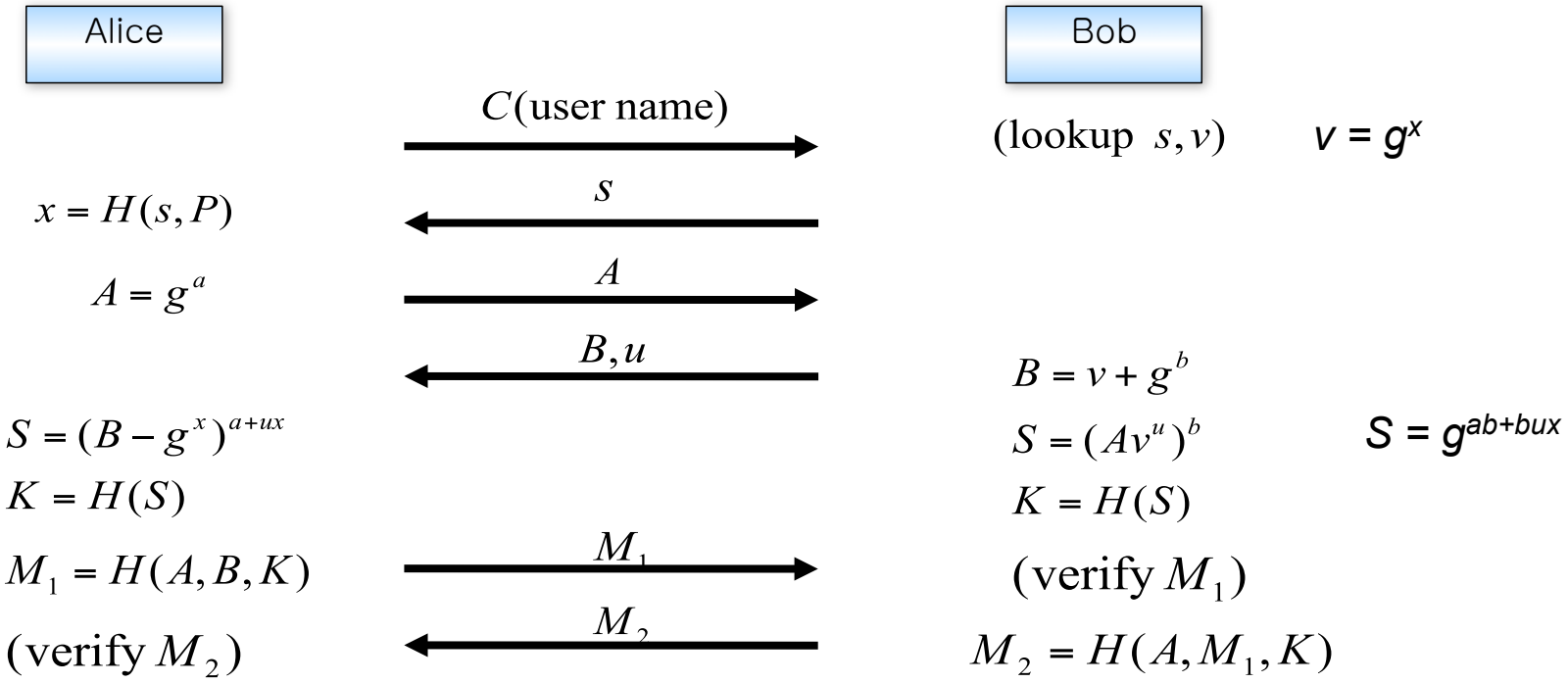
A, B : Corresponding public keys.

$H()$: One - way hash function.

K : Session key.

SRP protocol

To establish a password P with Bob, Alice picks a random salt *s*, and computes *x* and *v*. Provides Bob with *s* and *v*.



SRP demo

http://srp.stanford.edu/demo/

optimizing SRP message rounds

$C \Longrightarrow S$	C
$C \Longleftarrow S$	s
$C \Longrightarrow S$	A
$C \Longleftarrow S$	B
$C \Longrightarrow S$	M_1
$C \Longleftarrow S$	M_2

original

$C \Longrightarrow S$	C, A
$C \Longleftarrow S$	s, B
$C \Longrightarrow S$	M_1
$C \Longleftarrow S$	M_2

optimized

$C \Longrightarrow S$	C, A
$C \Longleftarrow S$	s, B
$C \Longrightarrow S$	M_1

one-way authentication optimized

SRP properties

- ✓ mutual authentication
- ✓ session key
- ✓ resistant to dictionary attacks
- ✓ server compromise does not make it easy to find password
- ✓ password compromise does not lead to revealing past session keys (forward secrecy)
- ✓ session key compromise does not lead to password compromise
- ✓ does not take long
  - public key crypto is expensive

Decentralized User Authentication in a Global File System

"Professor" David Mazières

Bio, Vita

Classes

Photo credit: Lori Bode, Squink Industries

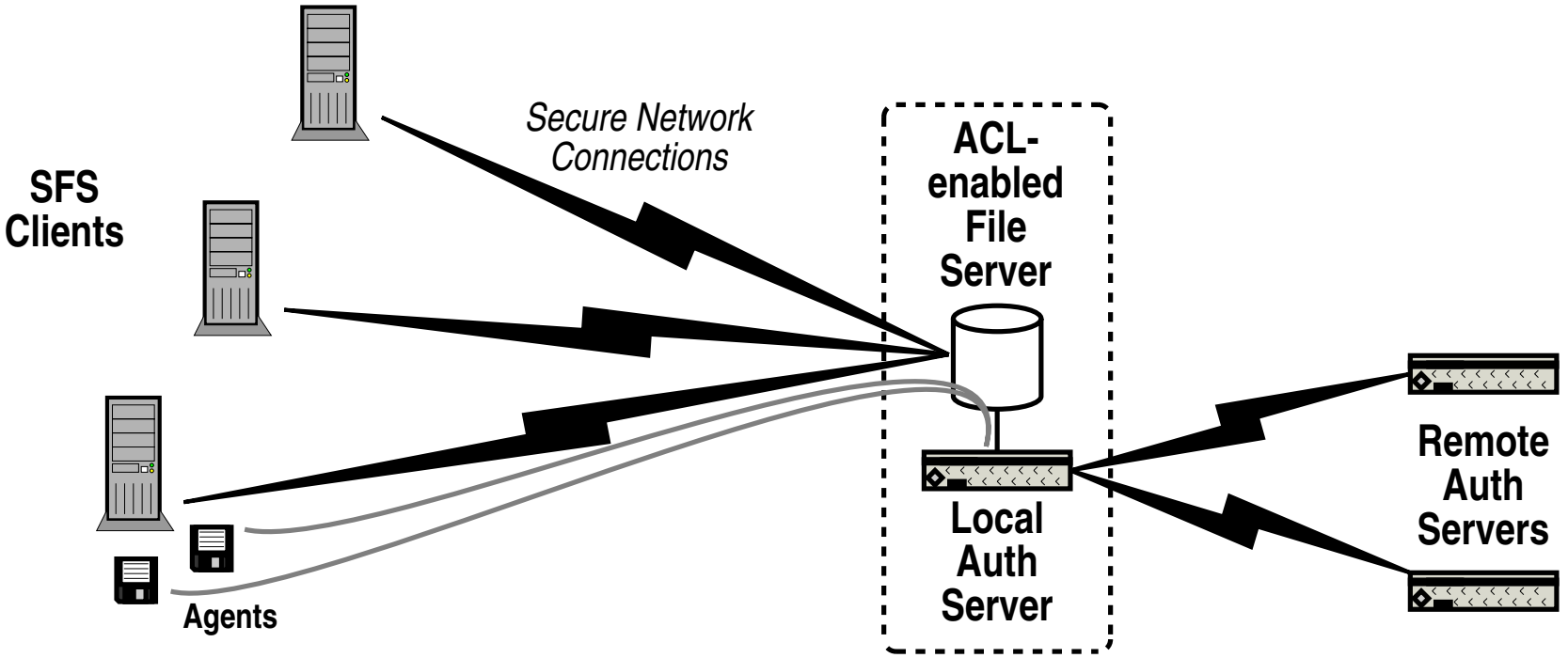
Papers

Email

Contact

Secure Computer Systems group

architecture



Goals

- Authenticate users to access the file system
- Support remote administrative domains
- Use only local information at access time
- Avoid certificates

Why not certificates?

- Complicated infrastructure
- Certificate chain hard to compute (e.g., SDSI)
- Or inflexible trust structure (e.g., VeriSign)
- Overkill for a file system?

SFS Servers

- Each server has a public key
- Key part of the name ("self-certifying")
  - `mit.edu,anb726muxau6phtk3zu3nq4n463mwn9a`
- Use key to authenticate server and set up a secure connection
  - Connection provides confidentiality & integrity

Self-Certifying Names

- Public keys are explicit
  - Always together with the name
- No PKI necessary
  - Avoids organizational and technical issues
- Keys are obtained out-of-band
  - Perhaps falling back on people

Authentication Servers

- One server per administrative domain
  - Identified by self-certifying hostname
- Authenticate users
  - Unix passwords, public keys, SRP, …
- Manage local names and groups
- Export user and group records to remote servers

Groups

- Defined within an administrative domain
- Has a list of members and a list of owners
- Each user may define their own groups
  - E.g. alice.friends
- Members/owners can be remote or local

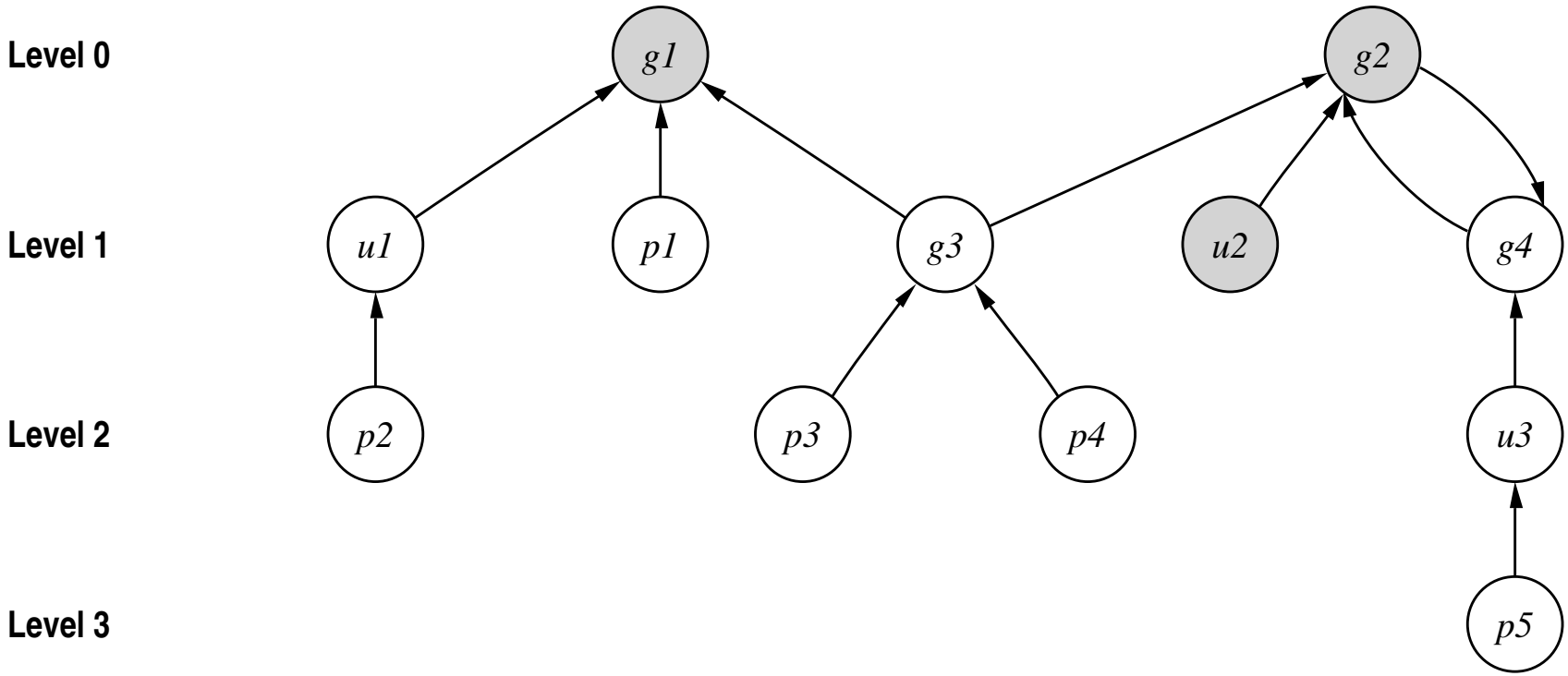
Group members

Member type	Example
Local user	U=beznosov
Local group	G=beznosov.571B-students
Remote user	U=billg@microsoft.com,wxyweq…
Remote group	G=faculty@cs.ubc.ca,r34qduk…
Public key	P=d43dft5tr50lkxsdre42…

Group members

- Local users & groups
  - As defined by the authentication server
- Public key hashes
  - Allow ad-hoc users
  - Protect privacy
- Remote users & groups
  - Retrieved from remote servers
  - Authenticity protected by self-certifying name

membership graph example



Group Caching

- Group definitions may be distributed on many servers
- Each authentication server resolves and caches entire group membership
- Cache ensures all necessary information is locally available at time of access
  - Though it may be out of date

Resolving Membership

- Expand group names
- Query remote servers for group & user definitions
- Recursively query any new remote names
- Cache updated every hour
- Use version numbers to send deltas

Problems

- Freshness
  - Eventual consistency
  - Use out-of-date data for an hour
  - Longer if server unavailable
- Revocation
  - Easy to revoke users (with a delay of 1 hour)
  - Hard to revoke server keys

Scalability

- **All** relevant group members cached on local server
- students@berkeley.edu may be large
- registered-voters@gov.bc.ca wouldn't work
  - It *would* work with certificates
- Limit members to 1,000,000 to prevent DOS
- Most sharing groups are small
  - 571B-students
  - ece-registered_students

ACLs

- Each file and directory has an ACL
  - Stored in first 512 bytes
- Lists local users and groups and access rights
  - Read, write, modify ACL
- Remote names and public keys have to be indirected through a group
  - Save on space
  - Easier to change membership

user record in ACL

User Name	Public Key
ID	Privileges
GID	SRP Information
Version	Audit String

group record in ACL

Group Name	Owners
ID	Members
Version	Audit String

Certificates Revisited

- What did we lose?
  - Human-readable namespace
  - Key management/revocation
  - Offline operation
  - Scalability
- Are these not important for a global FS?

credits

These slides incorporate parts of the following:

- “Decentralized User Authentication in a Global File System” presentation slides from CS294-4, Stanford, N. Borisov, 2003-10-06.
- “The Secure Password-Based Authentication Protocol” by Jeong Yunkyoung.
- Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attack, Bellovin and Merritt (IEEE S&P 1992).
- The Secure Remote Password Protocol, T. Wu (NDSS 1998).