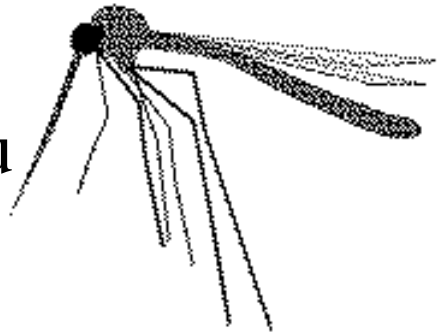
Communications for Mobile People

Mary Baker

mgbaker@cs.stanford.edu

http://mosquitonet.stanford.edu



MosquitoNet Group

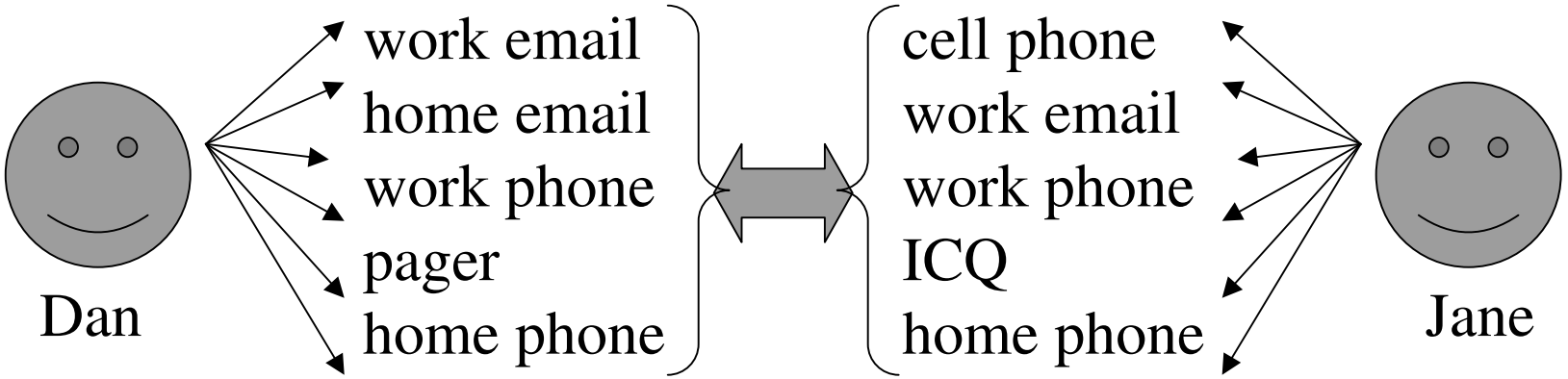
Departments of Computer Science and Electrical Engineering

Stanford University

Mobile people

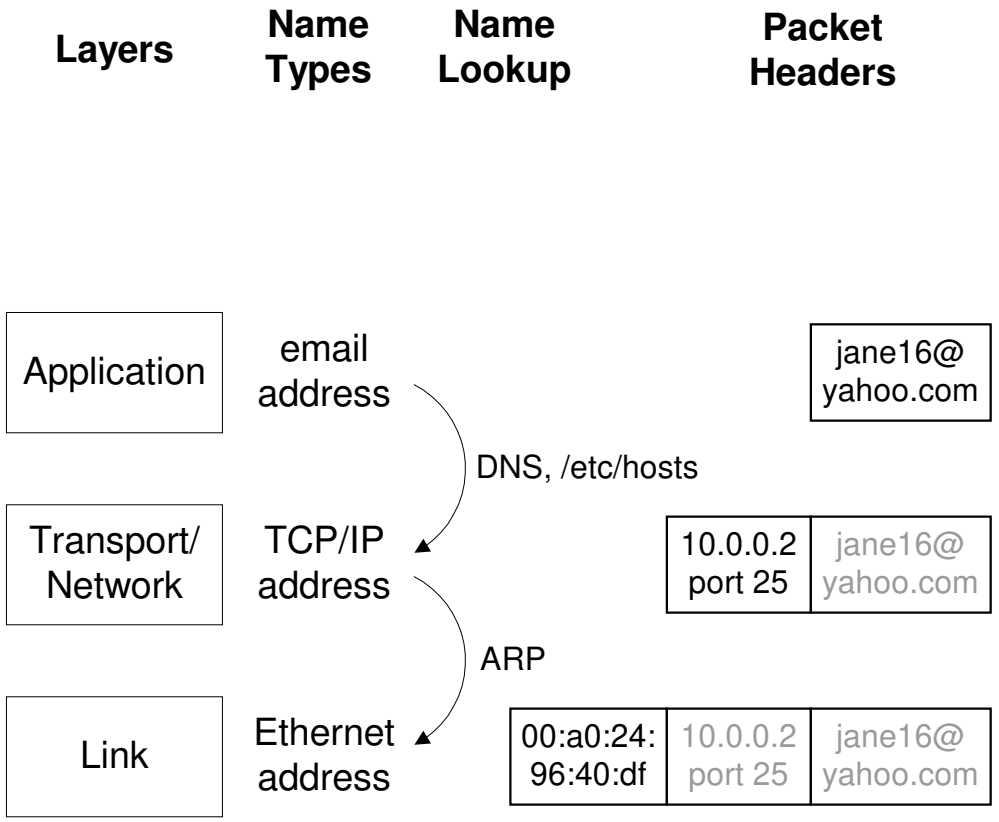
- *Mobile people* move between different applications, different devices, and different roles
  - Desktop PC, laptop, PDA, cell phone, pager, hotel phone
- Why do they do this?
  - One device does not serve all purposes
- Much previous mobility work provides "anywhere/anytime" connectivity to a single device
- People are the true endpoints of much communication

Problems



- On what device do I reach a mobile person in a timely manner? (Mobile People Architecture)
- How do I name mobile people as endpoints, rather than their devices? (IdentiScape)

Current network model



Each layer provides

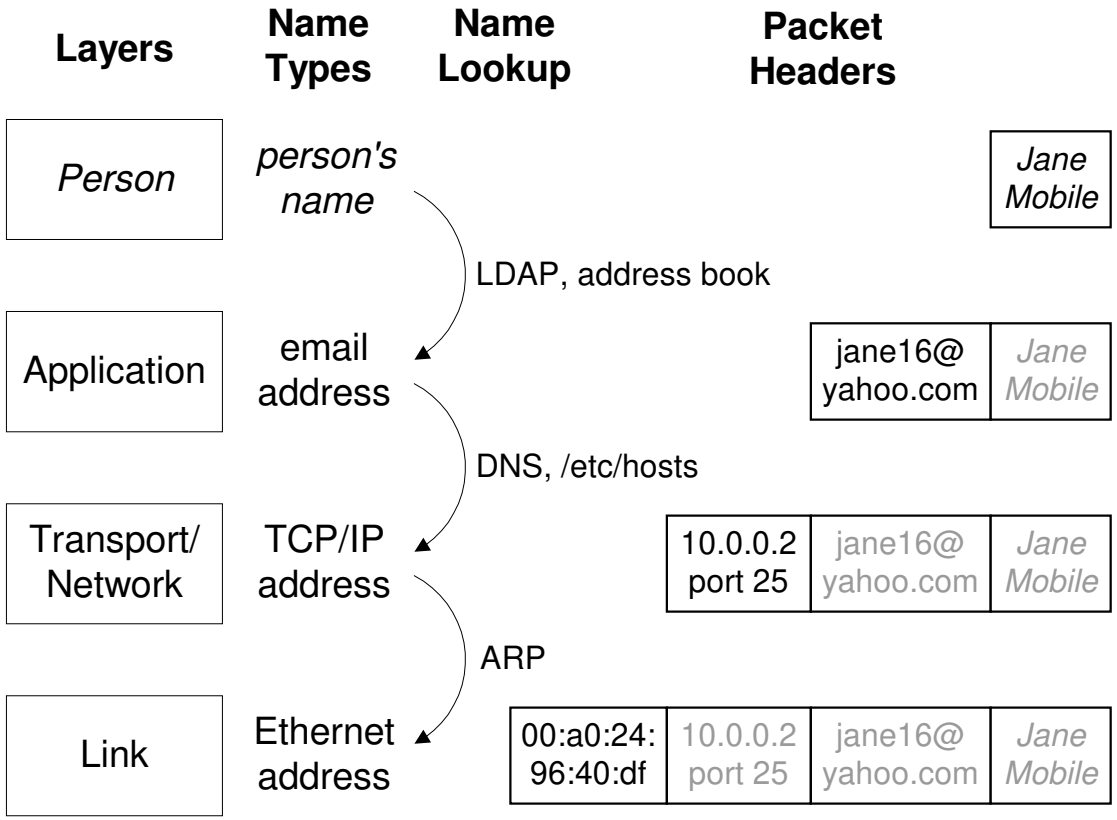
- Routing
- Naming
- Mapping to layer below

Problem: there's no layer with people as endpoints

Solution: extend model

Extend network model to incorporate people

- New person layer
- People are communication endpoints



Person layer requirements

- A way to route communications between people
  - Person-level router
  - Mobile People Architecture
- A naming scheme to identify people uniquely
  - Personal Online ID (“IdentiName”)
  - IdentiScape project
- A way to map to application-layer names
  - IdentiName => application-specific addresses
  - IdentiScape project

Mobile People Architecture (MPA)

- Routing communications between people
- Make it possible to reach mobile people easily
  - Anytime
  - Anywhere
  - On any communication device
  - From any communication device
  - With receiver controlling nature of communication
  - While maintaining receiver privacy

Motivation – receiver control

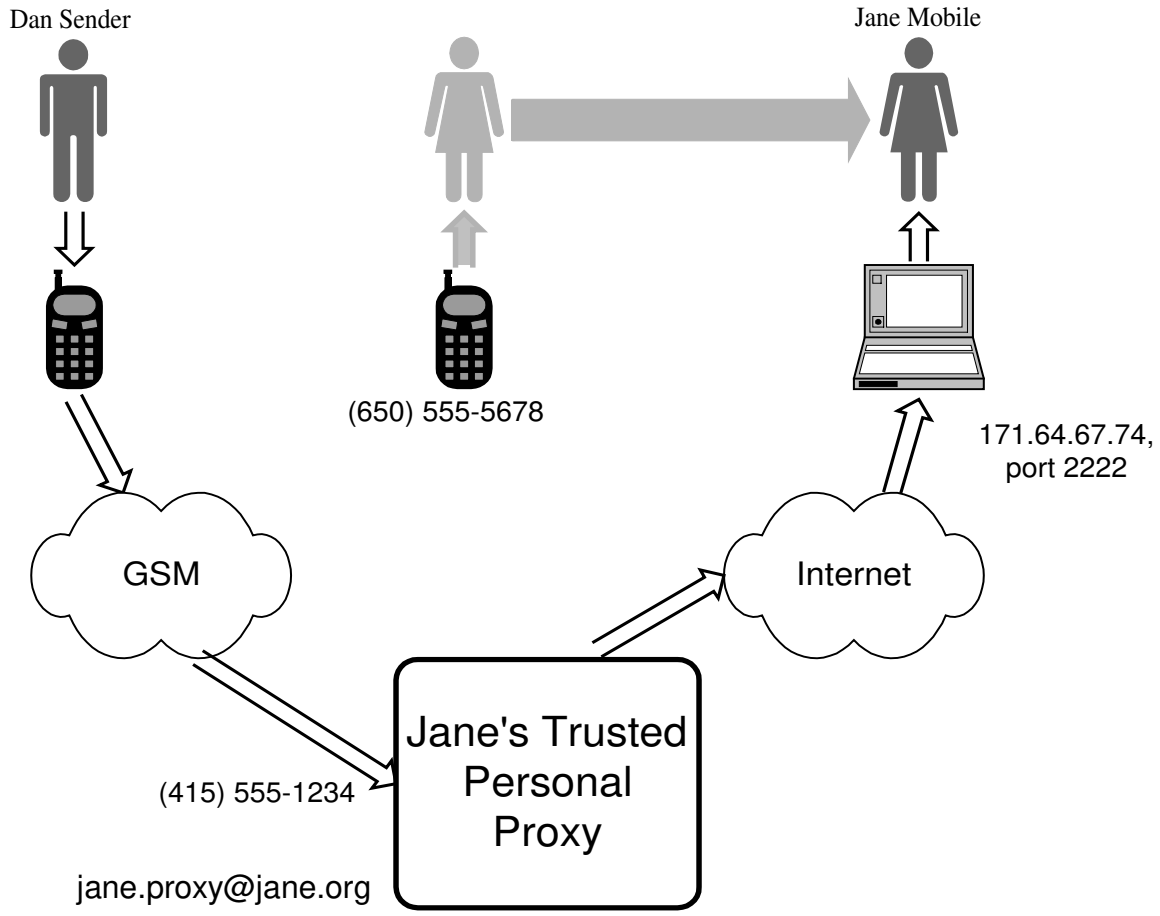
- Currently sender controls how/where/when messages sent
  - Sales calls at home during dinner
  - Email spam
  - Useless pages
- *But as a recipient, I want control over my reachability*
  - Only calls from the daycare center can go to my pager
  - No phone calls while I'm having dinner
  - No more "Make money fast at home!!!!" email
  - Would like to handle these issues in one place if possible

Motivation – privacy

- Sender may be able to infer receiver's location from the address or phone number that actually works
  - Email address me@myoffice.com
  - Home phone
- Once I give out direct addresses, I can't revoke them
  - I can only change them
  - Filtering callers must now be done for each application/address
- *But as a recipient, I may want to keep my location or direct addresses a secret*

Personal proxy as person-level router

- Naming: Dan only knows Jane's name
- Mapping: Dan's phone uses Jane's name to look up her Proxy phone # & calls her there
- Routing: Her Proxy converts call to email & sends it to Jane's laptop

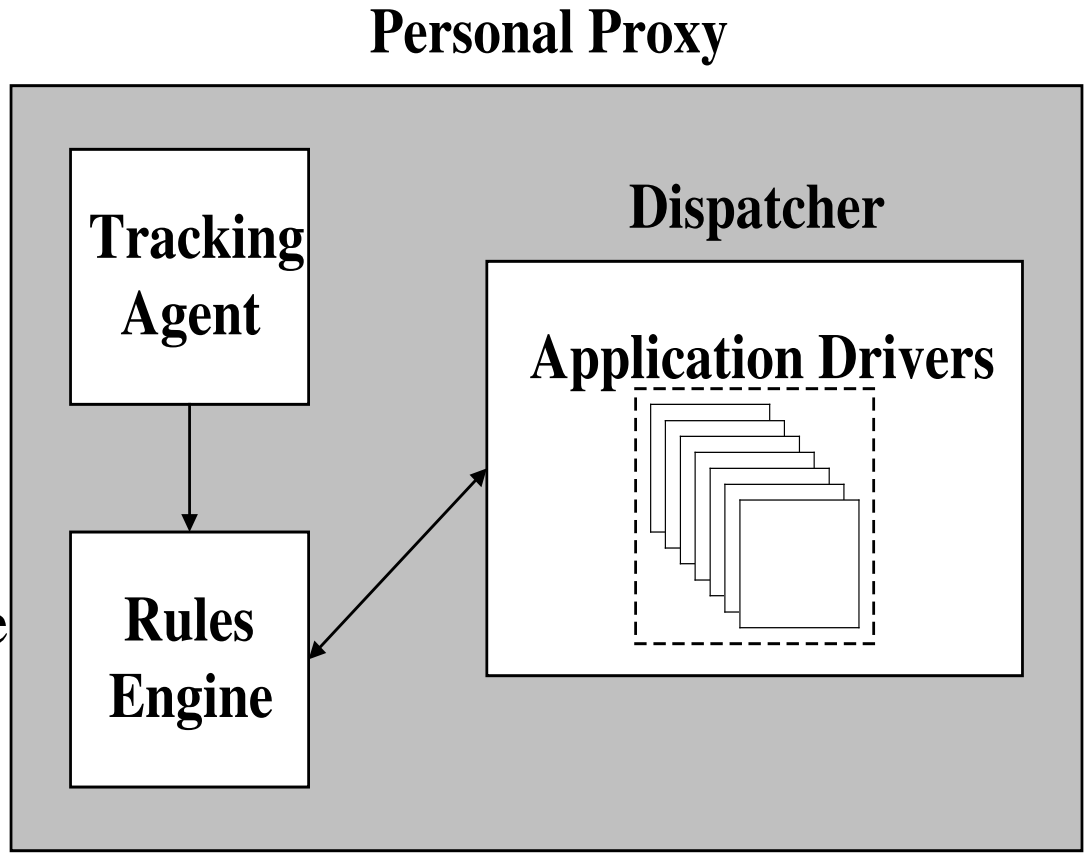


Personal proxy design philosophy

- “Personal” service implemented at the edge of the network (near the person)
- Scalability
  - Set top box (or PC) at home
  - Hosted at an ASP
- Trust
  - Sensitive data & functions located where user chooses
  - User knows what components are involved
- Deployment
  - Does not require changes to infrastructure of network

Personal proxy design

- Tracking Agent tracks receiver moving between devices/applications
- Rules Engine implements filtering preferences
- Dispatcher converts and routes communications to the mobile person using Application Drivers



Tracking agent

- Tracks mobile person's current connectivity state
  - Application-specific addresses
  - Communication formats that can be handled at those addresses
- Via registrations
  - Automatic (with varying granularity)
  - Manual
  - Preset schedule
  - Combinations of these

Rules engine

- Passes directives to the Dispatcher on how to route a particular communication
- Uses current connectivity state and user preferences
- User preferences stored in form of rules

Rules

- Rule = (condition, action)
- Conditions
  - Is this from daycare?
  - Does this contain “Make money fast”?
- Actions
  - Send it to my pager
  - Drop it
  - Truncate to 50 bytes

Dispatcher

- Dispatcher is the routing component of Personal Proxy
- Uses directives from Rules Engine to convert & route communications
- Consists of plug-in application drivers
- Uses a goal-based planner to find a path through conversion drivers
  - Currently just breadth-first search

Prototype evaluation

- Deployment
  - Can be one server, set up by one individual
  - No need to modify the underlying infrastructure
  - Useful to individuals without need for global adoption
- Location privacy and data security
  - As secure as the Personal Proxy
- Thwarting spam
  - As effective as email filters (procmail)
  - But supports application-independent rules
- Extensibility
  - Plug&play driver framework, drivers queried for their abilities
  - No need to bring down system to install new drivers

Related work

- UMTS, TOPS, etc.
  - Often no location privacy
  - Not set up for true any-to-any communications
- Wildfire, uReach.com, etc.
  - Limited scope of applications
- Iceberg (UC Berkeley)
  - Underlying infrastructure changes
  - Larger sphere of trust
  - Iceberg paths more efficient
  - Iceberg has better extensibility (easier to share components)

IdentiScape goals

- Easily name people online
- Name maps to
  - Contact information for personal proxy
  - General contact information
  - Other stuff people want
- Reduce contact information management problems
  - Avoid update of other people's copies of our contact info
  - Contact other people reliably
  - Name reuse issues
  - Name change issues
  - Name robustness issues

Naming problems

- Name reuse
  - Defunct pizza parlor phone number reassigned to Jane
  - Jane gets lots of pizza orders
- Name changes
  - Email from Jane's lawyers arrives at Jane's old address
  - Old address controlled by party she's now suing
- Name robustness
  - Your address/number is too similar to someone else's

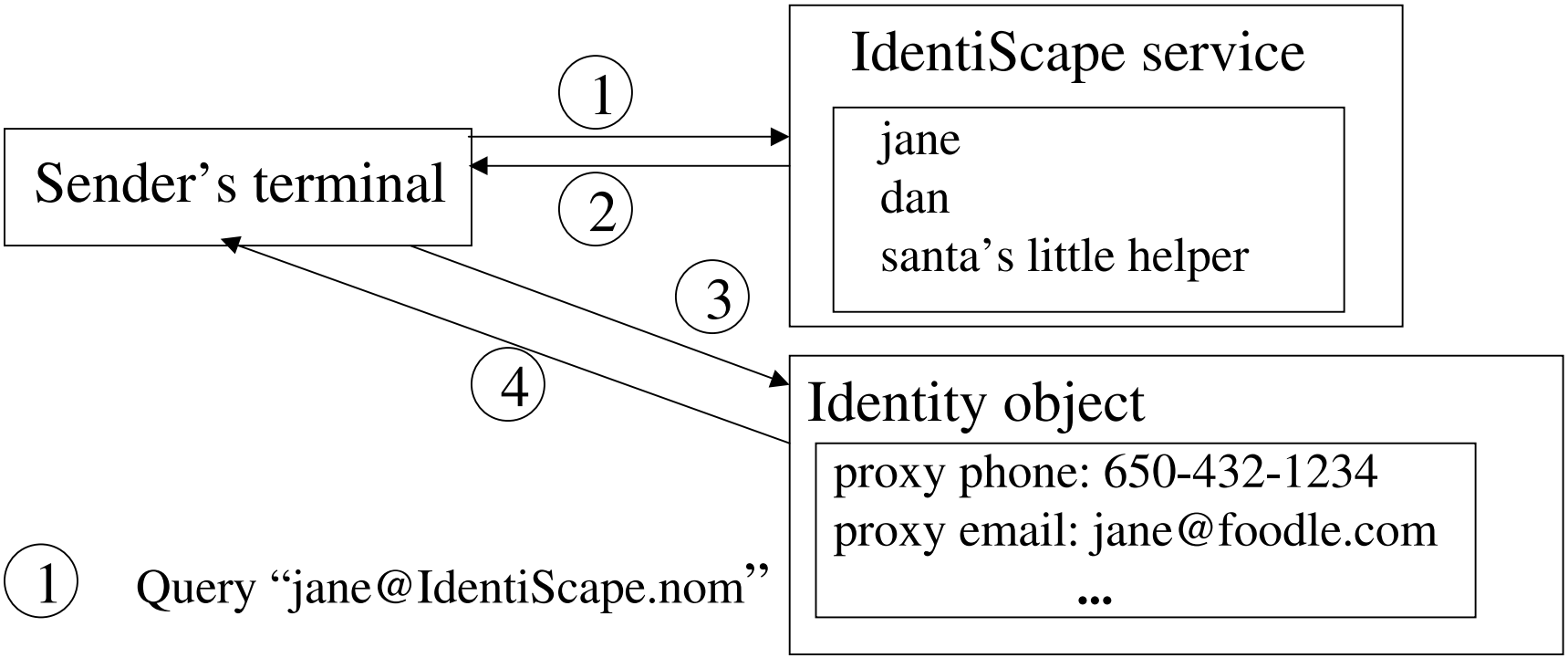
Idealized naming service attributes

- Ubiquity
  - I can have the same name everywhere
  - I can transfer my names over different media
  - My names don't give out private information
- Human-centricity
  - I can define/change my name
  - My name is "manageable" by humans
- Robustness
  - My name is not similar to anybody else's
  - It is easy to catch simple typos in a name
- Persistence
  - My names are valid as long as I want them to be
  - I control what my old names point to

IdentiScape solution

- Naming service(s) that
  - Allow callers to use one identifier to reach a person
  - Provide robustness of names
- Directory services (identity object services) that
  - Enable people to control the contents and accessibility of their own online identity information
- Separation of naming and directory information
  - Scalability
  - Trust

IdentiScape Architecture



1. Query "jane@IdentiScape.nom"
2. Return: address of identity object
3. Query identity object
4. Return: contact information

Scalability issues

- IdentiScape service just provides a pointer to identity object
  - Information changes infrequently (cacheable)
  - Adds delay (but name to pointer is cacheable)
- Identity object service
  - Scalability requirements usually less stringent
  - Can be very privately managed (on your home PC)
- Useful to individuals even if not widely deployed

Mix and match architecture

- Can use IdentiScape without MPA
  - For managing names and contact information
- Can use MPA without IdentiScape (give out proxy addresses)
  - For timely contact
  - For receiver control over communications
  - For privacy
- Identity object may be collocated with personal proxy
  - Identity object allows personal proxy to move
- Time scales of IdentiScape/MPA information differs
  - IdentiScape information changes more slowly
    - On order of changes to business cards
  - Personal proxy deals with changes on finer time scale
    - I'm at office phone now
    - In five minutes I'm only available by PDA

Persistence problem

- Involuntary name changes inevitable
  - IdentiScape.nom goes out of business
  - I forget to pay my bill to IdentiScape.nom
- People will use (leak) names from other name spaces
  - These names are used within organizations
  - These names are used with reference to organizations

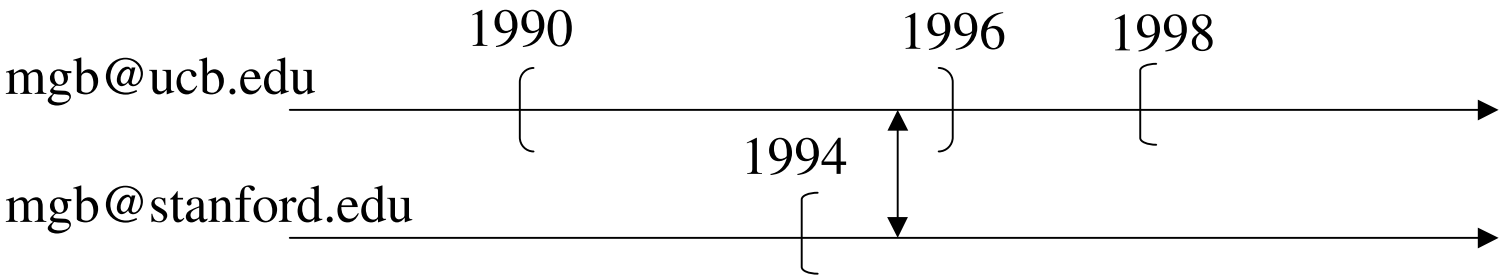
Solutions to persistence problem?

- Solution: global service with flat namespace?
  - Single “ownership” or unpleasant names?
  - Who will trust it?
  - Someone else will start one too
  - Doesn’t solve name leakage
- Solution: global coverage by independent name services?
  - Doesn’t provide organization-independent names
- Solution: name history service
  - Given (old name,date), look up current name
  - This could be implemented in a peer-to-peer manner
  - Participants are entities with interest in such history

History service

- Authenticated list of name transitions
  - Signed by name holder
  - Time stamped
- "Persistence" and control over old names
  - You'll reach me with my old name if you run it through history service
  - Nobody else can prove they used that name at that time
  - Name space manager cannot retract existence of old name

Example use of history service



- In 1990 mgb gets a name from UCB
- In 1994 mgb gets a name from Stanford
- After 1994 name change inserted
- In 1996 Berkeley removes mgb name
- In 1998 another mgb gets a name from UCB
- In 2050 user queries service: Current name (mgb@ucb.edu, 1992)??
  - Returns mgb@stanford.edu

Problems

- Who provides the keys?
  - Assume PKI for name services (similar to DNSSEC)
  - Local name spaces handle public key services within their spaces
- Who runs the history service?
  - Need a censorship resistant global archive
  - Archived documents are self-secured (preserve their own integrity)

  $\Rightarrow$ Long-term archival of signed documents
- Longevity of signed documents?
  - Old signed documents need old verification keys
  - Was signature produced during validity period of key?

  $\Rightarrow$ Need old key archival and secure time stamping

KASTS

- Like a notary public [Haber et al., 1995]
- Secure time stamping service (TSS)
  - Establishes time when a digital document is signed
  - Time stamp the signature when it is produced
- Archival of signature verification keys (KAS)
  - Allows users to request and receive correct signature verification key for a signer at any time in the past
  - Stores signed certificates from certificate authority (CA)
  - In particular, stores CA's master verification keys
    - Typically self-certified certificates
    - Originally distributed through a secure channel

Centralized time stampers

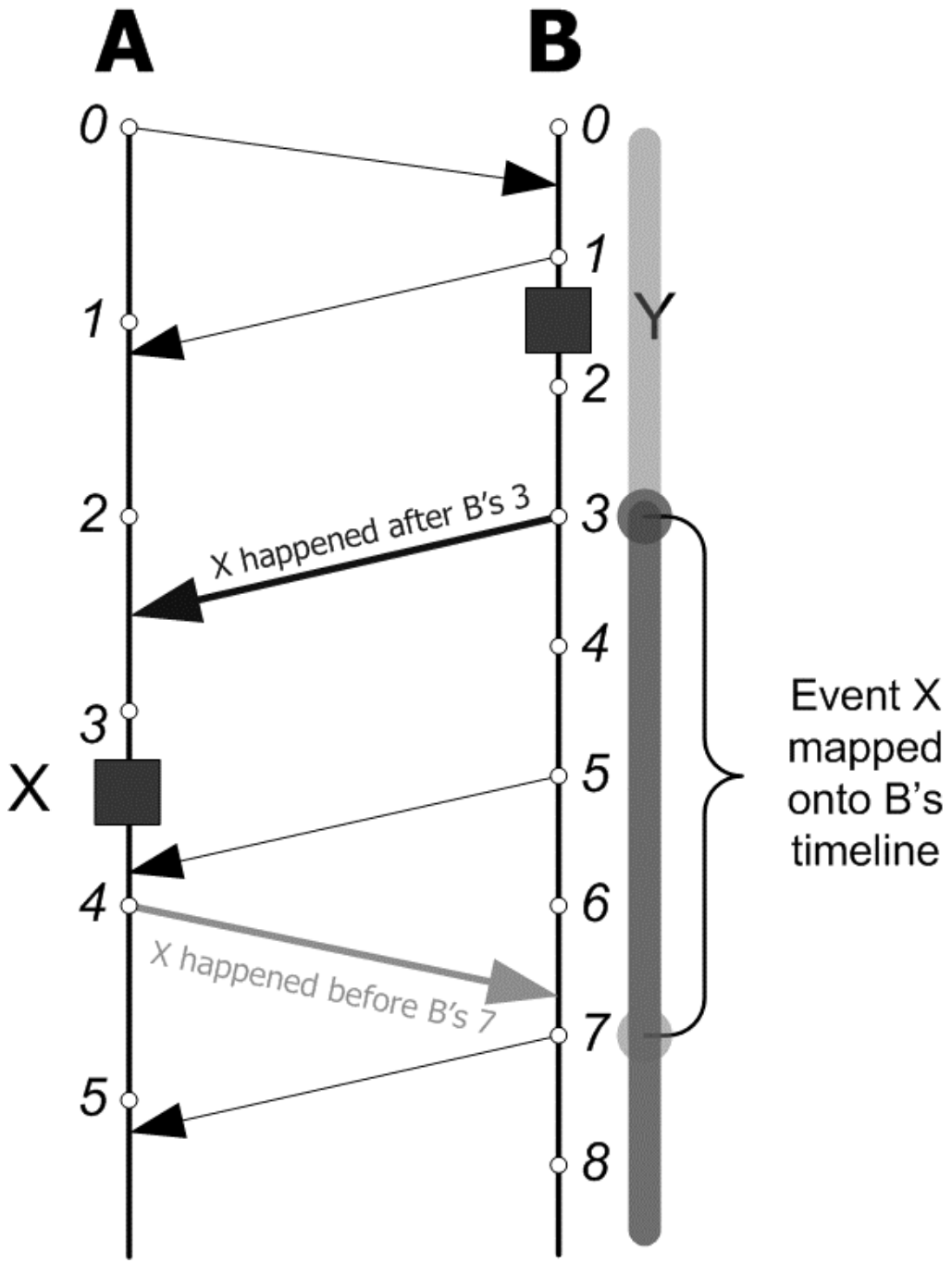
- Surety is an example [www.surety.com]
- Build up tree of documents signed during a round
  - "Root hash" represents the ordered set of leaves of the tree
  - Based on collision-resistant hash functions like SHA1
- Time stamp of digest is
  - Time at which round was created
  - Proof of inclusion of digest in the linking data structure
- Result of a "round" represented as a hash
  - Published independently (provides accountability)
    - Widely distributed
    - Write-once
  - This hash used as input to next round

How to use multiple TSSes

- People will use the TSSes they trust
- How do we verify time stamps from other TSSes?
- Distributed peer-to-peer system of TSSes?
  - Replaces publication medium through agreement
  - Uses Byzantine fault-tolerant techniques for agreement over time stamps and group membership
  - Potentially survives complete change in membership *over time*
  - Expensive
    - For 150 nodes, round change can take 30 hours in the worst case
    - Comfortable for some human-scale time granularities
    - Key revocation: 2 weeks is reasonable
  - Prokopius

Timeline entanglement

- Timeweave [Usenix Security 2002]
- Give up global consistency of event ordering
- Use group of TSSes that application task involves
- Link (entangle) past of one timeline into future of another


A
B
0
0
1
1
Y
2
2
3
X happened after B's 3
3
4
3
X
5
4
6
X happened before B's 7
7
5
8
Event X
mapped
onto B's
timeline

Timeline entanglement characteristics

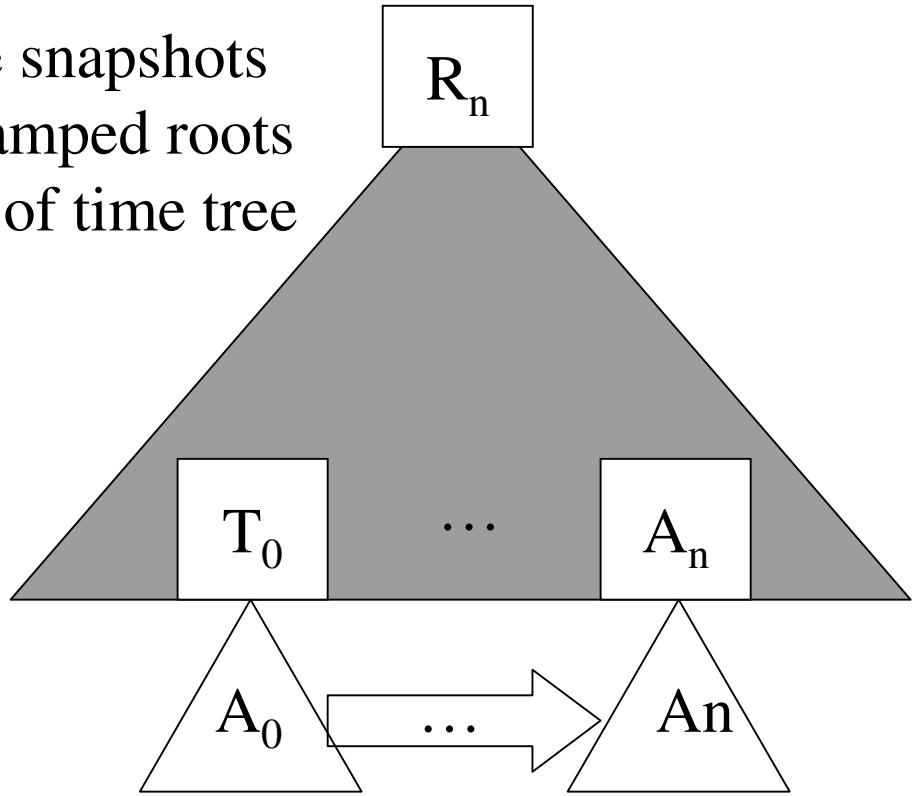
- Can survive demise or non-cooperation of originating service
  - Must have some service you still trust, though
- Less expensive – depends on
  - Number of entangled services
  - Rate of entanglement
  - For up to 1200 PCs, 10-minute entanglement, maintenance ranges between 2-8% of processing resources

Key archival service

- Maintains timed history of signature verification keys
  - Most notably the master verification keys used and published by CAs
- Accumulates key updates and revocation information
- At end of round key archive is modified and time stamped to reflect changes
  - Use hash trees to represent “time stampable” snapshots of CA
  - Uses authenticated search trees for accountability [Buldas et al., 2000]
- Snapshot roots archived in a Time Tree
  - Also an authenticated search tree
  - Ordered by time

Time tree

A's = archive snapshots
T's = time stamped roots
R_n = nth root of time tree



Related work

- Most similar to IdentiScape goals
  - Specialist Task Force 157 of European Telecommunications Standards Institute
  - Charged with finding “personal identifier of the 21st century” they combine name with public key
- OneName.com
  - They run the directory service as well as provide the account name
  - No help with name reuse or robustness issues
- Centralized time stamping services such as Surety
  - Require trust of single organization
  - What happens when they go out of business?
- LOCKSS (Lots of Copies Keep Stuff Safe)
  - Long-term archival of documents (doesn’t handle signing issues)

Conclusions

- People are the true end-points of much communication
  - Mobile communications should reflect this
- More support needed to integrate mobile communications into our lives
  - Increase receiver control of communications
  - Privacy is important
  - Ease of use is important
- Services at “edge” of network
  - Easier deployment
  - Users gain benefits without global adoption
  - Personal services close to person