Architectural Issues in Distributed, Privacy-Protecting Social Networking

Leonard N. Foner

MIT Media Lab

foner@media.mit.edu

http://foner.www.media.mit.edu/people/foner/

Outline

- Introduction
- System Architecture
- Privacy and Security
- The Sample Application: Yenta
- Evaluation

Introduction

- Historical perspective
  - Vaporware (CFP '95) to first release ('99)
  - Java? IM? Ha! Barely even early web...
  - Pre-file-swapping P2P
  - Copyright wasn't the enemy---ITAR was
- But years later, lessons still unlearned

Introduction

- Technology is not value-neutral---this is an advantage
- Don't hide policy decisions behind technological necessity
- Political change through technology, not white papers
- The issue we're addressing here: *Privacy*
  - An important political agenda and a fundamental right
  - The false dichotomy of privacy vs technology
  - Doing a better job leads to better design
- The importance of trust
  - That's what we're really doing here
  - Robustness, security, privacy
  - Users should *never* be surprised

What the architecture enables

- Allow large numbers of people to share information...
  - Collaboration
  - Matchmaking
- ...*without* forcing them to extend a lot of trust
  - Exposures
  - Reliability
- A system design that puts user privacy first
- Doing the right thing for *social* reasons leads to a system that is more technically *robust* as well
  - Hard to subvert
  - Hard to take away once it's given
- *Users don't have to think about the security!*

Privacy and Security
The Threat Model

- Decentralization helps with:
  - Crackers
  - Insiders
  - Subpoenas
- Cryptography helps with:
  - Packet sniffing
  - Traffic analysis
  - Spoofing and replays
  - Subverted agents
  - Subverted distribution
- What don’t we address?
  - Denial of service
  - Mobile code, Byzantine failures, trusted path to binaries, insecure local workstation, poor passphrases, rubber-hose cryptanalysis, ...

The fundamental approach: Decentralization

- Centralized solutions require too much trust even if there is no privacy concern...
  - Hardware failure
  - Overload
  - Business folds
- ...and worrying about privacy makes it much worse:
  - Bad faith
  - Crackers
  - Subpoenas
- Decentralized approaches can solve these problems

The sample application---Yenta:

Bringing together people with similar inter

- Automatically form clusters of users
- Uses for the clusters
  - Matchmaking (1-to-1)
  - Coalition-building & interest-group formation (n-to-n)
  - Finding & building communities
- Some scenarios
  - “Hey, I didn’t know you were working on that, too!”
  - “You mean, there are *others* with the same symptom cluster?”
  - “I’m a technical recruiter...”
  - “Does anybody else in the world share this interest?”
- Internet-based deployment

General class of problems

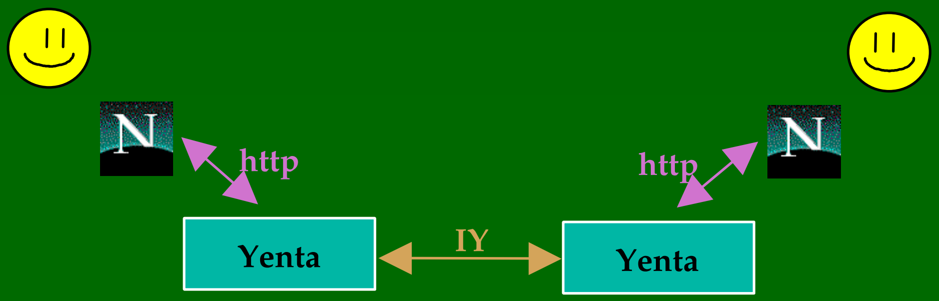
- More than one user in the system
- Users all peers of each other
- Users interact by sharing information
- Not every user knows about every other
- Users can be grouped into clusters based on attributes
- Partial ordering possible among user characteristics
- Some information must be protected from disclosure
- Each user runs the application on a local machine
- Application runs continuously and has persistent state
- High-availability network

Possible applications

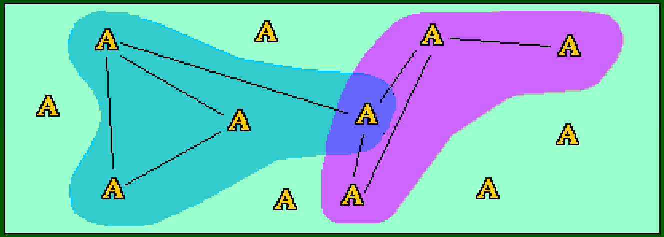
- Yenta---our focus here
- E-commerce
- Collaborative filtering
- Finding experts

Components of the solution

- Agent-based architecture
  - One agent per user
  - Decentralized
- All agents are pseudonymous
  - No connection between agent's name and user's true name
  - Unique agent identity worldwide
- Cryptography
  - All communications and storage are private
  - Agent identity cannot be forged



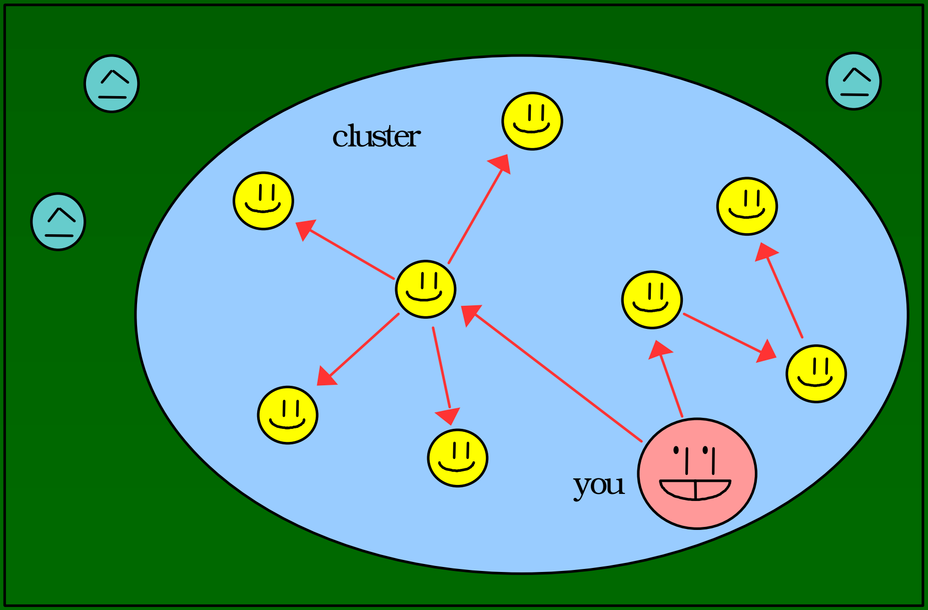
Forming clusters of agents



- Data structures
  - Cluster cache
  - Rumor cache
- Comparisons between peers
  - Peer-to-peer estimations of similarity for each agent
  - Referrals via short-term memories of recent contacts
    - Agents remember address & contents of recent messages
    - Works like word of mouth
- Bootstrapping

Using the clusters

- Sending a message to everyone in a cluster
- Messages between individuals
- Defeating traffic analysis



Reputations via Attestations

- If everyone is pseudonymous, how do I know anything about who I'm talking to?
- Attestations---things you say about yourself...
  - "I work at Yoyodyne."
  - "I won't spam you."
- ...that people who know you can *sign*
- Your agent sends attestations to each one it talks to
- You can use attestations to decide whether to
  - Introduce yourself
  - Accept messages
- Works like keysigning and the PGP web of trust

Design desiderata

- Must be open---publish sources!
  - No design review leads to weak systems
- Use existing crypto
  - Brand-new systems cannot be trusted
- Whole-system design
  - Weak pieces compromise entire system
- Minimize information collected
  - If you don’t want to be subpoenaed for it, don’t collect it
- Nobody’s perfect
  - Security is a goal, not an absolute

Security implementation

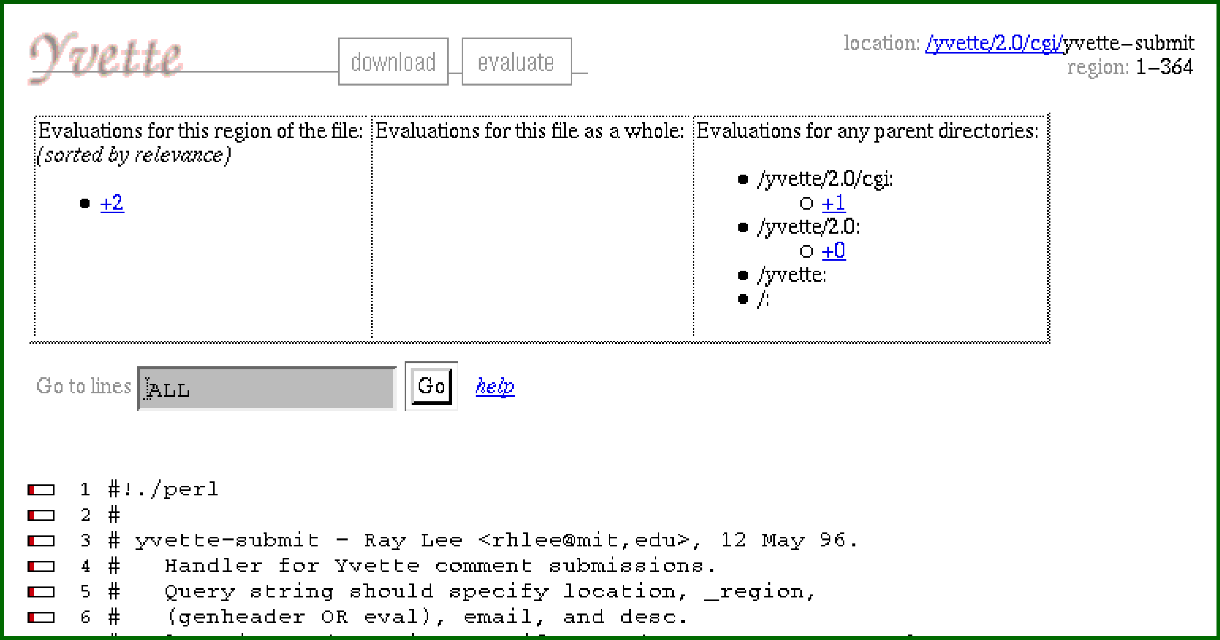
- No central points
- Strong crypto everywhere
  - Between Yentas
  - Between Yenta and browser
  - Persistent state on disk
- Message flooding
- Interest-mixing
- Unforgeable pseudonyms
- Public source

Why strong cryptography?

- Users must be able to trust the system
- Many adversaries with good resources
  - Industrial espionage
  - Government
- Many examples of bad actors
  - A culture of snooping—IRS/GAO report, ...
  - A litany of FBI problems; LAPD wiretaps, ...
  - Many foreign governments much worse ...
  - ...and we' re racing them to the bottom with the PATRIOT act...
- No technical cost---even strong crypto is fast

Trusting the application

- Signatures are part of the story
- Trusting the vendor can only go so far
- Reading source code is a big job!
- Collaboration amongst reviewers—Yvette



location: /yvette/2.0/cgi/yvette–submit
region: 1–364

Evaluations for this region of the file: *(sorted by relevance)*

- +2

Evaluations for this file as a whole:

Evaluations for any parent directories:

- /yvette/2.0/cgi:
  - +1
- /yvette/2.0:
  - +0
- /yvette:
- /:

```
#!./perl
#
# yvette-submit - Ray Lee <rhlee@mit,edu>, 12 May 96.
#   Handler for Yvette comment submissions.
#   Query string should specify location, _region,
#   (genheader OR eval), email, and desc.
```

Decentralization—An evaluation

- All the advantages you've heard already...
  - Privacy and security advantages
  - Overall system can be more robust
- ...but more work for the implementor:
  - Can't just fix it on the server
  - Will always have a mix of versions in the field
  - Users are already conditioned to expect centrality
- Making the implementor's life more difficult...
  - ...doesn't matter to users
  - ...might encourage fewer totally-buggy early releases
  - ...but might kill you if time-to-market is the*only* metric
- Making money requires you to think harder
  - No obvious revenue stream
  - Data mining is *profitable* and this tries to *prevent* it

Comparison and Advice

- Recent social-networking approaches---all centralized
  - LiveJournal
  - Friendster
  - Orkut
- Centralization encourages the wrong mindset
  - Creepy terms of service
  - *Somebody'* sthinking about money
  - Laughable security (passwords, subpoenas, PATRIOT)
  - A single point of failure
- Other P2P systems aren' t really social
  - Avoiding law enforcement
  - Coping with massive bandwidth needs
- Can we make a hybrid?
  - Decentralized architecture...
  - ...but human names & distributed discovery

Discussion?

foner@media.mit.edu

http://foner.www.media.mit.edu/people/foner/