Privacy-Aware Decentralized Architectures for Socially Networked Systems

Shirin Nilizadeh

Indiana University Bloomington
September 7, 2013
Online Social Networks (OSNs) have revolutionized the way our society communicates.

Reference:
Many applications leverage social networks to provide their services in novel ways.
But at the cost of user privacy:
Users are not in control of their private data

You give Google a worldwide license to use, host, store, reproduce, modify, create derivative works, communicate, publish, publicly perform, publicly display and distribute such content.

our partners, the advertisers that purchase ads on the site and the developers that build the games, application and websites you use.
Building a distributed P2P system is an alternative approach

- P2P networks are inherently resistant to censorship and centralized control

- Have been used for building large scale systems:
  1. Direct and real-time communication
  2. Distributed computation
  3. Internet service support
  4. Content distribution: Napster, Kazaa, eDonkey/Overnet, BitTorrent, and Gnutella
However, providing OSN’s functionalities in P2P systems raises new privacy concerns

- In P2P networks not everyone is trustworthy
- OSNs are dealing with personal information and files
- Some peers may collude to aggregate data
Objective:

Decentralized and efficient architectures for socially networked systems that provide strong security and privacy guarantees
Design a privacy-aware peer-to-peer network to support sensitive Q&A

Query topic: \{health, alcohol\} ?

Expertise: \{education, health\}

Bob

Evelyn

Security and Privacy Requirements

1. Sender Anonymity

2. Expertise/Interest unlinkability:
   cannot tell Alice is a pro-choice/life activist

3. Unobservable querying and responding
Our P2P design supports private queries and leverages social neighborhoods.
We evaluate the privacy provided against honest-but-curious attackers

Global Attacker: views all messages and infer the online/offline and idle status of all the nodes at any time

Colluding Attackers: some fraction of nodes are compromised and thus have partial knowledge

Capability: attackers can link (or not link) questions/answers from a particular expert
Anonymity of users degrades over time; however if answers are not linkable, anonymity improves greatly.

Simulation:
5 scale-free graphs with 60,000 nodes
Communities’ size: 85-115 nodes
Human models from Skype and Aardvark usage were used to simulate queries and answers.
Cachet: A Decentralized Architecture for Privacy Preserving Social Networking

Sonia Jahid, Shirin Nilizadeh, Prateek Mittal, Nikita Borisov, Apu Kapadia·DECENT: A Decentralized Architecture for Enforcing Privacy in Online Social Networks, 4th IEEE International Workshop on SEcurity and SOCial Networking (SESOC’12), 2012
Objective:

A generalized architecture for enforcing access control in a decentralized OSN. The focus is on providing data confidentiality, integrity, and availability in the presence of malicious nodes in a distributed setting.
Users’ Information is stored in a Distributed Hash Table (DHT)
Cachet uses an object-oriented data structure
Attribute Based Encryption supports flexible confidentiality policies

- User-defined attribute-based policy
- Frequent revocation
Providing confidentiality and integrity (read, write and append policies)

- Alice’s wall
  - Ref to Alice’s status = (objID; ABE(K; P); WPK)
  - ...
  - Ref to object n

Symmetric Encrypted with K

- Sign with WSK
- Ref to a comment
- Sign with ASK
Downloading and reconstructing a wall or an aggregated newsfeed is a lengthy process

1. Decrypting ABEncrypted update objects to yield:
   <update’s DHT key and symmetric decryption key>

2. Using DHT keys to find recent objects

3. Decrypting them with their corresponding symmetric keys.

Carol’s photo:
Social Caching

leverage *social trust relationships* to reduce expensive decryption operations
Online social contacts can provide *decrypted* objects to contacts who also satisfy the policy.
Need to build the social overlay first:
The presence protocol is also decentralized.

- Presence object: <Current IP address, Port>
Gossip-based social caching algorithm: A greedy approach

1. Creating the Presence Table

2. Selecting a Contact; If all contacts are visited or known to be offline, proceed to step 7

3. DHT Lookup and Connection

4. Pulling Information

5. Caching Information

6. Updating Presence Table; returning to Step 2 to locate the next social contact to connect to.

7. Performing DHT Lookups for online social contacts with No Mutual Social Contacts

<table>
<thead>
<tr>
<th>Friend</th>
<th># mutual friends</th>
<th>On/Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>25</td>
<td>On</td>
</tr>
<tr>
<td>Carol</td>
<td>20</td>
<td>Off</td>
</tr>
<tr>
<td>Mary</td>
<td>13</td>
<td>On</td>
</tr>
<tr>
<td>Diana</td>
<td>7</td>
<td>Off</td>
</tr>
<tr>
<td>Eve</td>
<td>2</td>
<td>On</td>
</tr>
</tbody>
</table>
We built a simulator to study cache performance

• Newsfeed application, prototype in Java

• Simulation environment
  – FreePastry Simulator
  – Social graph: Facebook friendship graph from the New Orleans regional network with 63,732 nodes and 1.54 million edges
  – 10%, 30%, 50% online friends
    • Based on Skype statistics, the 10-30% range is more pertinent

• Performance metrics
  – hitRate = fraction of newsfeed constructed from cache
  – progressiveHitRate(d) = fraction of newsfeed after d lookups
Social caching provides most of the newsfeed... 
...but not all of it (need the DHT)
Most of the social cache’s benefit comes from the first ~15 DHT lookups

The Average Progressive Hit Rate for users who have 100 to 200 social contacts
Speedup of loading the newsfeed: 5X-10X (25X-50X for the first 80-90% updates)

Even with only 10% of social contacts online, social caching provides performance improvement.
Contributions

- A hybrid combination of DHT and social contacts
- Demonstrated that a decentralized approach to privacy-preserving socially networked systems is practical
Thanks!