**Background: Sybil Attack**

- **Sybil attack**: Single user pretends many fake/sybil identities
  - Creating multiple accounts from different IP addresses
- Sybil identities can become a large fraction of all identities
  - Out-vote honest users in collaborative tasks

**Background: Defending Against Sybil Attack**

- Using a trusted central authority
  - Tie identities to actual human beings
- Not always desirable
  - Can be hard to find such authority
  - Sensitive info may scare away users
  - Potential bottleneck and target of attack
- Without a trusted central authority
  - Impossible unless using special assumptions [Douceur’02]
  - Resource challenges not sufficient -- adversary can have much more resources than typical user
SybilGuard Basic Insight: Leveraging Social Networks

Our Social Network Definition

- Undirected graph
- Nodes = identities
- Edges = strong trust
  - E.g., colleagues, relatives

SybilGuard Basic Insight

- $n$ honest users: One identity/node each
- Malicious users: Multiple identities each (sybil nodes)

Observation: Adversary cannot create extra edges between honest nodes and sybil nodes

SybilGuard Basic Insight

Goal of Sybil Defense

- Goal: Enable a verifier node to decide whether to accept another suspect node
  - Accept: Provide service to / receive service from
  - Idealized guarantee: An honest node accepts and only accepts other honest nodes

- SybilGuard:
  - Bounds the number of sybil nodes accepted
  - Guarantees are with high probability
  - Approach: Acceptance based on random route intersection between verifier and suspect
Random Walk Review

- Pick random edge d
- Pick random edge e
- Pick random edge c

Random Route: Convergence

- Using routing table gives Convergence Property
- Routes merge if crossing the same edge

Random Route: Back-traceable

- Using 1-1 mapping gives Back-traceable Property
- Routes may be back-traced

Random Route Intersection: Honest Nodes

- Verifier accepts a suspect if the two routes intersect
- Route length $w$: $\sim \sqrt{n \log n}$
- W.h.p., verifier’s route stays within honest region
- W.h.p., routes from two honest nodes intersect
Random Route Intersection: Sybil Nodes

- SybilGuard bounds the number of accepted sybil nodes within $g \times w$
  - $g$: Number of attack edges
  - $w$: Length of random routes

- Next ...
  - Convergence property to bound the number of intersections within $g$
  - Back-traceable property to bound the number of accepted sybil nodes per intersection within $w$

Bound # Intersections Within $g$

- Convergence: Each attack edge gives one intersection
  - $\Rightarrow$ at most $g$ intersections with $g$ attack edges

Intersection = (node, incoming edge)

Bound # Sybil Nodes Accepted per Intersection within $w$

- Back-traceable: Each intersection should correspond to routes from at most $w$ honest nodes
- Verifier accepts at most $w$ nodes per intersection
- Will not hurt honest nodes

Summary of SybilGuard Guarantees

- Power of the adversary:
  - Unlimited number of colluding sybil nodes
  - Sybil nodes may not follow SybilGuard protocol
  - W.h.p., honest node accepts $\leq g \times w$ sybil nodes
  - $g$: # of attack edges
  - $w$: Length of random route

<table>
<thead>
<tr>
<th>If SybilGuard bounds # accepted sybil nodes within</th>
<th>Then apps can do</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n/2$</td>
<td>byzantine consensus</td>
</tr>
<tr>
<td>$n$</td>
<td>majority voting</td>
</tr>
<tr>
<td>not much larger than $n$</td>
<td>effective replication</td>
</tr>
</tbody>
</table>
Overview

- Social Networks
- Multiplayer Games
- Class Feedback Discussion

Individual Player’s View

- Interactive environment (e.g. door, rooms)
- Live ammo
- Monsters
- Players
- Game state

High-Speed, Large-Scale, P2P: Pick 2

- Many console games are peer hosted to save costs
- 1000+ player games need dedicated servers

Question: Can we achieve all 3?

P2P

Local View
Replica objects

Primary object
High-Speed Internet

Local View

Inter-object writes must be reflected very quickly

Primary object

Replica objects

High-Speed

Local View

20 updates/sec
≈ 16 kbps per player
Delay must be < 150ms
[Beigbeder '04]

Primary object

Replica objects

Large-Scale Internet

Area-of-Interest (AOI) Filtering

- Large shared world
  - Composed of map information, textures, etc
  - Populated by active entities: user avatars, computer AI’s, etc
- Only parts of world relevant to particular user/player

Large shared world

Composed of map information, textures, etc

Populated by active entities: user avatars, computer AI’s, etc

Only parts of world relevant to particular user/player

Game World

Player 1

Player 2
Object Model

- Game world treated as collection of objects
- Single primary copy for each object
  - Maintained at a single owner node
  - Serialization point for updates
  - Determines actions/behavior of object
- Secondary replicas
  - Primary object may need to examine other objects to determine action
  - Need low latency access to object → must replicate object in advance
- How to find set of objects to replicate? → hardest part

Object Discovery – Solution

- Use publish-subscribe to “register” long-lived distributed lookups
- Publications created each time object is updated (or periodically when no update is done)
  - Publication contents = state of object

Publish-Subscribe (PubSub) Overview

- Key feature → subscription language
  - Subject/channel-based subscriptions (e.g. all publications on the IBM stock channel)
  - Rich database-like subscription languages (e.g. all publications with name=IBM and volume > 1000)
- State-of-the-art
  - Scalable distributed designs with channel-based subscriptions
  - Unscalable distributed designs with rich subscriptions
  - Goal → scalable distributed design with “richer” subscriptions
  - Example: $x \leq 200 \land y < 100$ → Support for multi-dimensional range predicates

Using DHTs for Range Queries

- No cryptographic hashing for key →
Using DHTs for Range Queries

- Nodes in popular regions can be overloaded
- Load imbalance!

DHTs with Load Balancing

- Mercury load balancing strategy
  - Re-adjust responsibilities
- Range ownerships are skewed!

DHTs with Load Balancing

- Each routing hop may not reduce node-space by half!
  - no log(n) hop guarantee

Ideal Link Structure
**MERCURY Routing [Sigcomm 2004]**

- Send subscription to *any one* attribute hub
- Send publications to *all* attribute hubs
- Tunable number of long links can range from full-mesh to DHT-like

**Object Discovery – Basic Design**

- Use publish-subscribe to “register” long-lived distributed lookups
- Publications created each time object is updated (or periodically when no update is done)
  - Publication contents = state of object

**Prefetching and Persistence**

- Basic design has problems
  - Collecting/updating existing state is too slow
  - High subscription update rate
- 1st fix: use Mercury only for object discovery and not state update
- 2nd fix: persistent publications/subscriptions
  - Publication lifetimes: subsequent subscriptions would immediately trigger transmission
  - Subscription lifetimes: enabled soft-state approach to subscriptions
- 3rd fix: predict future subscriptions
  - Creates a new hybrid of persistent storage and pub/sub

**Colyseus Architecture Overview**

1. Specify Predicted Interests: 
   - \((3 < x < 60 \& 10 < y < 200)\) TTL 30sec
2. Locate Remote Objects: 
   - \((P1 \text{ on } s1, P4 \text{ on } s2)\)
3. Register Replicas: 
   - \((R3 \text{ on } s2, R4 \text{ on } s2)\)
4. Synchronize Replicas: 
   - \((R3, R4)\)
5. Infer Interests: 
   - \((R3, R4, P2)\)
6. Optimize Placement: 
   - Migrate \(P1\) to server \(s2\)
View Inconsistency

Observations:
1. View inconsistency is small and gets repaired quickly
2. Missing objects on the periphery

Area-of-Interest (AOI) Filtering

- Only receive updates from players in your AOI
  - Colyseus [Bharambe ‘06]
  - VOK [Hu ‘06]
  - SimMUD [Knutsson ‘04]
- Problems:
  - Open-area maps, large battles
  - Region populations naturally follow a power-law
    [Bharambe ‘06, Pittman ‘07]
- Requirement: ~1000 players in same AOI

Smoothing Infrequent Updates

- Send guidance (predictions) instead of state updates
- Guidable AI extrapolates transitions between points
  - E.g., game path-finding code

- Problem: Predictions are not always accurate
  - Interactions appear inconsistent
  - Jarring if player is paying attention

Donnybrook: Interest Sets

- Intuition: A human can only focus on a constant number of objects at once
  [Cowan ‘01, Robson ‘81]
- Only need a constant number of high-accuracy replicas
- Interest Set: The 5 players that I am most interested in
  - Only need 1 update/sec from everyone else
Donnybrook: Interest Sets

- How to estimate human attention?
  - Attention(i) = how much I am focused on player i

\[
\text{Attention}(i) = f_{\text{proximity}}(d_i) + f_{\text{aim}}(\theta_i) + f_{\text{interaction-recency}}(t_i)
\]
Overview

- Social Networks
- Multiplayer Games
- Class Feedback Discussion

Discussion

- Lecture topics?
  - Balance of networking/dist sys/ubiq?
  - Research/adv topics/practical engineering?
- Text book vs. papers vs. lectures
- Projects
  - Free form vs. proposed
- Android vs. no Android?