CIS 700/003: Distributed Systems meet Social Networks

Sybil attack

March 23, 2010
Where are we?

- Measurement and analysis of MAD systems
- Building MAD systems
- Faults and Misbehavior
- Internet crime
  - Spoofing
  - Sybil attack
  - Denial of Service
  - Phishing
- Privacy and confidentiality
- Novel opportunities
- Experience
SybilGuard: Defending Against Sybil Attacks via Social Networks

Haifeng Yu  Michael Kaminsky  Phillip B. Gibbons  Abraham Flaxman

SIGCOMM 2006
Discussion points

- Is this approach practical?
- How good are their guarantees?
- Is it really difficult to establish new social links?
- Overhead?
- Evaluation convincing?
- How reliable is their estimation of N?
Rest of this class

- Other defenses against Sybil attacks
- SybilLimit
Do Sybil attacks occur in practice?

■ Why is it called a 'Sybil' attack?
  ■ Based on a 1973 book by F. R. Schreiber about a patient called 'Sybil Dorsett' (pseudonym)
  ■ 'Sybil' was suffering from dissociative identity disorder; she manifested 16 different personalities

■ Do Sybil attacks occur in practice?
  ■ Yes. For example, they have been observed in the Maze P2P system (Lian et al., ICDCS 2007)
  ■ Demonstrated to be surprisingly easy in practice, e.g., in the widely-used eMule system (Steiner et al., CCR 2007)
Other defenses against Sybil attacks

- Typical approach is to rely on the fact that the attacker is limited in some resource R
- R can be many different things. Examples:
  - **Storage**: Give each node a large amount of uncompressible data and randomly verify small excerpts.
  - **Bandwidth**: Send lots of traffic and require the node to respond within a certain time interval.
  - **Computation**: Ask the node to solve a difficult computational puzzle whose solution is easy to check (e.g., invert hash)
  - **Physical locations**: Treat all nodes within some distance as a single node
    - If attacker has nodes in a few locations, he can fabricate new locations!
  - **IP addresses**: Require node to respond to probe packets
Other defenses against Sybil attacks

- **More examples:**
  - **Real-world identities:** Certify mapping to passport
  - **Money:** Charge some amount of money for each new identity
  - **Human attention:** To get a new identity, require solving a task that computers cannot solve (e.g., CAPTCHA)
    - Can be defeated by hiring low-wage workers to solve CAPTCHAs
  - **Physical machines:** Bind identities to a TPM, etc.

- **Challenge:**
  - Assume the attacker has $r_A$ units of the resource, and the weakest legitimate user has $r_M$
  - If we want to allow the legitimate user into the system, the attacker can create $r_A / r_M$ identities
SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks

Haifeng Yu  Michael Kaminsky  Phillip B. Gibbons  Feng Xiao

Oakland 2008
Motivation

- SybilGuard's has several major limitations

- **Problem #1:** Each attack edge allows a large number of Sybil nodes to be accepted
  - Example: In a million-node network, nearly 2000 nodes/edge

- **Problem #2:** No guarantees if the number of attack edges is above a certain threshold
  - Example: In a million-node network, SybilGuard can't bound the number of sybils at all if there are >15,000 attack edges

- **Problem #3:** Assumes fast-mixing property
  - Has never been validated in the real world!
  - Is not obvious at all, given the presence of communities
Contributions of SybilLimit

<table>
<thead>
<tr>
<th>Number of attack edges $g$ (unknown to protocol)</th>
<th>SybilGuard accepts</th>
<th>SybilLimit accepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o(\sqrt{n}/ \log n)$</td>
<td>$O(\sqrt{n} \log n)$</td>
<td>$O(\log n)$</td>
</tr>
<tr>
<td>$\Omega(\sqrt{n}/ \log n)$ to $o(n/ \log n)$</td>
<td>unlimited</td>
<td>$O(\log n)$</td>
</tr>
<tr>
<td>below $\sim 15,000$</td>
<td>$\sim 2000$</td>
<td>$\sim 10$</td>
</tr>
<tr>
<td>above $\sim 15,000$ and below $\sim 100,000$</td>
<td>unlimited</td>
<td>$\sim 10$</td>
</tr>
</tbody>
</table>

- Guarantees are improved dramatically
  - Within $\log n$ of the optimum for this type of system
- Formal proofs for each property
How do they do this?

- **Idea #1:** Use many random routes, but shorter ones
- **Idea #2:** Intersect edges, not nodes
- **Idea #3:** Limit how often each edge is used
- **Idea #4:** Use a more robust technique to estimate the system parameter
Registering tails

- **Idea #1:** Intersect edges, not nodes
  - Use shorter routes (length $O(\log n)$) but multiple of them (square root of the number of edges between honest nodes)
  - For each route, remember only the last edge (the 'tail')
  - Verifier rejects suspect if their sets of routes do not share any tails
  - Effect #1: Stationary distribution on edges is always uniform
  - Effect #2: Reduces the number of slots available to attacker
Idea #2: Limit how often each tail may be used for admitting new suspects

- Because there are so many routes now (SybilGuard had just one), some of them are likely to escape to the Sybil region.
- But we can expect intersections to be distributed randomly, so we can enforce a limit on how often each tail can be used.
- Result: Limits #sybils accepted through 'escaping' tails.
Estimating the number of routes

Idea #3: Use a more robust technique to estimate the system parameter

- Recall that SybilGuard needs to estimate the length of the walk, but this fails if there are too many attack edges
- SybilLimit needs to estimate #routes
- Uses a novel technique that mixes real suspects with some random other suspects that are known to be mostly honest
- Guarantees that parameter will never be overestimated, regardless of the attacker's behavior
Lower bound

- SybilLimit bounds \#sybils accepted per attack edge to $O(\log n)$

- Could this be improved further?
  - Yes, but only by at most $\log n$
  - Any system that relies on the same property as SybilLimit (change in mixing time) would accept at least $\Omega(1)$ sybils per attack edge
Mixing time

- Provable guarantees of SybilLimit rely on the assumption that social network has small (O(\log n)) mixing time.

- Question: Is this true of real social networks?
  - To answer, crawled Friendster, LiveJournal, and DBLP.
  - Problem: Cannot show directly that mixing time is O(\log n).
    - O(\log n) is asymptotic behavior.
  - But it is sufficient to show that small values of w are sufficient for SybilLimit to work well.
Mixing time

<table>
<thead>
<tr>
<th>Data set source</th>
<th>Friendster</th>
<th>LiveJournal</th>
<th>DBLP</th>
<th>Kleinberg</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>932,512</td>
<td>900,822</td>
<td>106,002</td>
<td>1,000,000</td>
</tr>
<tr>
<td># undirected edges</td>
<td>7,835,974</td>
<td>8,737,636</td>
<td>625,932</td>
<td>10,935,294</td>
</tr>
<tr>
<td>( w ) used in SybilLimit</td>
<td>10</td>
<td>10</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>( r ) used in SybilLimit</td>
<td>8,000</td>
<td>12,000</td>
<td>3,000</td>
<td>10,000</td>
</tr>
</tbody>
</table>

- **Result:** Small \( w \) values are sufficient for large-scale social networks
  - Evidence (though not proof) that the small-mixing-time assumption holds for real social networks
How many sybils are accepted now?

**Figure 8. Friendster**

**Figure 9. LiveJournal**

**Figure 10. DBLP**

**Figure 11. Kleinberg**
Take-away points

- Sybil attacks are easy, dangerous, and real
  - Observed in deployed P2P systems (e.g., Maze)
  - Not difficult to do
  - Can subvert BFT, voting systems, DHT routing, ...

- Many defenses are available
  - Typically assume the attacker is limited in some resource
  - Examples: Computation power, human attention
  - Danger of excluding resource-limited but legitimate users

- Recent defenses based on social networks
  - Attacker assumed to be limited in #trust relationships;
    results in small min-cut between sybils and honest users
  - Implementation far from trivial (see SybilGuard/SybilLimit)
Where are we?

- Measurement and analysis of MAD systems ✔
- Building MAD systems ✔
- Faults and Misbehavior ✔

- Internet crime
  - Spoofing ✔
  - Sybil attack ✔
  - Denial of Service Next time
  - Phishing

- Privacy and confidentiality
- Novel opportunities
- Experience
Next week you will learn:

How to defend against Denial of Service attacks using a Denial of Service attack