Recall: Lec-5

- Reputation systems:
  - Why we need reputation/trust systems
  - What is a global reputation model
  - What is a personalized reputation model
  - Case studies: PageRank and Credence
Lecture Roadmap

• Sybil Attacks
• Sybil Defenses
• Case Study: DSybil
• Case Study: SybilGuard
Tom and Jerry Games
Tom and Jerry Games

P2P content-sharing systems
Tom and Jerry Games

Pollution attacks

P2P content-sharing systems
Tom and Jerry Games

Pollution attacks

P2P content-sharing systems

Reputation systems
Tom and Jerry Games

Pollution attacks

Sybil attacks

P2P content-sharing systems

Reputation systems
Tom and Jerry Games

- Pollution attacks
- Sybil attacks
- P2P content-sharing systems
- Reputation systems
- Sybil defenses
What is a sybil attack?

• Sybil attack:
  - Single adversary pretends many fake/sybil identities
  - Create a huge number of accounts or identities
  - Sybil identities launch attacks in arbitrary ways
  - Outvoting honest users in any collaborative system
Sybil
Let’s attack Credence

• Credence assumes:
  - there is no large-scale sybil attack
  - most of malicious users do not collude
  - adversary cannot control the majority of nodes to launch attacks in a smart way
  - Credence tries to handle sybil attackers by big files, but this does not work well

• Credence is vulnerable to sybil attacks
Let's attack Credence

File 1
- C: -1
- D: +1

File 2
- A: +1
- B: +1
- C: -1

File 3
- A: -1
- B: -1

File 4
- A: -1
- C: +1
- D: -1

File 5
- D: -1

File 6
- A: +1
- D: -1

File 7
- C: +1
- G: -1
Let's attack Credence

Assume File 5 is polluted and B is a sybil identity
Let's attack Credence
Let's attack Credence

File 1
C: -1
D: +1

File 2
A: +1
B: +1
C: -1

File 3
A: -1
B: -1

File 4
A: -1
C: +1
D: -1

File 5
D: -1

File 6
A: +1
D: -1

File 7
C: +1
G: -1
Let's attack Credence

File 1
- A: -1
- B: +1
- C: -1
- D: +1

File 2
- A: +1
- B: -1
- C: +1

File 3
- A: -1
- B: +1
- C: +1
- D: -1

File 4
- A: -1
- B: +1

File 5
- A: +1
- D: -1

File 6
- A: +1
- C: +1

File 7
- C: +1
- G: -1
• # of sybil identities >> # of normal users
• Sybil attackers can collude and launch arbitrary attacks
Scenarios of Sybil Attacks

- Sybil attacks exist in any public systems/services:
  - There is no stringent regulatory authority
  - Attackers can have much more resources than normal users
  - Attackers can control different identities to launch attacks in arbitrary ways, e.g., collusion
Scenarios of Sybil Attacks

- Scenarios are vulnerable to sybil attacks:
  - Recommendation/Reputation systems, e.g., Digg
  - Gaming systems
  - Social network systems, e.g., Twitter and Facebook
  - Any collaborative systems
Scenarios of Sybil Attacks

- Scenarios are vulnerable to sybil attacks:
  - Recommendation/Reputation systems, e.g., Digg
  - Gaming systems
  - Social network systems, e.g., Twitter and Facebook
  - Any collaborative systems

- What scenarios do not have sybil identities:
  - This is hard to say ...
  - It is hard to create massive sybil identities in bank systems
  - In some scenarios, it is not meaningful to launch sybil attacks
Lecture Roadmap

• Sybil Attacks
• **Sybil Defenses**
• Case Study: DSybil
• Case Study: SybilGuard
Defending against sybil is challenging

• Impossible to defend in decentralized systems unless special assumptions

• This has been formally proved by Douceur [IPTPS’02]
Defending against sybil is challenging

- Defending against sybil attacks in centralized systems:
  - Using a trusted central authority
  - Tying identities to actual human beings, e.g., social security number

- Not always desirable:
  - Tying identities to authority is difficult in most systems
  - Sensitive information may scare away normal users
  - Authority might be bottleneck and target of attacks
Technical Solutions

• Some defenses are proposed against sybil attacks:
  - Sybil detection: identifying if a node is a sybil identity
  - Sybil tolerance: making my system functions correctly without identifying sybil identities
Technical Solutions

• Sybil tolerance:
  - servicing correctly without detecting sybil identities
  - leveraging social networks or historical activities to upgrade the probability of communicating with good users
  - case study: DSybil [S&P’09]

• Sybil detection:
  - identifying if a node is a sybil identity
  - leveraging social networks or some observed behavior patterns, e.g., timing, to detect sybil identities
  - case study: SybilGuard [SIGCOMM’06]
Lecture Roadmap

- Sybil Attacks
- Sybil Defenses
- **Case Study: DSybil**
- Case Study: SybilGuard
DSybil [S&P’09]

- DSybil offers the following properties:
  - Designed based on feedback and trust
  - Significantly migrate the influence of sybil identities
  - Loss (# of consumed bad object) is provable within $O(D \log M)$
    even under worst-case attack, where
    $D$ is the dimension of the objects (less than 10) and
    $M$ is the max # of sybil identities voting on each object
  - The guarantee on the loss is optimal.
DSybil [S&P’09]

- Designed based on feedback and trust
- Significantly migrate the influence of sybil identities
- Loss (number of consumed bad objects) is provable within $O(D \log M)$ even under worst-case attack, where $D$ is the dimension of the objects (less than 10) and $M$ is the max # of sybil identities voting on each object
- The guarantee on the loss is optimal.

Assumptions?
Assumption: Voting Behaviors

- Leveraging typical voting behaviors of honest users
  - Heavy-tail distribution of honest users’ voting behaviors
  - The existence of very active users who cast many votes
Assumption: Voting Behaviors

- Leveraging typical voting behaviors of honest users
  - Heavy-tail distribution of honest users’ voting behaviors
  - The existence of very active users who cast many votes

This assumption makes sense, because this phenomenon has been revealed in real-world voting systems, e.g., Digg
The Design Insight in DSybil

- If user is already getting “enough help”, then do not give them more reputations

- This insight enables us to avoid giving reputation to sybil identities that want to obtain reputations freely
System Model

- Objects to be consumed are either good or bad
- Votes are only positive
- DSybil is deployed in personalized reputation systems
System Model

DSybils does not know which one is good

Each DSybils round has a pool of objects:
- DSybils recommends one object for Alice to consume
- Alice provides feedback after consuming the object
- DSybils adjusts reputation based on the feedback
System Model

Each identity is able to cast at most one vote/object.

At most $M$ (e.g., $10^{10}$) sybil identities voting on each product.
Initial Round: Classifying Objects

Each identity starts with initial reputation 0.2

We define an object is overwhelming if its reputation $\geq 1$
The First Round

E: 0.2
F: 0.2
Total: 0.4

H: 0.2
Total: 0.2

G: 0.2
Total: 0.2

H: 0.2
Total: 0.2
The First Round

There is no overwhelming object
The First Round

There is no overwhelming object

- 1. Recommend a random object to Alice
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, multiply reputation of voters by 2
The First Round

There is no overwhelming object

- 1. Recommend a random object to Alice
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, multiply reputation of voters by 2
The First Round

There is no overwhelming object

1. Recommend a random object to Alice
2. Adjust reputation after the feedback
   - if obj is bad, multiply reputation of voters by 0.5
   - if obj is good, multiply reputation of voters by 2
The Second Round

B: 0.2
F: 0.2
Total: 0.4

E: 0.4
H: 0.4
Total: 0.8

H: 0.4
C: 0.2
Total: 0.6
The Second Round

There is no overwhelming object

- 1. Recommend a random object to Alice
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, multiply reputation of voters by 2
The Second Round

There is no overwhelming object

- 1. Recommend a random object to Alice
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, multiply reputation of voters by 2
The Second Round

- B: 0.2
- F: 0.2
- Total: 0.4

- E: 0.8
- H: 0.8
- Total: 1.6

- H: 0.8
- C: 0.2
- Total: 1

There is no overwhelming object

- 1. Recommend a random object to Alice
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, multiply reputation of voters by 2
The Third Round

There are overwhelming objects
The Third Round

There are overwhelming objects

1. Recommend Alice a random overwhelming object
2. Adjust reputation after the feedback
   - if obj is bad, multiply reputation of voters by 0.5
   - if obj is good, no additional reputation given out (insight #2)
The Third Round

There are overwhelming objects

- 1. Recommend Alice a random overwhelming object
- 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, no additional reputation given out (insight #2)
The Third Round

There are overwhelming objects

• 1. Recommend Alice a random overwhelming object
• 2. Adjust reputation after the feedback
  - if obj is bad, multiply reputation of voters by 0.5
  - if obj is good, no additional reputation given out (insight #2)
There are overwhelming objects

1. Recommend Alice a random overwhelming object
2. Adjust reputation after the feedback
   - if obj is bad, multiply reputation of voters by 0.5
   - if obj is good, no additional reputation given out (insight #2)
The Third Round

There are overwhelming objects

1. Recommend Alice a random overwhelming object
2. Adjust reputation after the feedback
   - if obj is bad, multiply reputation of voters by 0.5
   - if obj is good, no additional reputation given out (insight #2)
The Fourth Round

There is one overwhelming object

1. Recommend Alice a random overwhelming object
2. Adjust reputation after the feedback
   - if obj is bad, multiply reputation of voters by 0.5
   - if obj is good, no additional reputation given out (insight #2)
Why DSybil works?
Recall: The First Insight in DSybil

- Leveraging typical voting behaviors of honest users
  - Heavy-tail distribution of honest users’ voting behaviors
  - The existence of very active users who cast many votes
Why DSybil works?

There is a set consisting of several users who vote the majority of good objects.
Why DSybil works?

There is a set consisting of several users who vote the majority of good objects.
Lecture Roadmap

• Sybil Attacks
• Sybil Defenses
• Case Study: DSybil
• Case Study: SybilGuard
SybilGuard [SIGCOMM’06]

• SybilGuard uses social network to detect sybil identities:
  - Reject providing services to sybil identities
  - Rely on social network
  - General to any decentralized system scenario

There is no solution capable of enabling you to avoid sybil attacks completely
Social Networks
Leveraging Social Networks

- Undirected graph
- Nodes = identities
- Edges = strong trust
  - e.g., colleagues and friend relationships
Leveraging Social Networks

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

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

Key Insight:
It is difficult for attackers to create extra edges between honest nodes and sybil nodes
Key Insight

• Small cuts
• Called attack edge
• Connecting two large groups

• But cannot search for such cut which is NP-hard problem
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 nodes
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 nodes
Goal of Sybil Defense

- **Goal**: Enable a verifier node to decide whether to accept another suspect node

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

  *Note that random route is different from random walk!!!*
Random Walk

Diagram showing nodes labeled a, b, c, d, e, and f connected by lines and arrows.
Random Walk
Random Walk
Random Walk
Random Walk
Random Walk
Random Walk
Random Walk
SybilGuard uses Random Route

- Randomized routing table:
  - Random 1 to 1 mapping between incoming and outgoing edges
  - Routes merge if crossing the same edge
Random Route

Diagram with nodes a, b, c, d, e, and f, and edges a → d, b → a, c → b, d → c, d → e, and e → d.
Random Route

a → d
b → a
c → b
d → c
d → e
e → d
Random Route

a → d
b → a
c → b
d → c
d → e
e → d
Random Route

\[
\begin{align*}
  \text{a} & \rightarrow \text{d} \\
  \text{b} & \rightarrow \text{a} \\
  \text{c} & \rightarrow \text{b} \\
  \text{d} & \rightarrow \text{c}
\end{align*}
\]

\[
\begin{align*}
  \text{d} & \rightarrow \text{e} \\
  \text{e} & \rightarrow \text{d}
\end{align*}
\]
Random Route

a
b
c
d
e
f

a → d
b → a
c → b
d → c
d → e
e → d
Random Route

- a → d
- b → a
- c → b
- d → c

- d → e
- e → d

- a → d (dashed line)
- f → e
Random Route

a           d
b           a
c           b
d           c
d           e
e           d
Random Route

- Back-traceable:
  - If we know the route traverses edge, we know the whole route
Back-traceable
Back-traceable

a → d
b → a
c → b
d → c
d → e
e → d
Back-traceable

\[ \text{Diagram:} \]

- a \rightarrow d
- b \rightarrow a
- c \rightarrow b
- d \rightarrow c
- d \rightarrow e
- e \rightarrow d

\[ \text{Diagram diagram:} \]

- a
- b
- c
- d
- e
- f

\[ \text{Diagram diagram:} \]
SybilGuard Working Principle

- Both V and S randomly choose d neighbors to launch d random routes
- Verifier accepts a suspect if d/2 intersections between routes
  - Route length w
Random Route Intersection: Normal

- Both V and S randomly choose $d$ neighbors to launch $d$ random routes
- Verifier accepts a suspect if $d/2$ intersections between routes
  - Route length $w$
  - Verifier’s route stays in honest region
  - Routes from two honest nodes intersect
Random Route Intersection: Sybil

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

• Provable guarantees:
  - Convergence property to bound the number of intersections within $g$
  - Back-traceable property to bound the number of accepted sybil nodes per intersection within $w$
Random Route Intersection: Sybil

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

- Provable guarantees:
  - 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

Must cross attack edge to intersect even if sybil nodes do not follow the protocol
Must cross attack edge to intersect even if sybil nodes do not follow the protocol.

Convergence: Each attack edge gives one intersection at most $g$.
Random Route Intersection: Sybil

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

- Provable guarantees:
  - 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 # Sybil Nodes Accepted within $w$

- Back-traceable:
  - Each intersection should correspond to routes from at most $w$ honest nodes
  - Verifier accepts at most $w$ nodes per intersection
SybilGuard Summary

- SybilGuard uses social network to detect sybil identities:
  - Assumption: Social network, small cuts, and fast-mixing
  - Honest node accepts \( \leq g^*w \) sybil nodes
  - General to any decentralized scenario
SybilGuard Summary

• SybilGuard uses social network to detect sybil identities:
  - Assumption: Social network, small cuts, and fast-mixing
  - Honest node accepts $\leq g \times w$ sybil nodes
  - General to any decentralized scenario

Can I make $g=1$ and $w=1$?
Next Lecture

• In the lec-7, I will cover:
  - Global services become popular
  - We want to control more
  - How do Firewall and NAT work?
  - How to control enterprise-scale network?