Distributed Mechanism Design and Computer Security

John Mitchell    Vanessa Teague
Stanford University

Acknowledgements: J. Feigenbaum, R. Sami, A. Scedrov ...
General problem

◆ Want to design distributed systems that behave well in presence of
  • Honest agents who host parts of system
  • Rational profit-maximizing agents who control some parts of the infrastructure
  • Irrational malicious agents
    - (maybe not too many of these)
Two models

◆ Tamper-proof nodes (FPS)

- Clients are strategic
- Distributed mechanism causes rational agents to bid actual utility [FPS]

◆ Strategic nodes

- Client identified with node
- Agents can
  - Enter own value
  - Run algorithm or lie
  - Pay correctly or not

Obedient processor
Specific research question

◆ Given
  • Distributed mechanism that elicits certain behavior when properly executed with tamper-proof nodes

◆ Design
  • Distributed mechanism that
    - Elicits same behavior
    ★ Includes incentives to execute correctly
    - Is robust against some forms of attack
This talk: Multicast cost sharing

- Distribute some good
- Each node has some utility for the good
- Each link has some cost
- Who gets the transmission?
- How much do they pay?
Outline

- Previous work on multicast cost sharing (Feigenbaum, Papadimitriou & Shenker)
  - Truthful mechanism
  - Distributed algorithm computes mechanism
- New work: strategic nodes model
  - Why we can't just use the FPS algorithm
  - Techniques to encourage compliance
    - Nodes save signed confirmation of msgs
    - Randomized auditing incents compliance
      - Alternative: neighbors rewarded for turning in cheaters
    - Punish, route around nodes that cause trouble
  - Security of new scheme
Two models

◆ Tamper-proof nodes (FPS)
  - Clients are strategic
  - Distributed mechanism causes rational agents to bid actual utility [FPS]

◆ Strategic nodes
  - Client identified with node
  - Agents can
    - Enter own value
    - Run algorithm or lie
    - Pay correctly or not
FPS: Maximum welfare

Welfare 2-4 = -2

Welfare 7-1 = 6

Welfare 1-3 +6 = 4

Welfare 3-2 +0+4 = 5

Utility 2

Utility 1

Utility 3

Utility 7

Utility 8

Cost 1

Cost 2

Cost 3

Cost 4
FPS Pricing Mechanism

◆ If agent does not receive the good
  • Agent pays nothing
◆ If agent receives the good
  • Agent pays:
    the minimum bid needed to get the transmission, given the other players’ bids
◆ Agent maximizes welfare by telling the truth

This is a VCG mechanism
Distributed implementation

1) Send welfare up tree
2) Send min welfare $W_{\text{min}}$ down tree
3) Compute payment $= \text{utility} - W_{\text{min}}$
Two models

- Tamper-proof nodes (FPS)
  - Clients are strategic
  - Distributed mechanism causes rational agents to bid actual utility [FPS]

- Strategic nodes
  - Client identified with node
  - Agents can
    - Enter own value
    - Run algorithm or lie
    - Pay correctly or not
Node can cheat its children

**The truth**

- **Source**
  - Welfare 2-3+2 = 1
  - Cost 3
  - Utility 2

- **Parent**
  - Welfare 7-5 = 2
  - Cost 5
  - Utility 7

- **Child**
  - Welfare 2
  - Cost 5
  - Utility 2

- **Wmin** = 1

**The cheat**

- **Source**
  - Welfare 2-3+2 = 1
  - Cost 3
  - Utility 2

- **Parent**
  - Welfare 2
  - Cost 5
  - Utility 7

- **Child**
  - Welfare 0
  - Cost 3
  - Utility 7

- **Wmin** = 0

Parent pays 1
Child pays 6

Child can’t see that parent doesn’t pay
More ways to cheat

- **Second example**
  - Node can cheat but all messages look consistent

- **Conclusion**
  - Need to use payment and messages to detect cheating
Authenticated protocol

- Assume public-key infrastructure
- Nodes collect signed messages so they can “prove” they paid correctly
  - Sign data from FPS algorithm
  - Add a const. number of other signed messages
- Content provider checks proofs
- Make punishment high enough that benefit of cheating is negative
Preventing mischief

Node J passes on parent’s data

\[ D = \text{Sign}(p, W_{\text{min}}), \text{Sign}(p, W_j) \]

Children verify J’s calculation of \( W_{\text{min}} \)
Content provider may verify payment
Incentives

◆ Nodes are audited (select randomly)
  • If node has proof of having paid correctly, OK
  • If node cannot show proof, punish
    - Fine node, or route around (exclude from transmissions)
    - Stop the transmission
  • When \( j \) is checked, so is one of its children, \( d_i \), to show that \( W^{\text{min}} \) is correct and \( W^{d_i} \) matches.
    - If correct, \( j \) gets reward 1.
    - If not, \( d_i \) gets large fine.

◆ Inconsistent messages
  • A node reporting inconsistent messages signed by another is rewarded. The other is punished.
Fines, welfares, best strategy

◆ If no-one cheats, welfares are the same as FPS
  • Except if j’s child is checked, j gets 1 extra
◆ If someone cheats
  • Cheater’s expected welfare is less than zero
◆ Theorem
  • If ancestors and children are welfare-maximizing, then node maximizes own welfare only by sending consistent values
Security against irrational agents

- **Introduce some malicious nodes**
  - How much can they reduce group welfare?
    - Exclude the compromised node’s utility
  - How much does it cost to be malicious?
    - If you have to be vulnerable, at least make the adversary pay a lot
Security of anti-mischief protocol

- Assume malicious node has honest neighbours
- Security “almost as good” as in tamper-proof nodes model
  - To avoid detection, must send messages consistent with some utility
  - Caveats:
    - utility chosen may be negative
    - Denial-of-service attacks easy
      - Don’t send any messages, so protocol doesn’t terminate
      - Detected cheating stops the protocol
Conclusion

◆ Start with a distributed algorithm computes mechanism with tamper-proof nodes

◆ Techniques to encourage compliance
  • Nodes save signed confirmation of msgs
  • Randomized auditing incents compliance

◆ Security against irrational agent
  • Close to model with tamper-proof algorithm