Covert Channels for Collusion in Online Computer Games

Steven J. Murdoch   Piotr Zieliński

University of Cambridge, Computer Laboratory,
15 JJ Thompson Avenue, Cambridge CB3 0FD, United Kingdom
http://www.cl.cam.ac.uk/users/{sjm217,pz215}/

6th Information Hiding Workshop
Toronto, Ontario, Canada
23 – 25 May 2004

UNIVERSITY OF CAMBRIDGE

Outline

Introduction

Competition Structures

Authentication Within Games

Real World Example

Further Work

Conclusions

UNIVERSITY OF CAMBRIDGE

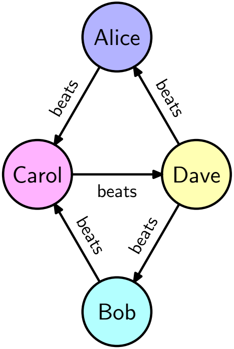
Collusion in Games

- The problem of collusion in games is well known, both between teams and within a team
- In Bridge, collusion is often permitted (within certain constraints)
- Covert channels can be used, and may be protected with a shared "key"
- Collusion needs communication, but what if communication is hard and/or disallowed?

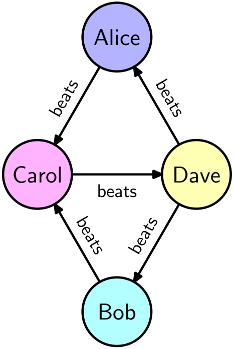
Knockout Tournaments

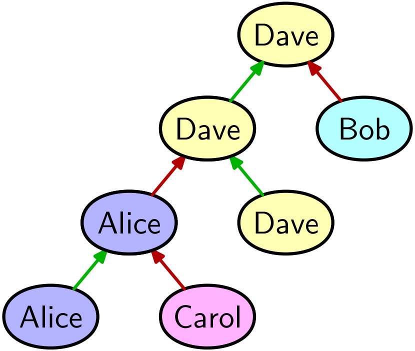
- $\lceil \log_2(n) \rceil$ rounds, losers knocked out
- Collusion is less effective, but works in certain conditions
- If the graph of results is cyclic and is known, then there are sometimes cases where one half of a colluding team should play badly for the benefit of the team

Collusion in Knockout Tournaments

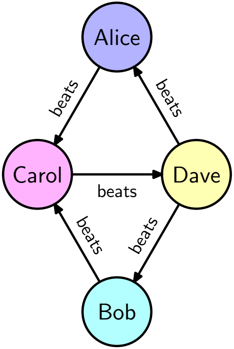


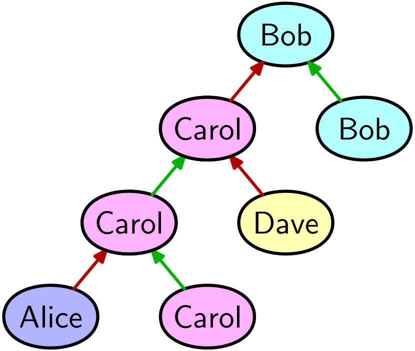
Collusion in Knockout Tournaments





Collusion in Knockout Tournaments





League Tournaments

- $n(n-1)/2$ matches, win $>$ draw $>$ lose
- How can collusion help in this kind of tournament?
- Enter multiple players, if Foxes and Chickens collude they can beat non-colluding "Optimal" players

	Fox	Chicken	Optimal
Fox	—	Fox	—
Chicken	Fox	—	—
Optimal	—	—	—

Authentication Within Games

- To collude in a league tournament, all that is necessary is authentication of the Fox by the Chickens
- In face-to-face competitions it is trivial
- Where players are programs, normal inter-process communication could be used
- Authentication is difficult a programs only sees its opponent's moves

Timing

- Well known set of techniques for covert channels in multi-level secure systems
- Modulate some system-wide property (CPU load, available memory, etc.)
- Modulate timing of moves
- Latency and jitter can reduce the capacity of the channels, even to zero

Choice of Equivalent Moves

- Known of in person-to-person games (placement of cards etc.)
- If, at a point in a game, there are multiple moves which will not change the outcome of the game then information can be carried by the move selected
- Unlike timing, moves will not be changed when sent between players
- Does not suffer from jitter, but there still can be false positives

Using a Shared Key

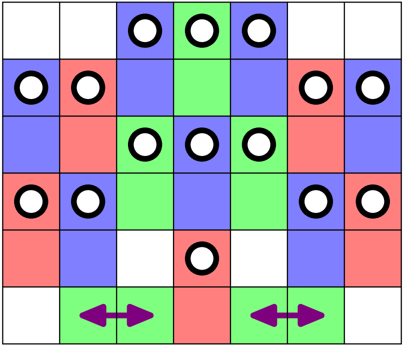
- If the sender sends a message, regardless of the coding, the receiver will receive the message without corruption (no false negatives)
- But when the receiver receives a message, how does it know that the sequence of moves is a real message (maybe some false positives)?
- Solution: sender and receiver share a key, use this to seed a pseudo-random number generator (PRNG) and use the result to select the moves
- The probability of a false positive decreases exponentially with the number of moves

Real World Example

- These techniques were developed and implemented as an entry to the Cambridge University Computing Society programming competition
- Up to 10 programs were permitted to be entered per person
- The game to be played was a variant of Connect 4, where players could pass
- First stage is a league – each program plays every other program, 2 points for a win, 1 for a draw
- Second stage is a knockout tournament, taking the top 5 from the league

Game Strategy Chosen

Since players can pass, moves cannot be forced, so to ensure a draw it is sufficient to block any winning moves:



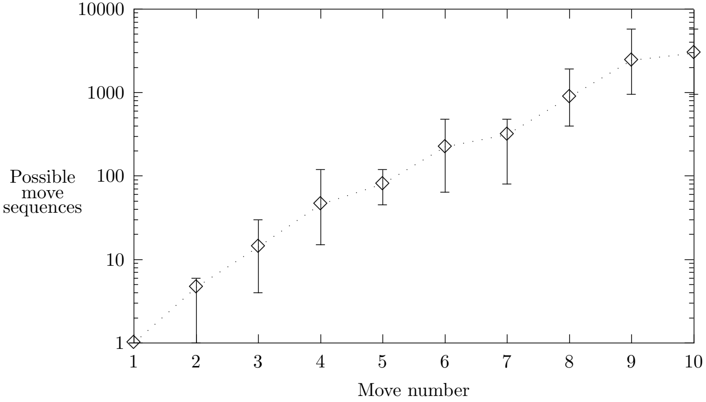
The Problem of Rabbits

- With only Chickens, Foxes and Optimal players it is clear that the fox will win
- But what if there are programs which play randomly (Rabbits)?
- Foxes and Chickens will draw against them, but the Optimal players will win
- Effectively the Rabbits will be colluding with the Optimal players
- So for the Fox to win, the Chickens must outnumber the Rabbits

Implementation

- 3 entrants, 10 programs each, 6 Foxes (to fill the knockout stage) the rest Chickens
- Versions of the program in C++, Ada95, Java and Postscript
- Linear Congruential PRNG used
- Probability of a false positive was in the range

Results (1)


10000
1000
100
10
1
Possible move sequences
1
2
3
4
5
6
7
8
9
10
Move number

Results (2)

No	Category	Won	Drew	Lost	Points
1	Fox	58	26	0	142
⋮	⋮	⋮	⋮	⋮	⋮
5	Fox	49	31	4	129
		cut-off point			
6	Fox	48	32	4	128
7	Semi-Optimal	16	67	0	99
⋮	⋮	⋮	⋮	⋮	⋮
43	Semi-Rabbit	1	52	31	54

UNIVERSITY OF CAMBRIDGE

Collusion Resistant Competitions

- Collusion may be undesirable, so can a competition be designed to prevent it?
- Finding the winner of a competition can be considered as a vote, where every voter is a candidate too
- The election should be resistant to collusion, and fair (but how can these properties be defined?)
- Can all the desirable properties be obtained at once? For elections, Arrow's Theorem says they can not

Tournaments as Elections

- Single Transferable Vote is one option
- Chickens are eliminated in early rounds, so their influence is not counted
- Final round will likely result in multiple players, all of which draw with each other — how can they be separated?
- Chickens can affect order in which players are eliminated so manipulation may still be possible

Hiding and Detecting Collusion

- Even if PRNG is used to choose moves, this is not suspicious by itself
- Does a player lose convincingly, or just throw in the towel? Defend against this by causing Chickens to lose plausibly
- Does a player's skill seem to vary a lot? Defend against this by losing probabilistically

Conclusions

- Covert channels can be found and can used for reliable authentication
- If you run (or enter) a competition, make sure you know what property of participants is *really* being tested.
- Preventing and detecting collusion is hard but may be possible
- Election design may have some answers for this problem

Final Result

Final Result

Questions?