Independent Cake Poker Investigation

November 29 2010

Thomas Bakker and Noah Stephens-Davidowitz

Contents

1 Introduction

During a period of approximately 18 months from March 2009 to August 5th 2010, the Cake Poker network did not have encryption. As a result, anyone with access to network infrastructure over which a connection between a user and Cake Poker was routed could have seen the user's secret hole cards in real-time. This vulnerability was made public on July 26th by Poker Table Ratings (see http://www.pokertableratings.com/blog/2010/07/ptr-security-alert-cake-poker-network/1).

Depending on the location and technical capabilities of a potential cheater, she could have exploited this in many ways, gaining access to

- One player's cards. The cheater might have connected to the same Wifi network as another player, for example.
- Multiple players' cards. The cheater might connect to an unsecured or poorly secured network at the site of a big poker tournament, have access to the network infrastructure of an ISP with many poker players on it, or otherwise gain access to the network traffic of a large number of poker players.
- Every player's cards. The cheater might have worked for Cake or Cake's ISP or otherwise gain access to all data between clients and Cake's servers.

We were asked by Cake Poker to look over the NL100+ hands played during these 18 months to determine if this vulnerability was exploited.

2 Data

Cake gave us all NLHE cash game hands with $1 blinds or higher during the period when the vulnerability existed. Our conclusions are only valid for these specific games. Further work might be done to look into other game types as well.

The hand histories are Cake's internal hand histories, which identify every user by her unique identification number, so the fact that Cake allows frequent name changes does not affect our investigation. Compressed in zip archives, they are approximately 19 GB, uncompressed approximately 200 GB. These files contained about 80 million NLHE hands.

Cake also provided us with a list of its employees as well as the employees of skins on the Cake network.

2.1 Verification of Completeness

We believe that it's necessary to verify that we did indeed receive all hands from the specified games and stakes during the existence of the vulnerability. To perform this verification we are currently working with Poker Table Ratings, a company that tracks hands played at many online poker sites. They are supplying us with the IDs of hands they have in their database. We then check if our database contains all of those hands. This process is ongoing. We will give an update when this finishes.

We have also put a (compressed) list online containing all hand IDs we have. This allows people with the necessary computer skills to check for themselves if we are not missing any hands. The files containing these hand IDs can be found at http://www.bakker.cc/cake/Hand_IDs/.

Also, anyone who wants to check that we have the hands they were in, please e-mail us at cake.investigation@gmail.com with an archive of the hands you played at Cake as an attachment. We will, of course, keep these hands just as confidential as we are keeping the other 80 million and will delete them when we have extracted the IDs.

None of these methods can conclusively prove that nobody tampered with the hand histories we received: Somebody might have removed hands that happened to not come up using one of the methods mentioned above, or they might have changed subtle details in hands to hide cheating. However, we don't see any way to check if this has happened, and we think it to be extremely unlikely.

3 Goal

We want to determine if anyone *significantly* exploited this vulnerability. Ideally, we would be able to catch anyone who exploited this vulnerability at all, but this is simply impossible. For example, if someone exploited this vulnerability for exactly one hand, there is simply no way to prove that that was cheating and not simply a lucky guess (or even a lucky misclick). So, we are only attempting to find *significant* cheating, by which we mean cheating that was frequent enough to be statistically distinguishable from lucky guesses.

4 Methodology

Given the public nature of the vulnerability and the history of cheaters with access to other players' hole cards being caught because of their blatantly suspicious actions (see here), we believe that a person attempting to exploit this vulnerability should have anticipated some scrutiny and would have strong incentive to be subtle about his cheating. Detecting this kind of subtle cheating is far from trivial. One cannot, for example, simply look for players with statistically improbable winrates or players whose play looks qualitatively suspicious. These methods would only work for cheaters who were not expecting to be scrutinized (the AP and UB superusers, for example). In fact, we believe that, if a cheater was sufficiently careful, he could have easily earned large amounts of money quickly while still appearing completely unremarkable to manual inspecion, winrate analysis, and many other naive methods of detection. (This seems to have occurred in the past. For example, multiple players on Poker Stars and Full Tilt reported that cheaters had installed spyware on their computers in order to see their hole cards and win large amounts of money off of them. Both sites investigated, but to our knowledge neither site issued refunds. See here.)

4.1 Main method

Our principle idea was to look at the correlation between a player's actions and his opponent's hand strength. A player with access to his opponent's hole cards should naturally vary his play with the strength of his hands in some detectable way. This seems like the most natural and fundamental way to look for such cheating.

The simplest measurement of such a correlation is the variable t in the best fit to the equation

$$\text{decision} = s + t \cdot (\text{villain hand strength})$$

Here, *decision* is defined as

action	value
fold	0
check or call	1
bet or raise	2

and *villain hand strength* is villain's equity against a random hand.

For a given player, we took all of his decisions, calculated villain's hand strength, and then fit this equation (using the least squares method of linear regression) to the data. The variable t now describes how dependent the *decisions* a player makes are on the strength of his opponent's hand. (It is roughly a measure of hand-reading ability.) In theory, cheaters should have a high value of t.

In practice, this is true, but there are better methods. We found two variations to be much more useful:

$$\text{decision} = a + b \cdot (\text{villain hand strength}) + c \cdot (\text{hero hand strength})$$

Since players tend to make stronger plays when they have a strong hand, adding in *hero hand strength* removes some variance. Here b would represent hand-reading ability. We also tried and used the following:

$$\text{decision} = d + e \cdot (\text{hero equity vs villain})$$

where *hero equity vs villain* is just the hot-and-cold equity of hero's hand against his opponent's hand.

We chose to use linear regression (as opposed to discrete regression variants) because it was simple and allowed us to easily experiment with different methodology. For example, we tried tried many different values for decision, such as

action	value
fold	0
call	1
raise	4

This specific method did not improve our findings in testing, but as you'll see, we did end up using one such variation.

4.1.1 Detection of players who can see some player's cards

The above method measures these correlations globally. However, due to the nature of the vulnerability, it is conceivable that a cheater was only able to see the hole cards of a few players. We think (and our testing confirmed) that this would usually be noticed by the above method. However, if a cheater played a sufficient amount against players whose hole cards he could not see, he would avoid such detection. To also look for these cheaters, we applied the above method to each pair of players who played together at some point. That is, for every player, we looked at these correlations separately for each of his opponents. We then looked for outliers in this data.

4.2 Secondary method

Even though we have strong faith in our primary method, we felt is was necessary to also include other tests. This was particularly important because the above methods can't be used over extremely small sample sizes, but also simply as a backup. Our secondary method looks for situations where a player makes an unexpected play that happens to be correct if he knew his opponent's cards. There are numerous ways to define such plays, and we chose one that we felt was most general. (See 6.5.)

5 Testing and Refining our Methods

To verify that our methods actually work, and to see how significantly someone would have to cheat in order to be detected, we tested them on generated, fake hand histories. We used AIs[1] to generate hand histories of many different types of players. These players included ordinary players, but also several types of cheaters:

- Players who could see the hole cards of one specific opponent.
- Players who could see the hole cards of all of their opponents.
- Players who knew certain vague characteristics of their opponents' hands, such as whether they had a draw.
- Players with either of the above capabilities that only used them in big pots, small pots, on the river, or once every few hands.

We then repeatedly performed our tests on this data, fine-tuning and refining them. We found that we were indeed able to detect very subtle forms of cheating. We were able to easily identify the cheaters that we created over relatively small sample sizes in spite of the fact that we believe almost none of the cheaters we created were detectable by previously existing methods, such as winrate analysis, manual review of hands played, or even some more basic and less direct methods of statistical analysis.

6 Procedure

6.1 Data collected

We wrote a custom parser to extract the necessary data from Cake's raw hand histories. We collected the following data for each situation in which a player had to make a decision with only one other player left in the pot:

[1] These AIs are were designed by Thomas for research purposes. They were not made to play on poker sites, nor are they able to.

Item	Description
Hero's unique id	A unique identifier for the player.
Villain's unique id	A unique identified for the other player in the hand
Decision	0 for fold, 1 for check or call, 2 for bet or raise
Facing bet	Boolean: true if player is facing a bet/raise, false otherwise
Hero hand strength	Player's equity against a random hand
Villain hand strength	Player's opponent's equity against a random hand
Hero vs. villain equity	Hero's equity against his opponent's hand
Pot size	Size of pot when decision was made
Stakes	Size of big blind
Player count	Number of players dealt into the hand
Table type	Boolean that is true if the table is six-max and false otherwise

We also collected separately for each hand the players involved, their profit or loss in the hand, the stakes, and the number of players involved in the hand.

6.2 Partitioning the data

We first divided the data into time periods: We grouped together hands from March 2009 to June 2009, July 2009 to November 2009, December 2009 to February 2010, March 2010 to May 2010 and June 2010 to August 15, 2010. We added about ten days of overlap to neighboring groups to lessen the chance that cheating was missed because it occured across a division. We also looked separately through the period when the vulnerability was public, July 26th to August 5th.

Separately for heads up, six max, and full ring games,[2] we divided players by number of decisions (into groups with between n and 1.5n decisions), discarding players with less than 200 decisions.

To lessen the chance that we missed a cheater who cheated only at high stakes or only in large pots, we similarly divided the data according to decisions made with blinds $10 and higher and in pots $100 or larger. Because of the smaller player pool at higher stakes and the relative rarity of large pots at smaller stakes, we were unable to further divide by game type, and in a few cases we combined groups (creating new groups with between n and 2.25n decisions) to get a significant sample.

6.3 Looking for players who could see multiple other players' hole cards

We calculated best fits for the equations

$$\text{decision} = a + b \cdot (\text{villain hand strength}) + c \cdot (\text{hero hand strength}),$$

$$\text{decision} = d + e \cdot (\text{hero strength vs villain})$$

[2] We defined heads up games as all games with two players dealt into the hand regardless of table type. For games with more than two players dealt into the hand, we used the maximum allowed number of players at the table instead of player count. This has the obvious problem of including some short-handed play in our full ring games. But, because players often tend to choose games based on the table type and not player count and because full ring tables often fluctuate in player count, this definition gave us larger sample sizes on more players.

as in section 4.1 under four different filters (see section 8 for a discussion of how we made these choices):

1. No filter: this includes all decisions made by a player.
2. Facing bet: includes all decisions made by a player while facing a bet or raise. [3]
3. Fold: All decisions made by a player while facing a bet with decision redefined as follows:

decision	value
fold	0
call	1
raise	1

4. No bet: All decisions made by a player when not facing a bet or raise.

We then made two plots for each filter, one of b vs. c and one of e vs. c and manually looked for outliers[4].

6.4 Looking for players who could see one other player's hole cards

As mentioned in Section 4.1.1, we also applied the above tests to all of a player's actions broken up by opponent. We gathered data in essentially the same way as above. However, we left out the "no bet" filter because we found it least necessary in testing and the run-time of this process was quite significant.

Unfortunately, we could not break up this data by sample size because most players don't have a large number of opponents with similar sample sizes to compare with each opponent. However, while the data gathered in the previous section was decidedly non-normal, the data for an individual player against each of his opponents actually looks a lot like data from normal distributions with standard deviation proportional to $\frac{1}{\sqrt{\text{sample size}}}$ and the same mean for each opponent. There is some theoretical reasoning behind why this should be true for regression coefficients provided that the player does not vary some specific aspects of his play too much against different opponents, but suffice it to say that it proved to be quite an accurate assumption. So, we first removed opponents against whom the player had made less than six raises, six calls, or twenty-six total actions (so that we didn't consider any players whose extremely small sample sizes were overly influenced by a few hands or an unusual session). We then calculated the weighted mean, $\bar{x}$, of the regression coefficient for the remaining players and "normalized" the data: If player i had regression coefficient x_i over sample size n_i, we define

$$||x_i|| = \sqrt{n_i} \cdot (x_i - \bar{x}) + \bar{x}$$

The $||x_i||$ were then normally distributed for most opponents.

[3] The unfiltered data is a bit biased because a player will only fold when facing a bet, so players who happen to face bets more will have the ability to fold more and that will affect the regression coefficients. A call is also obviously much stronger than a check, so treating them differently makes sense.

[4] Our definition of outlier was quite broad, casting quite a wide net, because false positives were not a problem for us. The data was not at all normally distributed and typically fell into clusters of players with similar styles, so automated outlier testing would not have been nearly as accurate. We were also actually able to save time by identifying outliers manually while our computers imported more data or ran other tests.

Because of this normality, we were able to use an automated test for outliers. We chose R's default implementation of the Grubbs test and looked for outliers with a less than one in 10,000 chance of occuring randomly (this number was chosen empirically during testing).

In practice, outliers were "reverse" a bit more than half the time. In other words, many of our outliers resulted from a player tending to adjust his play less dependent on this opponent's hand strength than others. We looked into some of these and found that they fairly consistently fell into two categories: Either the opponent played erratically or was fond of bluffing and slowplaying, in which case the player's standard play resulted in the outlier, or the player played erratically against the opponent (this was often a high stakes player playing lower), in which case the player's own play resulted in the outlier. A few of these were not this easily explainable, and we looked into them separately to see if they suggested any cheating.

Outliers in the more suspicious direction were always flagged as suspects.

6.5 Our secondary method

We simply looked for plays where a player folded a very strong hand when his equity was in fact quite low or called with a weak hand on the river when his hand was best (or equal to his opponent's). More specifically, we flagged a play as suspicious if it matched either of the following criteria:

1. Hero hand strength > 0.92, hero equity vs. villain $< .3$, and decision = fold
2. Hero hand strength < 0.3, hero equity vs. villain > 0, street = river, and decision = fold

where hero hand strength is again a player's equity against a random hand.

We then counted all examples of such plays to find out how often they happened. Using this number, we then calculated for each player with at least two suspicious plays the chance that a random player would have made as many suspicious plays as him in as many decisions. If the chance was less than one in one thousand, and if the player made more suspicious plays than he made incorrect plays that would have been suspicious if they worked out (i.e. similar folds with equity over 30% or river calls with the worst hand),[5] we looked at the suspicious plays manually.

Players that were indeed suspicious upon manual inspection were flagged as suspects.

6.6 Looking for chip dumping

In order to be able to elimate most suspects who had lost money, we had to be sure that they had lost the money legitimately, so we had to look for chip dumping. We felt that this was quite important, so we did this in two ways. Firstly, we looked manually at all suspects who had lost money who had played high enough stakes to have chip dumped a relatively significant amount (compared to what they lost). We also designed a fairly simple algorithm to identify chip dumping that was significant enough to influence our findings:

1. For each player, find opponents to whom he'd lost at least two large pots[6].

[5] This is quite a lax restriction. Most of the suspicious plays were river folds, and a river fold typically needs to be right a lot more than 50% of the time for it to be better than calling.

[6] Here, we defined a "large pot" as at least 25 big blinds and $400.

2. For each such opponent, if large pots totalled more than 15% of hands played between the two players, look at hands manually.
3. Otherwise, if the player lost at least $10,000 in less than 150 hands, look at the hands manually.

This algorithm worked quite nicely in practice, and we identified many chip dumpers without much effort (and reported them all to Cake on the small chance that they had yet to be caught). Only two players were identified as suspects specifically because they were chip dumping, and a few more were already suspects but became more suspicious because they chip dumped.

6.7 Handling suspects

We first discarded all suspects who were not particularly suspicious (i.e. had only been a small outlier on one test or two highly correlated tests), were flagged for being able to see all player's hole cards, had lost money or broken roughly even, had not been caught chip dumping, and did not play high enough stakes to realistically dump an amount that would make them a winning player. This was quite helpful, since the vast majority of suspects were only small outliers (due to our generous definition of outlier) and many of those were also losing players (almost any test that looks for outliers on any metric of play-style will find many players who do not play well, even if the metric approximates hand-reading ability).

Each suspect's hands were imported into Holdem Manager. We then dealt with each suspect individually, and most suspects underwent a unique screening process based on variables such as the set of tests that they came up in, the number of hands played, game type and stakes played, amount of money earned, and anything else that we found relevant. All suspects underwent the following:

- A check for opponents who lost a large amount of money to the suspect (who may have been cheated or may have participated in elaborate chip dumping, as in Section 7.1)
- A check for opponents who won a large amount of money from the suspect (who may have been partners helping to conceal winnings)
- A check for periods of larger-than-normal winnings.
- A review of winnings and winnings adjusted for all-in equity filtered by stakes and game type.
- Analysis of various statistics that might lead to further suspicion, such as river call win percentage, river aggression, river call efficiency, VPIP, and PFR

All but the least suspicious players also went through at least some manual hand review. Players who came up in the test outlined in Section 4.1.1 had all hands in which both of the suspect and his suspected victim entered the pot reviewed. Players who came up from our secondary test had all hands played against players against whom they made a suspicious play reviewed. Others had their hands reviewed based on our discretion: Those who had played relatively small samples typically had all hands in which they entered the pot reviewed manually; for those with larger samples, we reviewed an amount that we felt was sufficient given the results of our testing, typically filtering for hands that were some combination of relevant to the testing (i.e. heads up hands if the player came up only in heads up testing), played during the player's most profitable period, played against the player's most profitable opponent, and played at the player's winningest stake. We always reviewed hands under the assumption that we were looking at cheating and simply needed to prove it.

For players who remained suspicious after all of that, we turned to statistical analysis of their play. We identified common situations that the player found himself in in which either our testing or our hand review suggested that the player might have cheated. We then looked directly at how his play varied based on the strength of his opponent's hand. Essentially, we redesigned and implemented customized versions of our main method of testing for the specific player. In most cases, we were able to conclusively explain our suspicions based on perfectly normal behavior. Each situation was different, but two fairly typical examples are a player who was simply responding naturally to his opponent's very varied bet sizing and a player who had played only a few hundred hands and was an outlier because of a few not-at-all-suspicious hands that had many decisions distorted the data.

The only players who remained suspicious after all of this testing were the five elaborate chip dumpers mentioned in Section 7.1. (The colluders mentioned in Section 7.2 were identified by an entirely different method since we suspected them of collusion immediately after testing.) All of them were shown to be chip dumps by applying the process in the above paragraph to their opponents and seeing that, in addition to the fact that they knew their opponents' hole cards and used that knowledge to win, their opponents also knew their hole cards and used that knowledge to lose.

6.8 Cross-checking against employee list

Lastly, we checked to see if any player who had come to our attention was on the list of employees that Cake gave us. None were.

7 Results

We identified 187 suspects[7] and reviewed them all individually. We found no evidence of anyone exploiting the vulnerability. While it's completely impossible to prove without doubt that nobody cheated in this way (for example, a player could cheat in only one hand, and this cheating would be completely indistinguishable from a misclick or a lucky guess), we think that it's quite unlikely that anyone would have done so in a way that we wouldn't have caught.

7.1 "Elaborate" chip dumping

While we didn't find anyone who exploited the vulnerability, we did find five examples of what we've been calling "elaborate chip dumping." These were players who went to fairly extreme lengths to hide the fact that they were chip dumping, taking hundreds of hands to move relatively small amounts of money between accounts in a way that looks very much like

[7] 122 of these were from our primary method, 29 from our secondary method, and 36 from suspicious plays. Our definition of a suspect here is simply someone whose hand histories we imported into Holdem Manager. Unfortunately, this definition did vary fairly significantly throughout, though the underlying methodology did not. After taking this into account, the suspects were spaced quite evenly throughout the hand histories.

normal play. These players presumably share hole cards as they do this, and so we were able to detect them.

Perhaps the most interesting example of this was a player who won about 25 buy-ins (or about $5,000) over about 900 hands off of two separate players in numerous sessions at three different stakes (and broke about even in about 250 legitimate hands against other opponents). His play looks qualitatively well within the range of normal, and he made numerous plays that were not consistent with his goal of chip dumping–various bad river calls and bad bluffs, including many plays that look bad both with and without his opponent's hole cards showing. So, he was essentially exactly what we were looking for: a player who knew his opponent's hole cards but anticipated scrutiny and went to fairly extreme lengths to hide this fact. His cheating was further obscured by the fact that his opponents were trying to lose money to him, which was of course not something that we considered when designing our tests. However, using our main method of detection, he was clearly an extremely blatant outlier in five of our seven tests(all but the two that only looked at spots in which he didn't face a bet, which were the two that we considered to be least accurate). Nobody who didn't cheat came anywhere close to being such a large outlier.[8]

7.2 One case of collusion

Our method for detecting a player who could see only one other player's hole cards came up with a rather strange result: We had five tests identify larger outliers than we'd ever seen (two of them with such low chances of happening randomly that R simply gave them a probability of zero of having happened randomly), three of them reporting that one player could almost certainly see one of his opponent's hole cards and two others reporting that his opponent could actually also see his. We immediately suspected collusion, and a brief look at their play in three-way pots (which was actually data that our methods did not have access to) confirmed our suspicions.

Our methods weren't designed to catch collusion and would certainly fail to catch most forms of it. However, the specific way that these players tended to play once they got heads up in a hand happened to show up nicely on our tests.

We think it prudent to note that these players were quite unsuccessful colluders. They were down money when playing together, and they played on a skin that is no longer on the Cake network.

8 Conclusion

We have created, tested and applied a methodology for detecting players who had access to their opponents' cards. In our tests on generated data, our methods were able to detect even very subtle cheaters. Working with the real data further confirmed that our methods were accurate; we were able to identify several situations where a player had knowledge of his opponent's cards in cases of both collusion and chip dumping.

In the data we examined, we did not find anyone who we believe to have exploited the vulnerability. We are confident in our methods, so we believe that in the hands that we examined nobody was a victim of the vulnerability.

[8] Cake had already caught these players because two of the accounts were owned by the same person and the third was closely related.

Appendix I: Timeline

- March 2009: Vulnerability was introduced.
- July 26, 2010: PokerTableRatings.com reported the vulnerability to the public.
- August 5, 2010: Cake Poker fixed the vulnerability.
- August 16, 2010: We signed a contract to initiate this investigation.
- October 8, 2010: We received all the hands from NL100+.
- November 23, 2010: We finished our investigation.

Appendix II: Critique and Alternatives

Throughout the course of this investigation, we considered a wide variety of methods of detection. We feel that we chose wisely, but we acknowledge that our methods are imperfect and that there exist viable alternatives. We wanted to address some of the more difficult choices that we made. One frequent theme is that we felt strongly that we should avoid methods that would only catch cheaters with a specific "style" of cheating or that made broad assumptions about how cheaters would play.

The inherent flaw in using statistics

Whenever anyone uses statistics, he is essentially assuming that extremely unlikely events don't happen. In this case, there is some extremely small chance that some player cheated in a way that we fully anticipated and expected to catch would have gone undetected simply because of the randomness of the deck and the use of statistics in our methodology. Statistics is an extremely powerful tool, but aces still sometimes get cracked, people still sometimes win the lottery, and extremely stable particles still sometimes decay. Anyone using statistical analysis must consider that they are simply the equivalent of a person who watched one atom of uranium for one second, saw it decay, and concluded that this was a likely event.

Different filters for our main method

We originally planned to use different filters in our testing. Instead of running both regressions on four different filters, we intended to run specific regressions on various specific filters in accordance with how we assumed cheaters would cheat.

For example, we planned to run a regression in which we treated calls, checks and folds equally in spots in which a player was far behind his opponent. The idea here is that a player who knew his opponent's hand would be much more likely to bet in these situations when his opponent was weak (for example, if he had three-high to his opponent's four-high) and much more likely to do something else when his opponent was strong. So, we anticipated finding a strong negative correlation between opponent hand strength and decision from cheaters in these spots.

Most of these tests did work in testing with our AIs, but they weren't as decisive as we'd hoped. They also significantly lowered sample size, and they often lowered sample size in ways that varied hugely with the suspect's style of play (in the above example, a loose, passive player will find himself in many many more such situations than a tight, aggressive player). Most importantly, they made implicit assumptions about how a cheater would choose to play. While someone who can see another player's hole cards will inevitably have to show some correlation between his play and his opponent's hand, he may not do so in a way that we anticipate. In the above example, he may choose to frequently float in such spots or he may simply choose other spots to use his advantage.

Bet sizing

Perhaps the strongest potential criticism of our method is that we ignored bet sizing in our analysis. We had strong practical considerations for doing this. Adding bet-sizing would have complicated our model significantly. In our view, there is no natural way to consider it, and we worried that whatever method we did choose risked adding bias unintentionally. Perhaps more importantly, it would have made our AI testing practically useless because our AIs do not vary their bet sizes (and changing that in a reasonable way would be an impractically large project).

This created the risk that our data might have been completely corrupted by bet-sizing tells. There could have been a very large class of players who made their hand strength clear based on their bet sizing and another relatively large class of players who exploited this fact and therefore came up as cheaters in our test. This did not worry us too much because we were prepared to deal with a large number of false positives, and if necessary to create a secondary test to filter out some of the more blatant data created by bet-sizing tells. In practice, this never really materialized. For whatever reason, bet-sizing tells just didn't seem to have much of an effect on our data.

A larger risk is that a player might cheat through bet-sizing. There are a number of ways that he might do this. In the extreme case, one could consider a cheater who cheats quite blatantly but to our methodology looks completely innocent. For example, every time he has a hand that he wouldn't bet if he knew his opponent's hand but would otherwise, he would minbet. We think that such a style of cheating is extremely unlikely, requiring a cheater who went out of his way to avoid detection in a way that happened to be specifically designed to avoid our testing (which was of course designed after the vulnerability was fixed). However, of all the specific examples of cheating that would avoid detection, we find this to be the most troublesome.

Changes to our secondary method

Our secondary method defines a suspicious play much less specifically than most suggestions implied. Indeed, the hands that our suspicious method flagged are often from one of a few categories that most people probably wouldn't find very suspicious: folds of relatively hands to overbets, A-high calls after flop and turn check through on boards with many straights or flushes possible, calls of very small bets with high card hands, and folds of strong hands to an opponent showing a lot of strength (i.e. folding to a river 4-bet or bet/folding the second nut flush on a four-flush board).

Many people suggested very specific lines to look for (i.e. "Hero bets flop and turn and then check/raises river when his opponent misses a draw"), and looking for such lines would largely cure the problem of calling not-very-suspicious hands suspicious. However, any such definition cuts down on sample size dramatically and makes a large assumption about how our suspected cheater might try to cheat. We could try to come up with a very large number so as to consider all possible types of cheaters, but that is an extremely difficult problem, comparable to that of solving the game of poker itself.

We could also have filtered out specific situations that we saw frequently. But, these weren't much of a problem for us (as the number of suspects from this test didn't slow us down significantly) and applying any such filters (for example, systematically ignoring hands in which more than two bets went in on the river before the "suspicious" fold) would complicate our model and risk ignoring truly suspicious and blatant activity. Indeed, many of the hands that we determined manually would likely have been missed if we'd decided to apply such filters (though all of these hands turned out to be anomalous).

Using preflop data

We originally wanted to use preflop data. However, the problem with this is simply that it's not clear how a cheater would exploit knowledge of preflop hands. Some cheaters might fold dominated hands preflop, but most probably would recognize that their postflop edge would more than make up for that (and indeed even without an edge gained from cheating, factors such as betting impetus and position are often much more important than actual pot equity). Some might be quite aggressive preflop, trying to move opponents off weak hands whenever possible, but most would probably prefer seeing a flop to further exploit their edge. Some might see an extremely high percent of flops, but others may prefer to be less obvious. So, we decided that while we'd like to use the preflop data, there's really not a good way to use it.

Ignoring this data leaves us vulnerable to a cheater who only cheated preflop. However, we think it quite implausible that a cheater would be able to do this as this would either require him to forget information that he already knows (preflop, he might know his opponent has 66 vs. his AA, but on the flop he must ignore this information if a six comes) or to only play preflop (which would obviously be a huge handicap).

If we were to look for similar cheating in tournaments, we'd have to devise a method to account for the case in which a player only plays preflop, since in many tournaments, the majority of play is preflop.

AI analysis

One clever recommendation that we got from forum member Aaron Brown was to use our AIs to detect cheaters. The basic idea that he proposed was to create an AI that cheated and one that didn't and have each AI analyze each decision made by each player in the database. If a player agreed with the cheating AI significantly more than the one that didn't cheat, she would be a suspect.

We originally liked this idea, however we eventually came to realize that it would not work. The problem is that an AI cheats relatively subtly, it would be extremely unlikely to find a real cheater with the same pattern of obfuscation. If it does so blatantly, then its play will also have a lot of other differences from the play of a non-cheating AI that would not necessarily indicate cheating–It would, for example, bluff and slowplay much more often. We feel that these problems are unavoidable, and as a result, we'd be unlikely to find cheaters through this method, but we would identify many innocent players who either played very differently from our basic AI or happened to play similarly to our cheating AI.